UNIT-I

INTRODUCTION:

Human–computer interaction (HCI), alternatively man–machine interaction (MMI) or computer–human interaction (CHI) is the study of interaction between people (users) and computers.

With today's technology and tools, and our motivation to create really effective and usable interfaces and screens, why do we continue to produce systems that are inefficient and confusing or, at worst, just plain unusable? Is it because:

1. We don't care?
2. We don't possess common sense?
3. We don't have the time?
4. We still don't know what really makes good design?

DEFINITION:

"Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them."

GOALS:

- A basic goal of HCI is
  - to improve the interactions between users and computers
  - by making computers more usable and receptive to the user's needs.
- A long term goal of HCI is
  - to design systems that minimize the barrier between the human's cognitive model of what they want
  - to accomplish and the computer's understanding of the user's task

WHY HCI IS IMPORTANT:

- User-centered design is getting a crucial role!
- It is getting more important today to increase competitiveness via HCI studies (Norman, 1990)
- High-cost e-transformation investments
- Users lose time with badly designed products and services
- Users even give up using bad interface
  - Ineffective allocation of resources.

User Experience (UX) Design: Students will learn about the principles and methods of UX design and how to apply them to real-world problems.
Human-Computer Interaction (HCI) Research: Students will learn about the latest research in HCI and how to design and conduct studies to investigate HCI-related topics

DEFINING THE USER INTERFACE:

User interface, design is a subset of a field of study called *human-computer interaction*(HCI).
Human-computer interaction is the study, planning, and design of how people andcomputers 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, physical limitations and abilities people possess,
- how information processing systems work,
- what people find enjoyable and attractive.
- Technical characteristics and limitations of the computer hardware and softwaremust also be considered.

The *user interface* is to the part of a computer and its software that people can see, hear, touch, talkto, or otherwise understand or direct. The user interface has essentially two components: input and output.*Input* is how a person communicates his / her needs to the computer.
Some common input components are the keyboard, mouse, trackball, one's finger, and one's voice.
Output is how the computer conveys the results of its computations and requirements to the user.
Today, 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.

The use of the human senses of smell and touch output in interface design still remain largely unexplored.
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.

THE IMPORTANCE OF GOOD DESIGN:

With today's technology and tools, and our motivation to create really effective and usable interfaces and screens, why do we continue to produce systems that are inefficient andconfusing or, at worst, just plain unusable? Is it because:

• We don't care?
• We don't possess common sense?

• We don't have the time?
• We still don't know what really makes good design?

But we never seem to have time to find out what makes good design, nor to properly apply it. After all, many of us have other things to do in addition to designing interfaces and screens.

So we take our best shot given the workload and time constraints imposed upon us. The result, too often, is woefully inadequate.
Interface and screen design were really a matter of common sense, we developers would have been producing *almost identical* screens for representing the real world.

Example bad designs

– Closed door with complete wood
– suggestion : glass door

THE IMPORTANCE OF THE USER INTERFACE:
A well-designed interface and screen is terribly important to our users. It is their window to view the capabilities of the system.

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:

Poor clarity forced screen users to spend one extra second per screen.

- Almost one additional year would be required to process all screens.
- Twenty extra seconds in screen usage time adds an additional 14 personyears.

The benefits of a well designed screen have also been under experimental scrutinyfor many years.

- One researcher, for example, attempted to improve screen clarity and readability by making screens less crowded.
- Separate items, which had been combined on the same display line to conserve space, were placed on separate lines instead.

- The result screen users were about 20 percent more productive with the lesscrowded version.

Proper formatting of information on screens does have a significant positive effecton performance.

- In recent years, the productivity benefits of well-designed Web pages have also been scrutinized.

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.
Ultimately, that an organization's customers benefit because of the improvedservice they receive.
Identifying and resolving problems during the design and development process also has significant economic benefits
How many screens are used each day in our technological world?
How many screens are used each day in your organization? Thousands? Millions?
Imagine the possible savings. Proper screen design might also, of course, lower thecosts of replacing "broken" PCs.

A BRIEF HISTORY OF THE HUMAN-COMPUTER INTERFACE:

- The need for people to communicate with each other has existed since we first walkedupon this planet.
- The lowest and most common level of communication modes we share are movements and gestures.
- Movements and gestures are language independent, that is, they permit people who do not speak the same language to deal with one another.
- The next higher level, in terms of universality and complexity, is spoken language.
- Most people can speak one language, some two or more. A spoken language is a very efficient mode of communication if both parties to the communication understand it.
- At the third and highest level of complexity is written language. While most people speak, not all can write.
- But for those who can, writing is still nowhere near as efficient a means of communication as speaking.
- In modem times, we have the typewriter, another step upward in communication complexity.
- Significantly fewer people type than write. (While a practiced typist can find typing faster and more efficient than handwriting, the unskilled may not find this the case.)
- Spoken language, however, is still more efficient than typing, regardless' of typing skill level.
- Through its first few decades, a computer's ability to deal with human communication was inversely related to what was easy for people to do.
- The computer demanded rigid, typed input through a keyboard; people responded slowly using this device and with varying degrees of skill.
- The human-computer dialog reflected the computer's preferences, consisting of one style or a combination of styles using keyboards, commonly referred to as Command Language, Question

and Answer, Menu selection, Function Key Selection, and Form Fill-In.

- Throughout the computer's history, designers have been developing, with varying degrees of success, other human-computer interaction methods that utilize more general, widespread, and easier-to-learn capabilities: voice and handwriting.
- Systems that recognize human speech and handwriting now exist,although they still lack the universality and richness of typed input.

INTRODUCTION OF THE GRAPHICAL USER INTERFACE:

The Xerox systems, Altus and STAR, introduced the mouse and pointing andselecting as the primary human-computer communication method.

The user simply pointed at the screen, using the mouse as an intermediary.

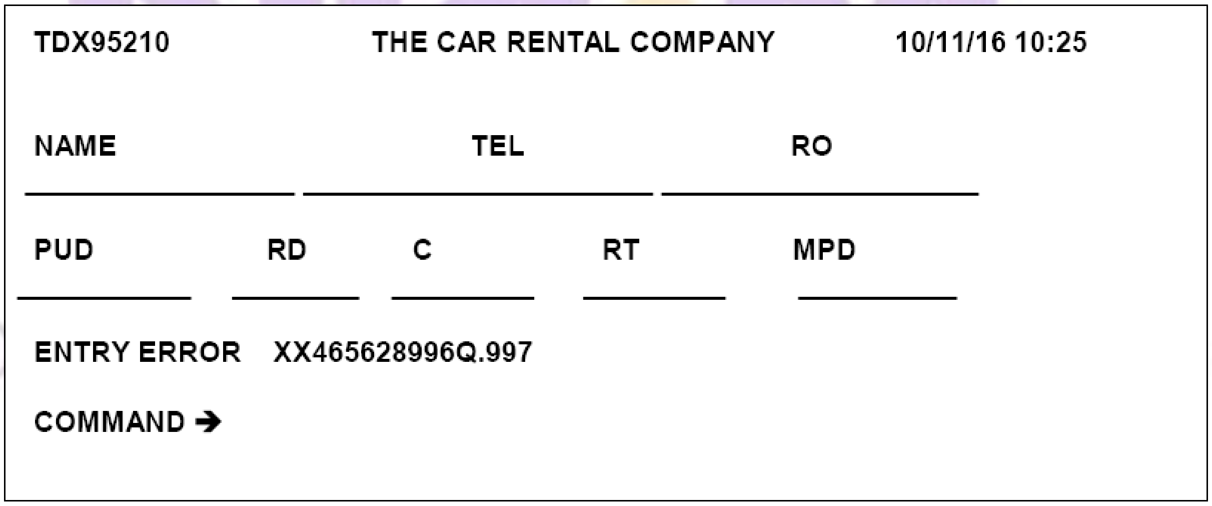
These systems also introduced the graphical user interface as we know it a newconcept was born, revolutionizing the human-computer interface.

A BRIEF HISTORY OF SCREEN DESIGN:

While developers have been designing screens since a cathode ray tube display was first attached to a computer, more widespread interest in the application of good design principles to screens did not begin to emerge until the early 1970s, when IBM introduced its 3270 cathode ray tube text-based terminal.

A 1970s screen often resembled the one pictured in Figure.

It usually consisted of many fields (more than are illustrated here) with very cryptic and often unintelligible captions.


TDX95210 THE CAR RENTAL COMPANY 10/11/16 10:25

NAME TEL RO

PUD RD C RT MPD

ENTRY ERROR XX465628996Q.997

COMMAND ➔

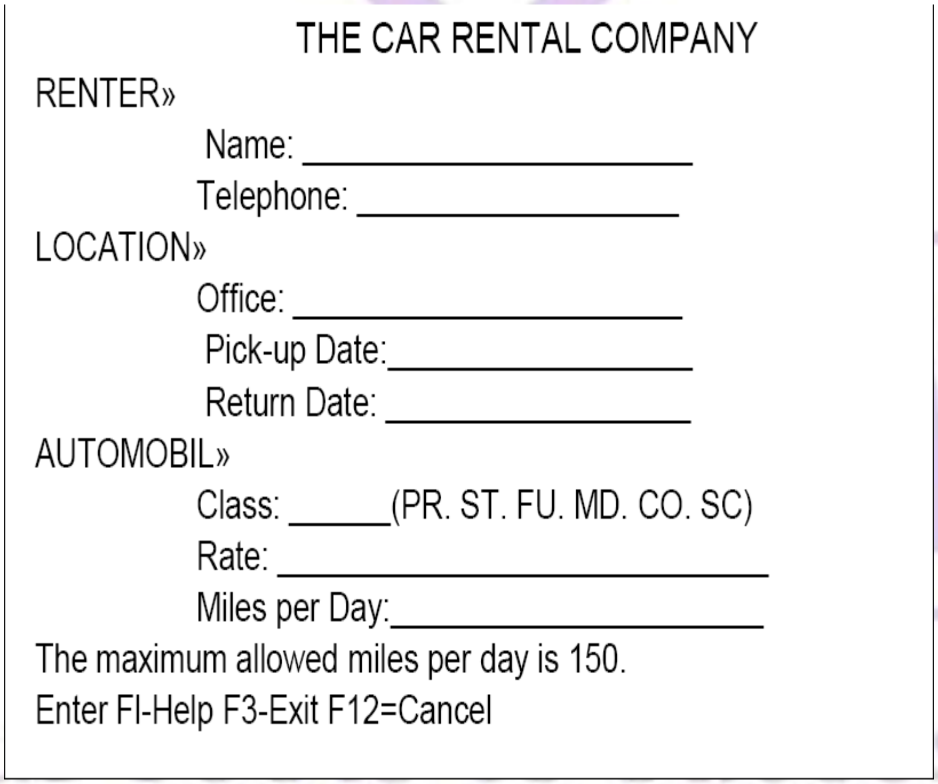
It was visually cluttered, and often possessed a command field that challenged the user to remember what had to be keyed into it.

Ambiguous messages often required referral to a manual to interpret.
Effectively using this kind of screen required a great deal of practice and patience.
Most early screens were monochromatic, typically presenting green text on black backgrounds.
At the turn of the decade guidelines for text-based screen design were finally made widely available and many screens began to take on a much less cluttered look through concepts such as grouping and alignment of elements, as illustrated in Figure1.2.

User memory was supported by providing clear and meaningful field captions and bylisting commands on the screen, and enabling them to be applied, through function keys. Messages also became clearer.

These screens were not entirely clutter-free, however. Instructions and reminders to the user had to be inscribed on the screen in the form of prompts or completion aids such as the codes PR and Sc.

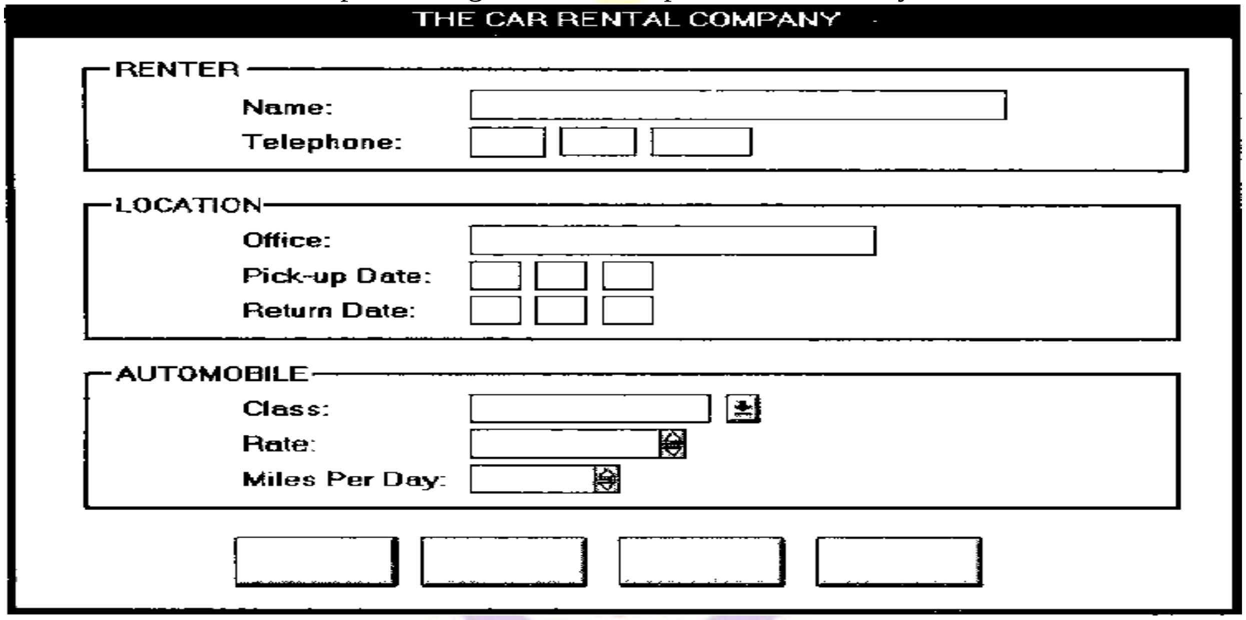
Not all 1980s screens looked like this, however. In the 1980s, 1970s-type screens were still being designed, and many still reside in systems today.


THE CAR RENTAL COMPANY

RENTER»
Name: ______________________
Telephone: __________________
LOCATION»
Office: ______________________
Pick-up Date:_________________
Return Date: _________________
AUTOMOBIL»
Class: ______(PR. ST. FU. MD. CO. SC)
Rate: ____________________________
Miles per Day:______________________
The maximum allowed miles per day is 150.
Enter FI-Help F3-Exit F12=Cancel

The advent of graphics yielded another milestone in the evolution of screen design, asillustrated in Figure above While some basic "design principles did not change, groupings and

Alignment, for example,
Borders were made available to visually enhance groupings and buttons and menus for Implementing commands replaced function keys.



Multiple properties of elements were also provided, including many different font sizesand styles, line thicknesses, and colors.
The entry field was supplemented by a multitude of other kinds of controls, includinglist boxes, drop-down combination boxes, spin boxes, and so forth.
These new controls were much more effective in supporting a person's memory, nowsimply allowing for selection from a list instead of requiring a remembered key entry.
Completion aids disappeared from screens, replaced by one of the new listing controls.
Screens could also be simplified, the much more powerful computers being ableto quickly present a new screen.
In the 1990s, our knowledge concerning what makes effective screen design continued to expand. Coupled with ever-improving technology, the result was even greater improvements in the user-computer screen interface as the new century dawned.

THE POPULARITY OF GRAPHICS:

- A graphical screen bore scant resemblance to its earlier text-based colleagues.
- Older text-based screen possessed a one dimensional
- Graphic screens assumed a three-dimensional look.
- Controls appeared to rise above the screen and move when activated.
- Information could appear, and disappear, as needed.
- Text could be replaced by graphical images called icons.
- These icons could represent objects or actions

- 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 dropdown menus of.
- Objects and actions were selected through use of pointing mechanisms.
- Increased computer power.
- User's actions to be reacted to quickly, dynamically, and meaningfully.
- WIMP interface: windows, icons, menus, and pointers.
- Graphic presentation is much more effective 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;
- Graphics also can add appeal or charm to the interface and permit greater customization to create a unique corporate or organization style.

GRAPHICAL SYSTEMS ADVANTAGES AND DISADVANTAGES

- Reduce the memory requirements.
- More effective use of one's information.
- Dramatically reduce system learning requirements.
- Experience indicates that for many people they have done all these things.

ADVANTAGES

- Symbols recognized faster than text
- Faster learning
- Faster use and problem solving
- Easier remembering
- More natural
- Exploits visual/spatial cues
- Fosters more concrete thinking
- Provides context
- Fewer errors
- Increased feeling of control
- Immediate feedback
- Predictable system responses
- Easily reversible actions
- Less anxiety concerning use
- More attractive
- May consume less space

- Replaces national languages
- Easily augmented with text displays
- Smooth transition from command language system.

DISADVANTAGES

- Greater design complexity.
- Learning still necessary
- Replaces national languages
- Easily augmented with text displays
- Smooth transition from command language system
- Lack of experimentally-derived design guidelines
- use a pointing device may also have to be learned
- Working domain is the present
- Human comprehension limitations
- Window manipulation requirements
- Production limitations
- Few tested icons exist
- Inefficient for touch typists
- Inefficient for expert users
- Not always the preferred style of interaction
- Not always fastest style of interaction
- Increased chances of clutter and confusion
- May consume more screen space
- Hardware limitations

THE CONCEPT OF DIRECT MANIPULATION:

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 simply 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 natural ways. One problem in direct manipulation, however, is that 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 controlpanel, a familiar concept to most people, in a window's border.
This control panel is manipulated, not the window itself. Actions are rapid and incremental with visible display of results, 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.

EARLIER DIRECT MANIPULATION SYSTEMS:

- The concept of direct manipulation actually preceded the first graphical system. The earliest full-screen text editors possessed similar characteristics.
- 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. It took the advent of graphical systems to crystallize the direct manipulation concept, however.

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 maybe 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 forpointing.
- 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.

- Which style of interaction-direct manipulation, indirect manipulation, or a combination of both-is best, under what conditions and for whom, remains a question whose answer still eludes us.

CHARACTERISTICS OF THE GRAPHICAL USER INTERFACE:

A graphical system possesses a set of defining concepts. Included are sophisticated visual Presentation, pick-and click interaction, a restricted set of interface options, visualization, object orientation, extensive use of a person's recognition memory, and concurrentperformance of functions

Sophisticated Visual Presentation:

Visual presentation is the visual aspect of the interface. It is 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 photograph 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, pull down, pop- up, cascading), icons to represent objects such as programs or files, assorted screen-based controls (text boxes, list boxes, combination boxes, settings, scroll bar and buttons), and a mouse pointer and cursor.

The objective is to reflect visually on screen the real world of the user as realistically, meaningfully, simply, and clearly possible.

A graphical system possesses a set of defining concepts. Included are sophisticated visual presentation, pick-andclick interaction, a restricted set of interface options, visualization, object orientation, extensive use of a person's recognition memory, and concurrentperformance of functions.

Restricted Set of Interface Options: The array of alternatives available to the user is whatis presented on the screen or may be retrieved through what is presented on the screen, nothing less, nothing more. This concept fostered the acronym WYSIWYG.

Pick-and-Click Interaction: Elements of a graphical screen upon which some action is tobe performed must first 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 cue.
The primary mechanism for performing this pick-and-click is most often themouse 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 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.

Visualization: Visualization is a cognitive process that allows people to understand .Information that is difficult to perceive, because it is either too voluminous or too abstract

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 really graphical image, but toproduce one that conveys the most relevant information.

Effective visualizations can facilitate mental insights, increase productivity, and for faster and more accurate use of data.

Object Orientation: A graphical system consists of objects and actions. Objects are what people see on screen. They are manipulated as a single unit.

Objects can be composed of sub objects. For example, an object may be a document. The document's sub objects may be a paragraph, sentence, word, and letter.

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. Similar traits and behaviors exist in objects of the same object 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 changesit.

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 eachapplication.

Data may also be transferred between programs. It may be temporarily stored on a" clipboard" for later transfer or be automatically swapped between programs.

THE GRAPHICAL USER INTERFACE:

- A user interface is a collection of techniques and mechanisms to interact withsomething.
- In a graphical interface the primary interaction mechanism is a pointing device ofsome kind.
- This device is the electronic equivalent to the human hand. What the user interactswith is 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. Allobjects have standard resulting behaviors.

THE WEB USER INTERFACE:

The expansion of the World Wide Web since the early 1990s has been truly amazing. Once simply a communication medium for scientists and researchers, its many and pervasive tentacles have spread deeply into businesses, organizations, and homes around the world.
Unlike earlier text-based and GUI systems that were developed and nurtured in an organization's Data Processing and Information Systems groups, the Web's roots weresown in a market-driven society thirsting for convenience and information.

Web interface design is essentially the design of navigation and the presentation of information. It is about content, not data.

Proper interface design is largely a matter of properly balancing 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 is difficult for a number of reasons. First, its underlying design language, HTML, was never intended for creating screens to be used by the general population.

Its scope of users was expected to be technical. HTML was limited in objects and interaction styles and did not provide a means for presenting information in the most effective way for people.

Next, 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-towhere" is often unremembered or unknown.

Web interface design is also more difficult because the main issues concern information Architecture and task flow, neither of which is easy to standardize.

It is more difficult because of the availability of the various types of multimedia, and the desire of many designers to use something simply because it is available.

It is more difficult because users are ill defined, and the user's tools so variable in nature.

The ultimate goal of a Web that feels natural, is well structured, and is easy to use will reach fruition.

THE POPULARITY OF THE WEB:

While the introduction of the graphical user interface revolutionized the user interface, the 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 non secure channels or whether to accept or refuse cookies. Web usage has reflected this popularity. The number of Internet hosts has risendramatically:

- In 1984, hosts online exceeded 1,000;
- in 1987, 10,000;
- in 1989, 100,000,
- in 1990, 300,000;
- in 1992 hosts exceeded one million.

Commercialization of the Internet saw even greater expansion of the growth rate. In 1993, Internet traffic was expanding at a 341,634 percent annual growth rate. In 1996, there were nearly 10 million hosts online and 40 million connected people (PBS Timeline).

- User control has had some decided disadvantages for some Web site owners as well.
- Users have become much more discerning about good design.
- 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.
- People are quick to vote with their mouse, and these warnings should not go unheeded.

GUI VERSUS WEB PAGE DESIGN:

- GUI and Web interface design do have similarities. Both are software designs, they areused by people, they are interactive, they are heavily visual experiences presented through screens, and they are composed of many similar components.
- Significant differences do exist.

CONCEPT GUI WEB:

- 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.

GRAPHICAL USER INTERFACE:

- User hardware variations limited
- User hardware characteristics well defined.
- Screens appear exactly as specified.
- 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:
    - Install, configure, personalize, start, use, and upgrade programs.
    - Open, use, and close data files.
    - Fairly long times spent within an application. Familiarity with applications often achieved.
    - Controlled and constrained by program.
    - Windows, menus, controls, data, tool bars, messages, and so on.
    - Many transient, dynamically appearing and disappearing.
    - Presented as specified by designer. Generally standardized by toolkits and style guides

- Through menus, lists, trees, dialogs, and wizards. Not a strong and visible concept.
- Constrained by design.
- Generally standardized by toolkits and
- Style guides. User Focus • Data and applications • Information and navigation
- Enables maintenance of a better sense of context. Restricted navigation paths.
- Multiple viewable windows Interactions such as clicking menu choices, pressing buttons, selecting list choices, and cut/copy/paste occur within context of active program.
- Nearly instantaneous.
- Typically prescribed and constrained by toolkit.
- Visual creativity allowed but difficult.
- Little significant personalization.
- Unlimited capability proportional to sophistication of hardware and software. Targeted to a specific audience with specific tasks. Only limited by the amount of programming undertaken to support it
- 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.
- Integral part of most systems and applications. Accessed through standard mechanisms. Documentation, both online and offline,
- Usually provided.
- Personal support desk also usually provided
- Seamless integration of all applications into the platform environment a major objective.
- Toolkits and components are key elements in accomplishing this objective
- Tightly controlled in business systems, proportional to degree of willingness to invest resources and effort

WEB:

- User hardware variations enormous.
- Screen appearance influenced by hardware being used.
- Information and navigation
- 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
- Link to a site, browse or read pages, fill out forms, register for services, participate in transactions, download and save things.
- Movement between pages and sites very rapid. Familiarity with many sites not established.
- Infinite and generally unorganized.
- 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
- 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 an lack of consistency
- Poorer maintenance of a sense of context. Single-page entities.
- Unlimited navigation paths.
- Contextual clues become limited or are difficult to find.
- Basic interaction is a single click. This can cause extreme changes in context, which may not be noticed.
- Quite variable, depending on transmission speeds, page content, and so on. Long times can upset the user
- Fosters a more artistic, individual, and unrestricted presentation style.
- Complicated by differing browser and display capabilities, and bandwidth limitations.
- Limited personalization available.
- 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
- No similar help systems.
- The little available help is built into the page. Customer service support, if provided, oriented to product or service offered.
- Apparent for some basic functions within most Web sites (navigation, printing,and so on.)
- Sites tend to achieve individual distinction rather than integration.
- Susceptible to disruptions caused by user, telephone line and cable providers, Internet service providers, hosting servers, and remotely accessed sites.

PRINCIPLES OF USER INTERFACE DESIGN:

- An interface must really be just an extension of a person. This means that 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 than the previously used method or tool did.
- It must also be easy to learn, for people want to do, not learn to do.
- Finally, the system must be easy and fun to use, evoking a sense of pleasure and accomplishment not tedium and frustration.
- The interface itself should serve as both a connector and a separator
- a connector in that it ties the user to the power of the computer, and a separator in that it minimizes the possibility of the participants damaging one another.
- While the damage the user inflicts on the computer tends to be physical (a frustrated pounding of the keyboard), the damage caused by the computer is more psychological.
- Throughout the history of the human-computer interface, various researchers and writers have attempted to define a set of general principles of interface design.
- What follows is a compilation of these principles. They reflect not only what we know today, but also what we think we know today.
- Many are based on research, others on the collective thinking of behaviorists workingwith user interfaces.
- These principles will continue to evolve, expand, and be refined as our experience with Gills and the Web increases.

PRINCIPLES FOR THE XEROX STAR:

The 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.
Displaying objects that are selectable and manipulable must be created.

A design challenge is to invent a set of displayable objects that are represented meaningfully and appropriately for the intended application.
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. Standalone icons easily fulfilled this requirement.
The handles for windows were placed in the borders.

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. 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 viewerattention, it is a useful mechanism for providing this feedback (by changing shapes).

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.

- 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.
- 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?
- 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 described below.
They are fundamental to the design and implementation of all effective interfaces, including GUI and Web ones.
These principles are general characteristics of the interface, and they apply to allaspects.
The compilation is presented alphabetically, and the ordering is not intended to imply degree of importance.

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.

Clarity

The interface should be visually, conceptually, and linguistically clear, including

• Visual elements
• Functions
• Metaphors

• Words and Text

Compatibility

Provide compatibility with the following:

- The user
- The task and job
- The Product

Adopt the User's Perspective

Configurability

Permit easy personalization, configuration, and reconfiguration of settings.

- Enhances a sense of control
- Encourages an active role in understanding

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 asensible order that is easy to recollect and place in context.

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.

Control

The user must control the interaction.

- Actions should result from explicit user requests.
- Actions should be performed quickly.
- Actions should be capable of interruption or termination.
- The user should never be interrupted for errors

- The context maintained must be from the perspective of the user.

- The means to achieve goals should be flexible and compatible with the user's skills,experiences, habits, and preferences.
- Avoid modes since they constrain the actions available to the user.
- Permit the user to customize aspects of the interface, while always providing aProper set of defaults

Directness

Provide direct ways to accomplish tasks.

- Available alternatives should be visible.
- The effect of actions on objects should be visible.

Flexibility

A system must be sensitive to the differing needs of its users, enabling a level and type ofperformance 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.

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.

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.

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.

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.

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

Responsiveness

The system must rapidly respond to the user's requests Provide immediate acknowledgment for all user actions:

- Visual.
- Textual
- Auditory.

Transparency

Permit the user to focus on the task or job, without concern for the mechanics of theinterface. Workings and reminders of workings inside the computer should be invisible tothe user.

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 madeharder.
- Provide uniformity and consistency.

UNIT-II

OBSTACLES AND PITFALLS IN DEVELOPMENT PATH:

Developing a computer system is never easy. The path is littered with obstacles and traps, many of them human in nature. Gould (1988) has made these general observations about design:

- Nobody ever gets it right for the first time
- Development is chock full of surprises.
- Good design requires living in a sea of changes.
- Designers need good tools.
- Performance design goals
- People may make mistakes while using a good system also

COMMON PITFALLS:

- No early analysis and understanding the users needs and expectations.
- A focus on using design features or components .
- No usability testing.
- No common design team vision.
- Poor communication

COMMON USABILITY PROBLEMS:

- Ambiguous menus and icons.
- Languages that permit only single direction movement through a system.
- Input and direct manipulation limits.
- Complex linkage.
- Inadequate feedback.
- Lack of system anticipation.
- Inadequate error messages.

IRRITATING CHARACTERS:

- Visual clutter
- Impaired information readability
- Incomprehensible components
- Annoying distractions.
- Confusing navigation.
- Inefficient operations
- Inefficient page scrolling.
- Information overload

DESIGN TEAM:

- Development
- Human factors
- Visual Design
- Usability assesment
- Documentation
- Training

HUMAN INTERACTION WITH COMPUTERS:

Understanding How People Interact with Computers Characteristics of computer systems, past and present, that have caused, and are causing, people problems. We will then look at the effect these problems have –

- Why people have trouble with computers
- Responses to poor design
- People and their tasks

Why People Have Trouble with Computers:

- Extensive technical knowledge but little behavioral training.
- With its extensive graphical capabilities.
- Poorly designed interfaces.
- What makes a system difficult to use in the eyes of its user?
- Use of jargon
- Non-obvious design
- Fine distinctions
- Disparity in problem-solving strategies
- an "error-preventing" strategy
- Design inconsistency

PSYCHOLOGICAL:

Typical psychological responses to poor design are:

Confusion: Detail overwhelms the perceived structure. Meaningful patterns are difficult to ascertain, and the conceptual model or underlying framework cannot be understood or established.

Annoyance: Roadblocks that prevent a task being completed, or a need from being satisfied, promptly and efficiently lead to annoyance. Inconsistencies in design, slow computer reaction times, difficulties in quickly finding information, outdated information, and visual screen distractions are a few of the many things that mayannoy users.

Frustration: An overabundance of annoyances, an inability to easily convey one's intentions to the computer, or an inability to finish a task or satisfy a need can causefrustration. Frustration is heightened if an unexpected computer response cannot be undone or if what really took place cannot be determined: Inflexible and unforgiving systems are a major source of frustration.

Panic or stress: Unexpectedly long delays during times of severe or unusual pressure may introduce panic or stress. Some typical causes are unavailable systems or long response times when the user is operating under a deadline or dealing with an irate customer.

Boredom: Boredom results from improper computer pacing (slow response times or long download times) or overly simplistic jobs.

These psychological responses diminish user effectiveness because they are severe blocks to concentration.

--Thoughts irrelevant to the task at hand are forced to the user's attention,and necessary concentration is impossible.

--The result, in addition to higher error rates, is poor performance, anxiety,and dissatisfaction Physical.

Psychological responses frequently lead to, or are accompanied by, the following physical reactions.

Abandonment of the system: The system is rejected and other information sources are relied upon. These sources must, of course, be available and the user must have the discretion to perform the rejection.

In business systems this is a common reaction of managerial and professional personnel. With the Web, almost all users can exercise this option.

Partial use of the system: Only a portion of the system's capabilities are used, usually those operations that are easiest to perform or that provide the most benefits. Historically, this has been the most common user reaction to most computer systems. Many aspects of many systems often go unused.

Indirect use of the system: An intermediary is placed between the would-be user and the computer. Again, since this requires high status and discretion, it is another typical response of managers or others with authority.

Modification of the task: The task is changed to match the capabilities of the system. This is a prevalent reaction when the tools are rigid and the problem is unstructured, as in scientific problem solving.

Compensatory activity: Additional actions are performed to compensate for system inadequacies. A common example is the manual reformatting of information to match the structure required by the computer. This is a reaction common to workerswhose discretion is limited, such as clerical personnel.

Misuse of the system: The rules are bent to shortcut operational difficulties. This requires significant knowledge of the system and may affect system integrity.

Direct programming: The system is reprogrammed by its user to meet specific needs. This is a typical response of the sophisticated worker.

These physical responses also greatly diminish user efficiency and effectiveness. They force the user to rely upon other information sources, to fail to use a system's complete capabilities, or to perform time-consuming "work-around" actions

IMPORTANT HUMAN CHARACTERISTICS IN DESIGN:

Importance in design is perception, memory, visual acuity, foveal and peripheral vision, sensory storage, information processing, learning, skill, and individual differences.

- Perception
- Proximity
- Similarity
- Matching patterns
- Succinctness
- Closure
- Unity
- Continuity
- Balance
- Expectancies
- Context
- Signals versus noise

Memory: Memory is not the most stable of human attributes, as anyone who has forgotten why they walked into a room, or forgotten a very important birthday, can attest.
-*Short-term*, or working, memory.
Long-term memory
Mighty memory
Sensory Storage

Mental Models: As a result of our experiences and culture, we develop mental models of things and people we interact with.
A mental model is simply an internal representation of a person's current understanding of something. Usually a person cannot describe this mental mode and most often is unaware it even exists.
Mental models are gradually developed in order to understand something, explain things, make decisions, do something, or interact with another person.
Mental models also enable a person to predict the actions necessary to do things if the action has been forgotten or has not yet been encountered.
Movement Control : Once data has been perceived and an appropriate action decided upon, a response must be made.
In many cases the response is a movement. In computer systems, movements include such activities as pressing keyboard keys, moving the screen pointer by pushing a mouse or rotating a trackball, or clicking a mouse button

THE IMPLICATIONS IN SCREEN DESIGN:

Learning: Learning, as has been said, is the process of encoding inlong-term memory information that is contained in short-term memory.

It is a complex process requiring some effort on our part. Our ability to learn is important-it clearly differentiates people from machines.
Given enough time people can improve the performance in almost any task. Too often, however, designers use our learning ability as an excuse to justify complex design.
A design developed to minimize human learning time can greatly accelerate humanperformance.
People prefer to stick with what they know, and they prefer to jump in and get started. Unproductive time spent learning is something frequently avoided.
Skill: The goal of human performance is to perform skillfully. To do so requires linking inputs and responses into a sequence of action.

The essence of skill is performance of actions or movements in the correct time sequence with adequate precision. It is characterized by consistency and economy of effort.

Economy of effort is achieved by establishing a work pace that represents optimum efficiency.
It is accomplished by increasing mastery of the system through such things as progressive learning of shortcuts, increased speed, and easier access to information or data.
Skills are hierarchical in nature, and many basic skills may be integrated to form increasingly complex ones. Lower-order skills tend to become routine and may drop out of consciousness.
System and screen design must permit development of increasingly skillful performance.
Individual Differences: In reality, there is no average user. A complicating but very advantageous human characteristic is that we all differ-in looks, feelings, motor abilities, intellectual abilities, learning abilities and speed, and so on.
In a keyboard data entry task, for example, the best typists will probably be twice as fast as the poorest

and make 10 times fewer errors.
Individual differences complicate design because the design must permit people with widely varying characteristics to satisfactorily and comfortably learn the task or job, or use the Web site.
In the past this has usually resulted in bringing designs down to the level of lowest abilities or selecting people with the minimum skills necessary to perform a job.
But technology now offers the possibility of tailoring jobs to the specific needs of people with varying and changing learning or skill levels. Multiple versions of a system can easily be created.
Design must provide for the needs of all potential users.

HUMAN CONSIDERATIONS IN DESIGN:

The User's Knowledge and Experience
The knowledge possessed by a person, and the experiences undergone, shape the design of the interface in many ways. The following kinds of knowledge and experiences should be identified.
Computer Literacy - Highly technical or experienced, moderate computerexperience, or none
System Experience - High, moderate, or low knowledge of a particular system and its methods of interaction
Application Experience - High, moderate, or low knowledge of similar systems

HUMAN CONSIDERATIONS IN DESIGN:

Task Experience - Other Level of knowledge of job and job tasks
Systems Use - Frequent or infrequent use of other systems in doing job
Education - High school, college, or advanced degree
Reading Level - Less than 5th grade, 5th-12th, more than 12th grade
Typing Skill - Expert (135 WPM), skilled (90 WPM), good (55 WPM), average (40 WPM), or "hunt and peck" (10 WPM).
Native Language or Culture- English, another, or several.

JOB/TASK/NEED

Type of System Use - Mandatory or discretionary use of the system.

Frequency of Use - Continual, frequent, occasional, or once-in-a-lifetime use ofsystem

Task or Need importance - High, moderate, or low importance of the task beingperformed

Task Structure - Repetitiveness or predictability of tasks being automated, high,moderate, or low

Social Interactions - Verbal communication with another person required or notrequired

Primary Training - Extensive or formal training, self training through manuals, orno training

Turnover Rate - High, moderate, or low turnover rate for jobholders

Job Category - Executive, manager, professional, secretary, clerk

Lifestyle - For Web e-commerce systems, includes hobbies, recreational pursuits, and economic status.

PSYCHOLOCICAL CHARCTERISTICS:

Attitude - Positive, neutral, or negative feeling toward job or system
Motivation - Low, moderate, or high due to interest or fear
Patience - Patience or impatience expected in accomplishing goal
Expectations - Kinds and reasonableness
Stress Level - High, some, or no stress generally resulting from task performance
Cognitive Style - Verbal or spatial, analytic or intuitive, concrete or abstract.

PHYSICAL CHARACTRISTICS:

- Age Young middle aged or elderly
- Gender Male or Female
- Handness Left, right or ambidextrous
- Disabilities Blind, defective vision, deafness, motor handicap

HUMAN INTERACTION SPEEDS:

The speed at which people can perform using various communication methods has been studied by a number of researchers.

Reading: The average adult, reading English prose in the United States, has a reading speed in the order of 250-300 words per minute. Proof reading text on paper has been found to occur at about 200 words per minute, on a computer monitor, about 180 words per minute.

One technique that has dramatically increased reading speeds is called Rapid Serial Visual Presentation, or RSVP. In this technique single words are presented one at a time in the center of a screen. New words continually replace old words at a rate set by the reader. For a sample of people whose paper document reading speed was 342 words per minute. (With a speed range of 143 to 540 words per minute.) Single words were presented on a screen in sets at a speed sequentially varying ranging from 600 to 1,600 words per minute. After each set a comprehension test was administered.

READING:

Prose text - 250-300 words per minute. Proof reading text on paper - 200 words per minute. Proof reading text on a monitor - 180 words per minute.

LISTENING:

Speaking to a computer: 150-160 words per minute. After recognition corrections: 105 words per minute.

KEYING:

Typewriter
Fast typist: 150 words per minute and higherAverage typist: 60-7words per minute Computer.
Transcription: 33 words per minuteComposition: 19 words per minute.
Two finger typists
Memorized text: 37 words per minuteCopying text: 27 words per minute.
Hand printing

Memorized text: 31 words per minute.Copying text: 22 words per minute.

UNDERSTAND THE BUSINESS FUNCTION:

The objective of this phase is to establish the need for a system. A requirement is an objective that must be met. A product description is developed and refined, based on input from users, marketing, or other interested parties.

Business definition and requirements analysis

- Direct methods
- Indirect methods
- Requirements collection guidelines

Determining basic business functions

- Developing conceptual modes
- Understanding mental models
- Users new mental model

Design standards or style guides

- Value of standards and guidelines
- Document design
- Design support and implementation

System training and documentation

- Training
- Documentation

DIRECT METHODS:
The significant advantage of the direct methods is the opportunity they provide to hear the user's comments in person and firsthand. Person-to-person encounters permit multiple channels of communication (body language, voice inflections, and so on) and provide the opportunity to immediately follow up on vague or incomplete data. Here are some recommended direct methods for getting input from users**.**

- Individual Face-to-Face Interview
- Telephone Interview or Survey
- Traditional Focus Group
- Facilitated Team Workshop
- Observational Field Study
- User-Interface Prototyping

- Usability Laboratory Testing
- Card Sorting for Web Sites
- A technique to establish groupings of information for Web sites

INDIRECT METHODS:

An indirect method of requirements determination is one that places an intermediary between the developer and the user. This intermediary may be electronic or another person. Using an intermediary can certainly provide useful information. Working through an intermediary, however, takes away the multichannel communication advantages of face-to-face user-developer contact. Some electronic intermediaries do provide some advantages, as will be described shortly. Imposition of a human intermediary can create additional problems. First, there may be a filtering or distortion of the message, either intentional or unintentional. Next, the intermediary may not possess a complete or current understanding of the user’s needs, passing on an incomplete or incorrect message. Finally, the intermediary may be a mechanism that discourages direct user-developer contact for political reasons. Indirect methods include the following:

- MIS Intermediary
- Paper Surveyor Questionnaire
- Electronic Surveyor Questionnaire
- Electronic Focus Group
- Marketing and Sales
- Support Line
- E-Mail or Bulletin Board
- User Group
- Competitor Analyses
- Trade Show
- Other Media Analysis
- System Testing

DETERMINING BASIC BUSINESS FUNCTIONS:

Major system functions are listed and described, including critical system inputs andoutputs.

A flowchart of major functions is developed. The process the developer will use issummarized as follows:

Gain a complete understanding of the user's mental model based upon:

- The user's needs and the user's profile.
- A user task analysis.
- Develop a conceptual model of the system based upon the user's mental model.This includes:
- Defining objects.
- Developing metaphors.

UNDERSTANDING THE USER'S MENTAL MODEL:

The next phase in interface design is to thoroughly describe the expected system user or users and their current tasks.

The former will be derived from the kinds of information collected in Step 1 "Understand the User or Client," and the requirements analysis techniques described above.

A goal of task analysis, and a goal of understanding the user, is to gain a picture of the user's mental model.

A mental model is an internal representation of a person's current conceptualizationand understanding of something.

Mental models are gradually developed in order to understand, explain, and do something.

Mental models enable a person to predict the actions necessary to do things if the actions have been forgotten or have not yet been encountered.

PERFORMING A TASK ANALYSIS:

- User activities are precisely.
- Task analysis involves breaking down the user's activities to the individual tasklevel.
- Knowing why establishes the major work goals;
- Complete description of all user tasks and interactions.
- Work activities are studied using the techniques just reviewed;
- Direct observation, interviews, questionnaires, or obtaining measurements of actual current system usage.
- Listing of the user's current tasks.
- Another result is a list of objects the users see as important to what they do

DEVELOPING CONCEPTUAL MODELS:

- The output of the task analysis is the creation, by the designer, of a conceptualmodel for the user interface.
- A conceptual model is the general conceptual framework through which thesystem's functions are presented.
- Such a model describes how the interface will present objects, the relationshipsbetween objects, the properties of objects, and the actions that will be performed.
- A conceptual model is based on the user's mental model. Since the term mental model refers to a person's current level of knowledge about something, people will always have them.

DEVELOPING CONCEPTUAL MODELS:

Since mental models are influenced by a person's experiences, and people have different experiences, no two user mental models are likely to be exactly the same. Each person looks at the interface from a slightly different perspective. The goal of the designer is to facilitate for the user the development of useful mental model of the system.

This is accomplished by presenting to the user a meaningful conceptual model of the system .
When the user then encounters the system, his or her existing mental model will, hopefully, mesh well with the system's conceptual model. As a person works with a system, he or she then develops a mental model of the system. The system mental model the user derives is based upon system's behavior, including factors such as the system inputs, actions, outputs (including screens and messages), and its feedback and guidance characteristics, all of which are components of the conceptual model.
Documentation and training also playa formative role. Mental models will bedeveloped regardless of the particular design of a system, and then they will be Modified with experience.
What must be avoided in design is creating for the user a conceptual model that leads to the creation of a false mental model of the system, or that inhibits the user from creating a meaningful or efficient mental model.

Guidelines for Designing Conceptual Models:

- Reflect the user's mental model, not the designer's.
- Draw physical analogies or present metaphors.
- Comply with expectancies, habits, routines, and stereotypes.
- Provide action-response compatibility.
- Make invisible parts and process of a system visible.
- Provide proper and correct feedback.
- Avoid anything unnecessary or irrelevant.
- Provide design consistency.
- Provide documentation and a help system that will reinforce the conceptual model.
- Promote the development of both novice and expert mental models.

Defining Objects:

Determine all objects that have to be manipulated to get work done.
Describe:

- The objects used in tasks.
- Object behavior and characteristics that differentiate each kind of object.
- The relationship of objects to each other and the people using them.
- The actions performed.
- The objects to which actions apply.
- State information or attributes that each object in the task must preserve,display,or allow to be edited.
- Identify the objects and actions that appear most often in the workflow.

Make the several most important objects very obvious and easy to manipulate

Developing Metaphors:

- Choose the analogy that works best for each object and its actions.
- Use real-world metaphors.
- Use simple metaphors.
- Use common metaphors.
- Multiple metaphors may coexist.
- Use major metaphors, even if you can't exactly replicate them visually.
- Test the selected metaphors.

SCREEN DESIGNING:

Use of a screen or Web page, and a system, is affected by many factors. These include how much information is presented, how the information is organized, what language is used to communicate to the user, how distinctly the components are displayed, what aesthetics are used, and how consistent a screen or page is with other screens or pages. First, let's look at what aspects of poor design can be distracting to the user, what a user is looking for in good design, and the kinds of things screen users do interacting with a system or Web site. Then, we'll address the principles of good design

How to distract the screen user

- Unclear captions
- Improper type and graphic emphasis
- Misleading headings
- Irrelevant and unnecessary headings
- Inefficient results
- Clustered and cramped layout
- Poor quality of presentation
- Legibility
- Appearance
- arrangement
- Visual inconsistency
- Lack of design features
- Over use of 3D presentations
- Overuse of too many bright colors
- Bad typography

Variety of distractions

- Numerous audio and visual interruptions
- Extensive visual clutter
- Poor information readability
- In comprehensible screen components

- Confusing and inefficient navigation
- Inefficient operations
- Excessive or inefficient page scrolling
- Information overload
- Design in consistency
- Outdated information

What screen users want

- an orderly clean clutter free appearance
- An obvious indication of what is being shown and what should be done with it.
- Expected information located where it should be.
- A clear indication of what relates to what.
- Plain and simple english
- A clear indication of when an action can make a permanent change in data

What screen users do

- Identifies a task to be performed or need to be fulfilled.
- Decides how the task will be completed or need fulfilled.
- Manipulates the computers controls.
- Gathers necessary data.

Design goals:

- Reduce visual work
- Reduce intellectual work
- Reduce memory work
- Reduce mentor work
- Eliminate burdens or instructions.

SCREEN MEANING AND PURPOSE:

Each screen element:

- Every control
- All text
- Screen organization
- All emphasis
- Each color
- Every graphic
- All screen animation
- All forms of feedback

Must:

- Have meaning to screen users
- Serve a purpose in performing task organizing screen elements

Consistency

- Provide real world consistency
- Provide internal consistency
- Operational and navigational procedures
- Visual identity or theme
- Component
- Organization
- Presentation
- Usage
- Locations
- Follow the same conventions
- Deviate only when there is clear benefit to user

ORDERING OF SCREEN DATA & CONTENT:

- Divide information into units that are logical, meaningful and sensible.
- Organize by interrelationships between data or information.
- Provide an ordering of screen units of elements depending on priority.
- Possible ordering schemes include
- Conventional
- Sequence of use
- Frequency of use
- Function
- Importance
- General to specific.
- Form groups that cover all possibilities.
- Ensure that information is visible.
- Ensure that only information relative to task is presented on screen.
- Organizational scheme is to minimize number of information variables.
- Upper left starting point
- Provide an obvious starting point in the screen's upper left Corner.

SCREEN NAVIGATION AND FLOW

Provide an ordering of screen information and elements that:

- ➢ Is rhythmic guiding a person's eye through display
- ➢ Encourages natural movement sequences.
- ➢ Minimizes pointer and eye movement distances.

Locate the most important and most frequently used elements or controls at top left.
Maintain top to bottom , left to right flow.

Assist in navigation through a screen by

- Aligning elements
- Grouping elements
- Use of line borders

Through focus and emphasis, sequentially, direct attention to items that are

- Critical
- Important
- Secondary
- Peripheral

Tab through window in logical order of displayed information.

- Locate command button at the end of the tabbing order sequence,
- When groups of related information must be broken and displayed on separate screens, provide breaks at logical or natural points in the information flow.

In establishing eye movement through a screen, also consider that the eye trends tomove sequentially , for example –

- From dark areas to light areas
- From big objects to little objects
- From unusual shapes to common shapes.
- From highly saturated colors to unsaturated colors.

These techniques can be initially used o focus a person's attention.
Maintain top to bottom, left to right through the screen. This top to bottom orientation is Recommended for information entry for the following reasons –

- Eye movements between items will be shorter.
- Control movements between items will be shorter.
- Groupings are more obvious perceptually.
- When one's eyes moves away from the screen and then back, it returns to about same place it left, even if it is seeking next item in sequence.
-

Most product style guides recommend a left to right orientation.
Our earliest display screens reflected this left to right entry orientation.

Top to bottom orientation is also recommended for presenting displays of read onlyinformation that must be scanned.

VISUALLY PLEASING COMPOSITION:

Eyeball fixation studies also indicate that during the initial scanning of a display in a clockwise direction, people are influenced by the symmetrical balance and weight of the titles, graphics, and text of the display. The human perceptual mechanism seeks order and meaning, trying to impose structure when confronted with uncertainty. Whether a screen has meaningful and evident form or is cluttered and unclear is immediately discerned. A cluttered or unclear screen requires that some effort be expended in learning and understanding what is presented. The screen user who must deal with the display is forced to spend time to learn and understand. The user who has an option concerning whether the screen will or will not be used may reject it at this point if the perceived effort in understanding the screen is greater than the perceived gain in using it.

• Provide visually pleasing composition with the following qualities –

- balance
- Symmetry
- Regularity
- Predictability
- Sequentiality
- Economy
- Unity
- Proportion
- Simplicity
- Groupings.

Balance:

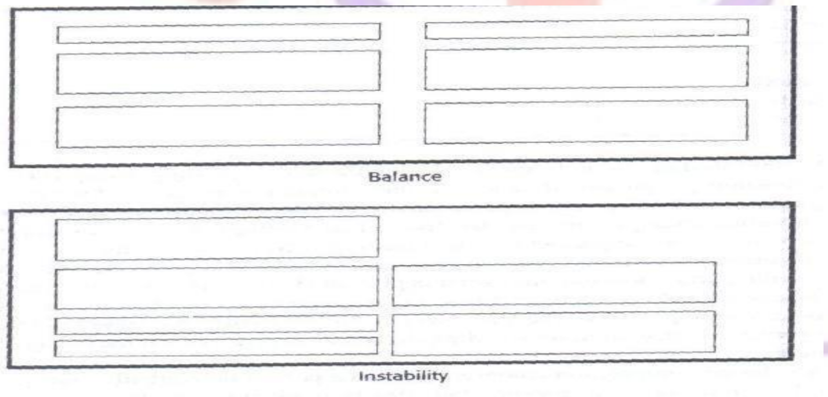


Figure 3.1 Balance (versus instability).

Symmetry:

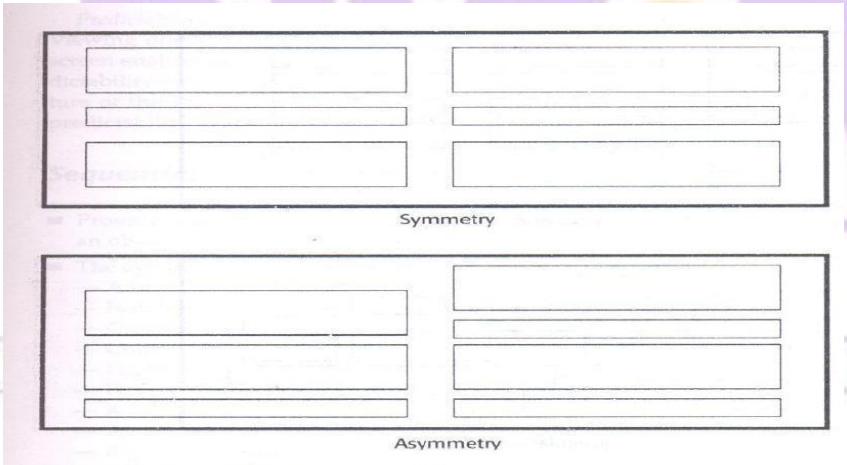


Figure 3.2 Symmetry (versus asymmetry).

Regularity:

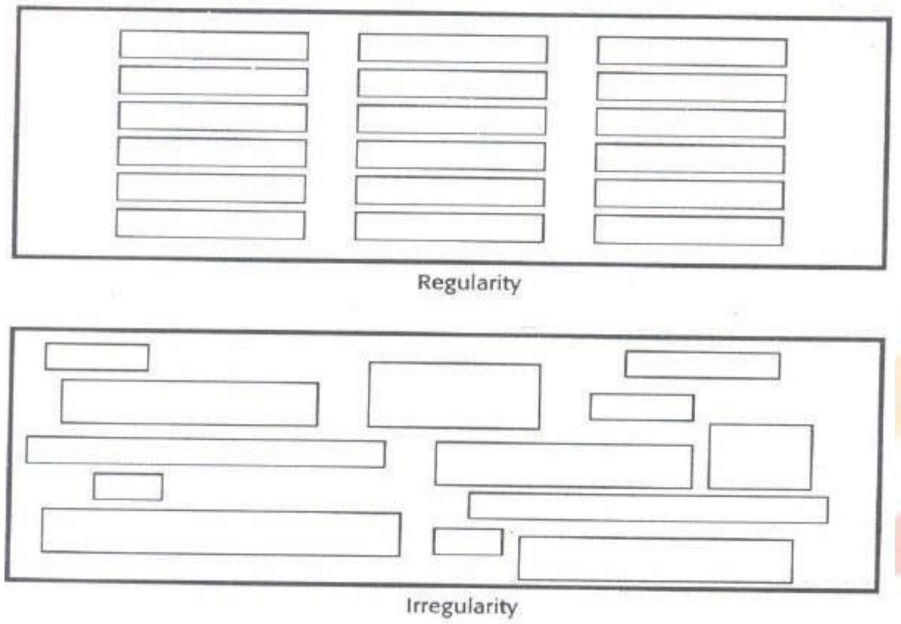


Figure 3.3 Regularity (versus irregularity).

Regularity, illustrated in Figure 3.3, is a uniformity of elements based on some principle or plan. Regularity in screen design is achieved by establishing standard and consistently spaced column and row starting points for screen elements. It is also achieved by using elements similar in size, shape, color, and spacing. The opposite of regularity, irregularity, exists when no such plan or principle is apparent. A critical element on a screen will stand out better, however, if it is not regularized.

Predictability:

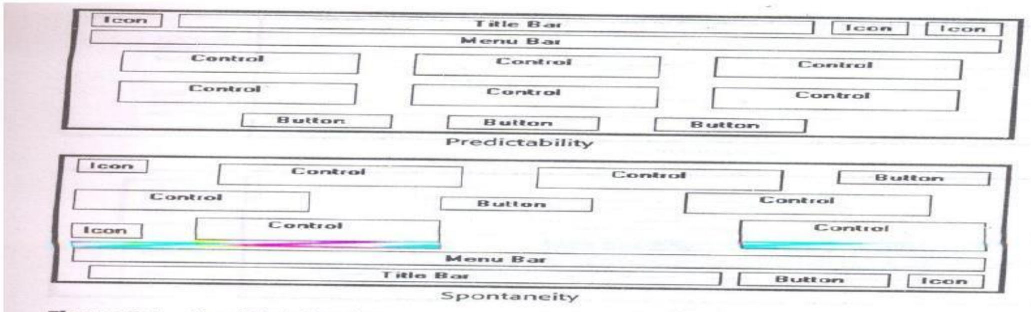


Figure 3.4 Predictability (versus spontaneity).

Predictability, illustrated in Figure 3.4, suggests a highly conventional order or plan.
Viewing one screen enables one to predict how another will look. Viewing part of a screen enables one to predict how the rest of the screen will look. The opposite of predictability — spontaneity — suggests no plan and thus an inability to predict the structure of the remainder of a screen or the structure of other screens. In screen design predictability is also enhanced through design consistency.

Sequentially:

- The eye trends to be attracted to :
- A brighter element before one less bright
- Isolated elements before elements in a group
- Graphics before text
- Color before black and white
- Highly saturated colors before those less saturated.
- Dark areas before light areas
- A big element before a small one
- An unusual shape before a usual one
- Big objects before little objects

Sequentiality, illustrated in Figure 3.5, is a plan of presentation to guide the eye through the screen in a logical, rhythmic order, with the most important information significantly placed. Sequentiality can be achieved by alignment, spacing, and grouping as illustrated.
The opposite of sequentiality is randomness, whereby an arrangement and flow cannot be detected. The eye tends to move first to the elements listed above, and then from one to the other. For example, it moves from highly saturated colors to unsaturated colors, from dark to light areas, from big to little objects, and from unusual to usual shapes.

Visually Pleasing Composition is a crucial aspect of Human-Computer Interaction (HCI) design. Here are some key principles and guidelines to create visually pleasing compositions in HCI:

Principles of Visually Pleasing Composition:

1. Balance: Balance refers to the arrangement of visual elements to create a sense of stability and harmony.
2. Proportion: Proportion refers to the relationship between the size of different elements in a composition.
3. Emphasis: Emphasis refers to the focal point in a composition that draws the user's attention.
4. Movement: Movement refers to the way the user's eye moves through a composition.
5. Pattern: Pattern refers to the repetition of similar elements in a composition.
6. Unity: Unity refers to the sense of oneness and coherence in a composition.
7. Variety: Variety refers to the use of different elements, such as color, texture, and shape, to create visual interest.

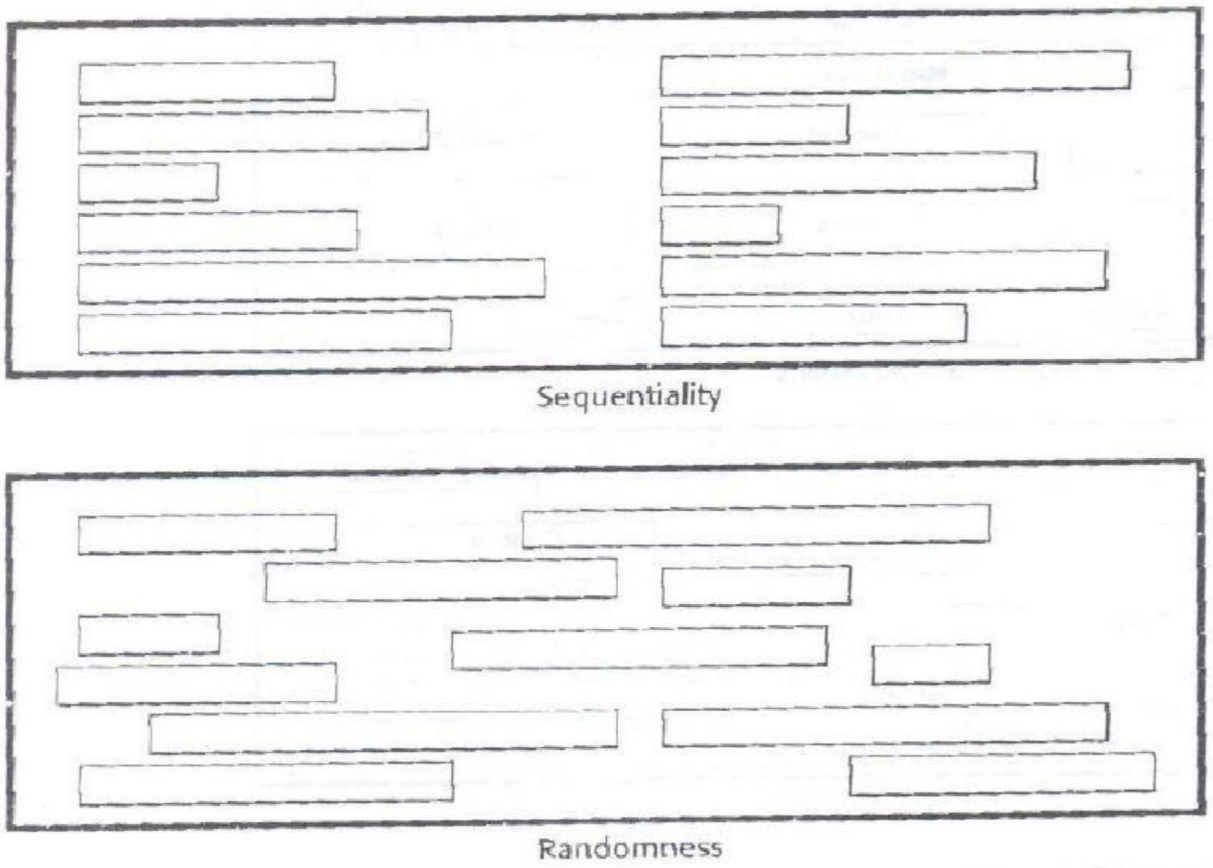


Figure 3.5 Sequentiality (versus randomness).

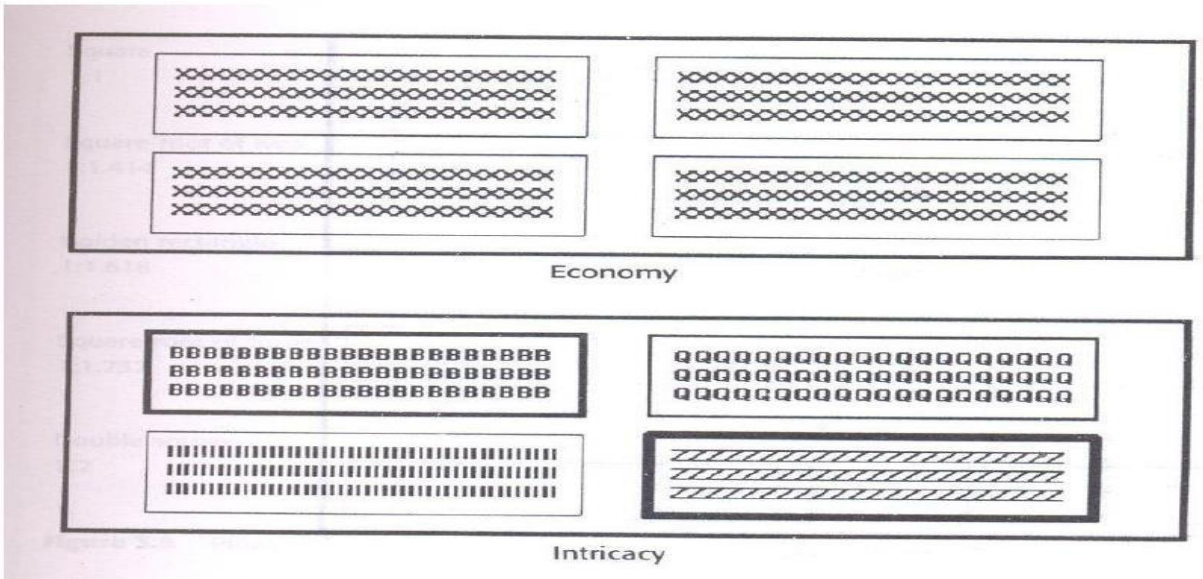


Figure 3.6 Economy (versus intricacy).

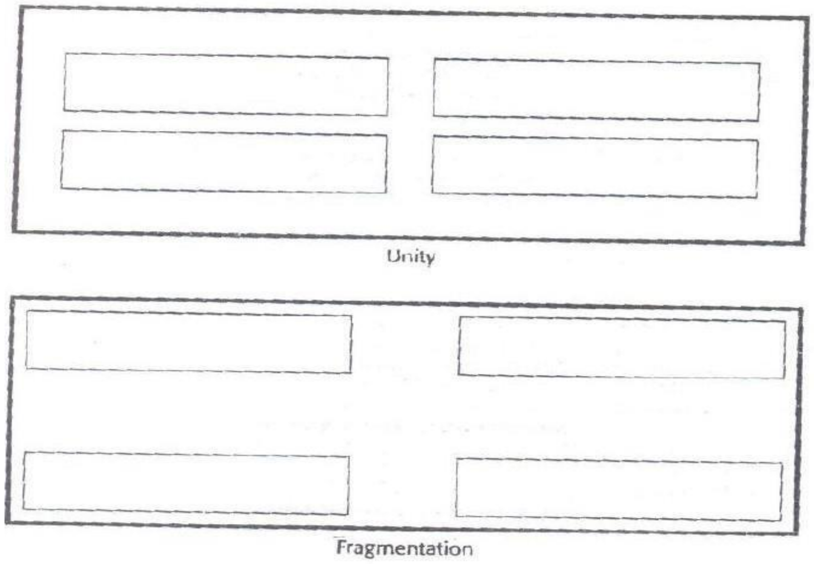


Figure 3.7 Unity (versus fragmentation).

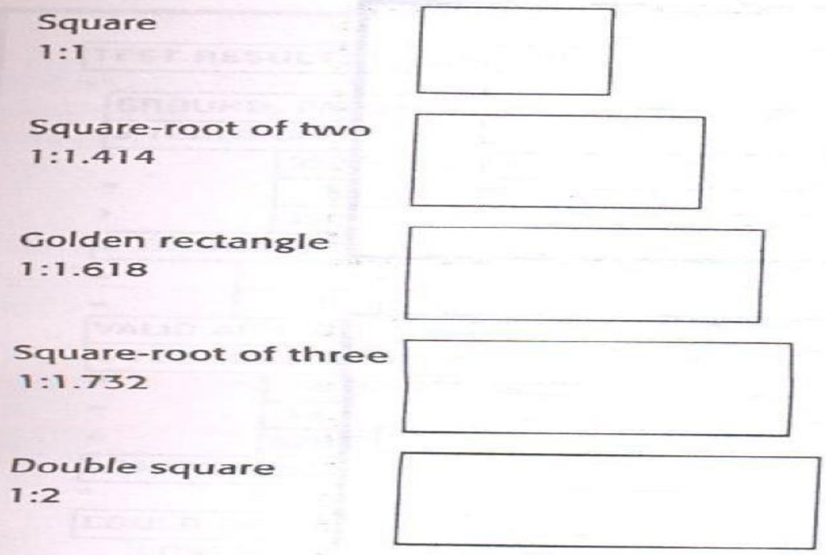


Figure 3.8 Pleasing proportions.

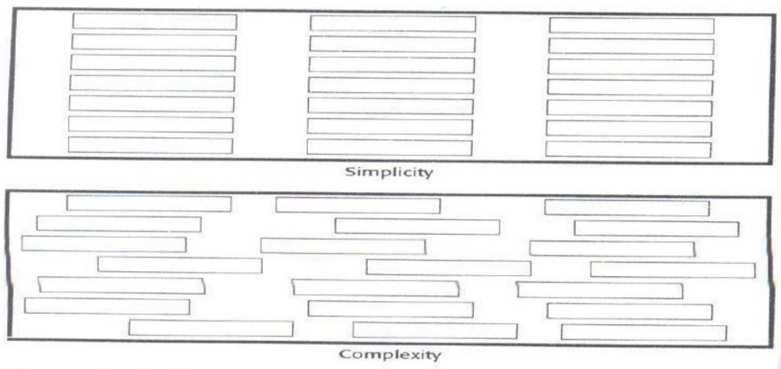
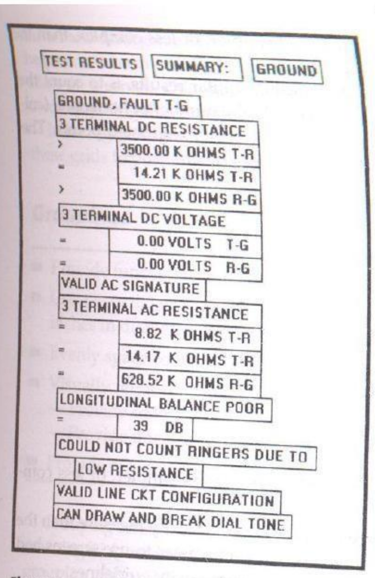


Figure 3.9 Simplicity (versus complexity).



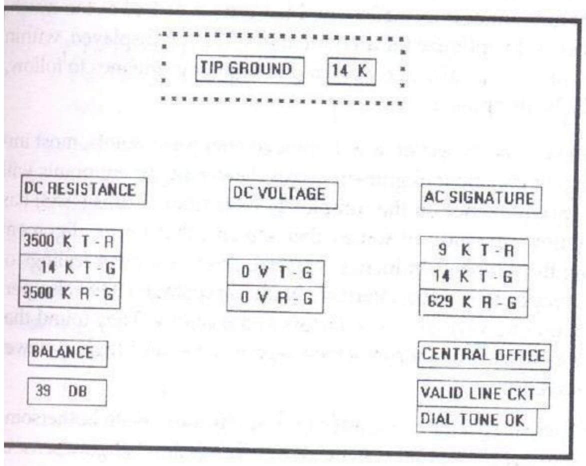
TEST RESULTS SUMMARY: GROUND
GROUND, FAULT T-G
3 TERMINAL DC RESISTANCE
> 3500.00 K OHMS T-R
= 14.21 K OHMS T-R
> 3500.00 K OHMS R-G
3 TERMINAL DC VOLTAGE
= 0.00 VOLTS T-G
= 0.00 VOLTS R-G
VALID AC SIGNATURE
3 TERMINAL AC RESISTANCE
= 8.82 K OHMS T-R
= 14.17 K OHMS T-R
= 628.52 K OHMS R-G
LONGITUDINAL BALANCE POOR
= 39 DB
COULD NOT COUNT RINGERS DUE TO
LOW RESISTANCE
VALID LINE CKT CONFIGURATION
CAN DRAW AND BREAK DIAL TONE

Figure 3.10 Original screen, from Tullis (1981), with title, captions, and data inscribed by rectangles.

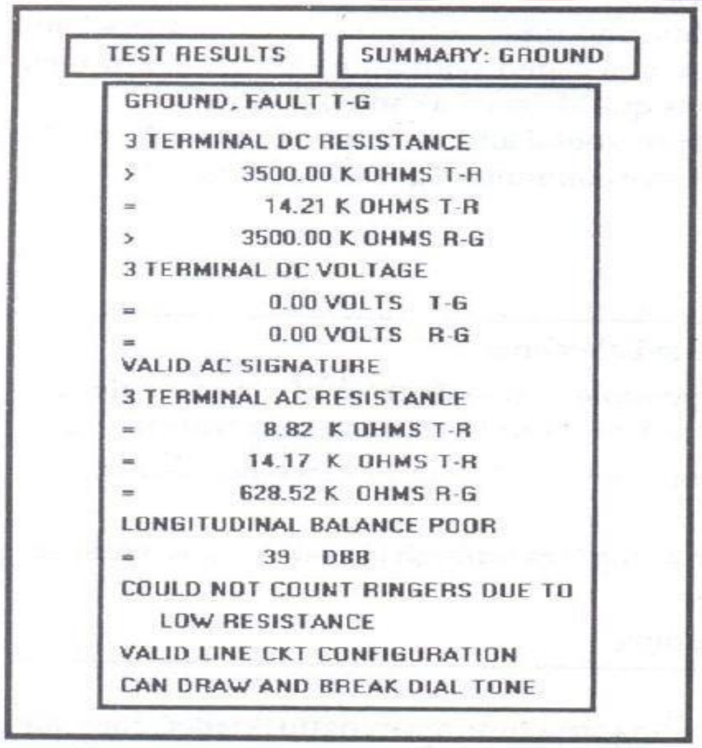
1. Draw a rectangle around each element on a screen, including captions, controls, headings, data, title, and so on.
2. Count the number of elements and horizontal alignment points (the number of columns in which a field, inscribed by a rectangle, starts).
3. Count the number of elements and vertical alignment points (the number of rows in which an element, inscribed by a rectangle, starts). This has been done for the text-based screens illustrated in Figures 3.10 and 3.11. These screens are examples from the earlier study by Tullis (1981) described in the introduction. They are an original read-only inquiry screen (Figure 3.10) from the screens whose mean search time was 8.3 seconds, and a redesigned screen (Figure 3.11) from the screens whose mean search time was 5.0 seconds. A complexity calculation using information theory for each screen is as follows:

Optimize the number of elements on a screen, within limits of clarity. Minimize the alignment points, especially horizontal or columnar.

- Figure 3.11 (redesigned):
  18 fields with 7 horizontal (column) alignment points = 43 bits.
  18 fields with 8 vertical (row) alignment points = 53 bits.
  Overall complexity = 96 bits.



gure 3.11 Redesigned screen, from Tullis (1981), with title, captions, and data inscribed by rectangles.



```
TEST RESULTS          SUMMARY: GROUND
GROUND, FAULT T-G
3 TERMINAL DC RESISTANCE
>      3500.00 K OHMS T-R
=        14.21 K OHMS T-R
>      3500.00 K OHMS R-G
3 TERMINAL DC VOLTAGE
=         0.00 VOLTS  T-G
=         0.00 VOLTS  R-G
VALID AC SIGNATURE
3 TERMINAL AC RESISTANCE
=         8.82  K OHMS T-R
=        14.17  K  OHMS T-R
=       628.52 K  OHMS R-G
LONGITUDINAL BALANCE POOR
=        39   DBB
COULD NOT COUNT RINGERS DUE TO
   LOW RESISTANCE
VALID LINE CKT CONFIGURATION
CAN DRAW AND BREAK DIAL TONE
```

Figure 3.12 Original screen, from Tullis (1981), with grouping indicated by bold boxes.

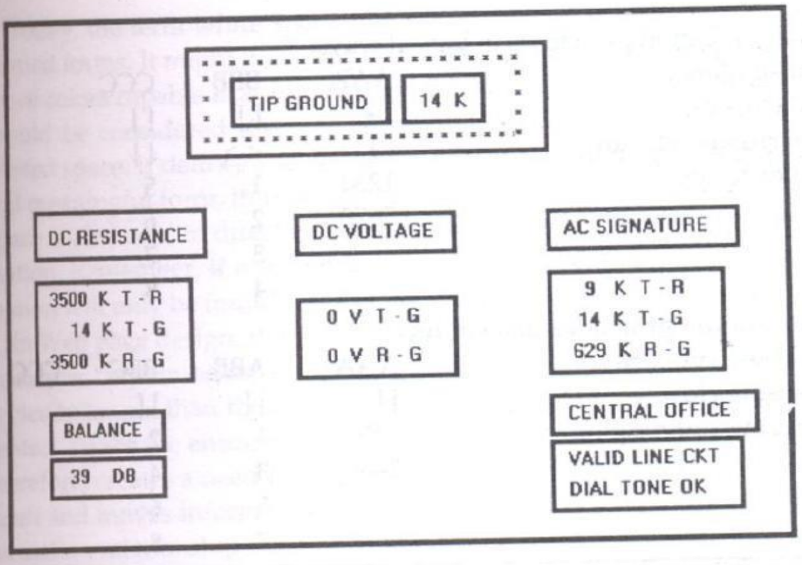


Figure 3.13 Redesigned screen, from Tullis (1981), with grouping indicated by bold boxes.

GROUPING USING BORDERS:

Grouping screen elements aids in establishing structure, meaningful relationships, and meaningful form. In addition to providing aesthetic appeal, past research has found that grouping aids in information recall and results in a faster screen search. The study by Grose et al. (1998) found that providing groupings of screen elements containing meaningful group titles was also related to shorter screen search times. In this study groupings also contributed to stronger viewer preferences for a screen.

- Provide functional groupings
- Create spatial groupings
- Provide meaningful titles for each grouping
- Incorporate line borders
- Do not exceed three line thick ness
- Create lines consistent in height and length
- For adjacent groupings with borders where ever possible
- Use rules and borders sparingly

FOCUS AND EMPHASIS

- Visually emphasize the
- Most prominent element
- Most important elements
- Central idea or focal point
- De emphasize less important elements
- To ensure that
- Too many screen elements are emphasized.
- Screen clutter
- Using too many emphasize techniques

- To provide emphasis use techniques such as :
- Higher brightness
- Reverse polarity
- Larger and distinctive font
- Underlining
- Blinking
- Line rulings
- Contrasting colors
- Larger size
- Positioning
- Isolation
- Distinctiveness
- White space

INFORMATION RETRIEVAL ON WEB:

- The most sought after web commodity is content.
- Behavior is often goal driven.
- Reading is no longer a linear activity.
- Impatience.
- Frequent switching of purpose.
- Web users access site for different reasons: a focused search for a piece ofinformation or an answer less focused for browsing or surf.
- High tech capabilities, fancy graphics do not compensable for inefficient or poorcontent.
- Initial focus on attention
- Page perusal
- Scanning guidelines
- Browsing
- Browsing guidelines
- Searching
- Problems with search facilities
- Search facility guidelines
- Express the search
- Progressive search refinement
- Launch the search
- Present meaningful results

SCANNING GUIDELINES:

To aid in finding information, a Web page must be structured to facilitate scanning. Studies report that about 80 percent of viewers scan a new page when it first appears. When they find something interesting, then they read. Only 16 percent read word-byword. People also spend about 12 percent of their time trying to find desired information.
People that scan generally read only headings, not full text prose. Useful information buried in dense text is often missed (Koyani et al., 2004). Difficult-to-scan pages may cause a viewer to give up and move on,

preferring to spend what is often limited time elsewhere. Organization. Pages should be organized to guide a person's eye through the page in a systematic way. Efficient eye movement, and scanning direction, can be made obvious through the alignment and columnization of page elements, and effective use of white space. Order information page information to be consistent with the viewers expected scanning path. Highlight. Emphasize important information, issues, and topics through use of varying font sizes and boldness. Links can serve as emphasis entities since the underlining makes them stand out. Headings and subheadings. Distinctive and obvious headings are often targets for page scanning. Well-written headings can be important cues in helping people classify and organize page information, understand page structure, and maintain orientation. Headings should be written clearly, conceptually relating to the information or functions that follow.

Organization

- Minimize eye movement
- Provide groupings of information
- Organize content in a logical and obvious way.

Writing:

- Provide meaningful headings and subheadings.
- Provide meaningful titles
- Concisely write the text.
  - Use bullets/ numbers
  - Array information in tables
  - Presentation
    – Key information in words or phrases
    – Important concepts

BROWSING GUIDELINES:

A person, in looking for a particular item of information on the Web, can implement "find" strategies commonly called browse and search. As mentioned previously, browsing is non-specific surfing. People wander around a Web site, or Web sites, at their own pace, following discovered links, scanning headings, and using other presented page cues to try and locate what they are interested in. Search is a much more structured find mechanism. In search, one or more keywords are entered into a search field,
sometimes with restrictive parameters, and then the user is presented with a collection of descriptions and links that might possibly contain the information of interest.

- Facilitate scanning
- Provide multiple layers of structure
- Make navigation easy
- Respect users desire to leave
- Upon returning help users reorient themselves.
- Users can browse deeply or simply move on.
- Provide guidance to help reorientation
- Understand terms to minimize to need for users to switch context.

PROBLEMS WITH SEARCHING:

- Not understanding the user.
- Difficulties in formulating the search.

- Difficulties in presenting meaningful results.
- Identify the level of expertise of user.

KNOW THE SEARCH USER:

- Plan for user's switchig purposes during search process.
- Plan for flexibility in the search process.
- Anticipate
- Nature of every possible query
- Kind of information desired
- How much information will result the search.

STATISTICAL GRAPHICS:

- A statistical graphic is data presented in a graphical format.
- A well designed statistical graphic also refered to as chart or graph.
- Use of statistical graphics
    - reserve for material that is rich, complex or difficult.
- Data Presentation
- emphasize the data
- Minimize non data elements
- Minimize redundant data
- Fill the graph's available area with data.
- Show data variation
- Provide proper context for data interpretation.

Scales and shading

- place ticks to marks scales on the outside edge of each axis.
- employ a linear scale.
- mark scales at standard or customary intervals
- Start a numeric scale at zero.
- display only a single scale on axis.
- provide aids for scale interpretation.
- clearly label each axis.
- Provide scaling consistency
- consider duplicate axis for large scale data.
- Proportion
- Lines
- Labeling
- Title
- Interpretation of numbers

TYPES OF STATISTICAL GRAPHS:

Statistical graphics take many forms. There are curve and line graphs, surface charts, scatterplots, bar charts, histograms, segmented or stacked bars, and pie charts

✓ curve and line graphs

- ✓ Single graph
- ✓ Four or five maximum
- ✓ Label identification
- ✓ Legend
- ✓ Tightly packed curves
- ✓ Important or critical data
- ✓ Comparing actual and projected data
- ✓ Data differences
- ✓ Surface charts
- ✓ Ordering
- ✓ Coding schemes
- ✓ Labels
- ✓ Scatter plots
- ✓ two dimensions
- ✓ Consistent intervals
- ✓ multiple data sets
- ✓ Significant points
- ✓ Bar graphs
- ✓ consistent orientation
- ✓ Meaningful organization
- ✓ Bar spacing
- ✓ Differentiation
- ✓ Important or critical data
- ✓ Related bar ordering
- ✓ Reference index
- ✓ labeling
- ✓ Segmented or stacked bars.
- ✓ Data category ordering
- ✓ Large segments
- ✓ Coding schemes
- ✓ labeling
- ✓ Flow charts
- ✓ Order of stps
- ✓ Orientation
- ✓ Coding conventions
- ✓ Arrows
- ✓ Highlighting
- ✓ One decission at each step
- ✓ Consistently order and word all choices
- ✓ Pie chart

TECHNOLOGICAL CONSIDERATION -INTERFACE DESIGN:

Interface design is also affected, and constrained by, characteristics of the hardware being used and the interface's controlling software.

Graphical systems:

Graphical system design must be compatible with the system's power, screen size, screen resolution, and displayable colors, fonts and other features. Designs for Web systems must also take into consideration the characteristics of the browsers being used and the bandwidth of the communication medium.

Screen design must be compatible with the capabilities of the system:

- system power
- Screen size
- Screen resolution
- Display colors
- Other display features

Screen design must be compatible with the capabilities of the

- Platform compatibility
- development and implementation
- Platform style guide

UNIT-III

A window is an area of the screen that contains a particular view of some area of the computer or some portion of a person's dialog with the computer.

Navigation Goals:

- A well designed navigation system facilitates quick & easy navigation between components whose structure & relationship are easily comprehensible.
- For the user, answers to the following questions must be obvious at all times during an interaction: Where am I now? Where did I come from? Where can I go from here? How can I get there quickly?
- General system navigation guidelines include the following.
- Control

For multilevel menus, provide one simple action to:
Return to the next higher-level menu.
Return to the main menu.
Provide multiple pathways through a menu hierarchy whenever possible. Menu Navigation Aids
To aid menu navigation & learning, provide an easily accessible: o Menu map or overview of the menu hierarchy.
A "look ahead" at the next level of choices, alternatives that will be presented when a currently viewed choice is selected.
Navigation history.
Web Site Navigation: In designing a Web Site Navigation scheme there are two things to take in consideration: Never assume that users know as much about a site as the site designers do.
Any page can be an entry point into the website. Web site navigational design includes:
Web site organization Divide content into logical fragments, units or chunks.
Establish a hierarchy of generality or importance:
Components of a Web Navigation System to move between Web site information fragments necessitates the creation of navigation links.
General link guidelines are:
-Sensible
-Available
-Obvious & Distinctive
-Consistent
-Textual
-Provide multiple navigation paths
Browser Command Buttons Hide the split between the browser & the Web site application by including navigational controls within the application.

- Web Site Navigation Bars
- Provide a global navigation bar at the top of each page.
- Provide a local category or typical links navigation bar on the left side of a page.
- Textual Phrases
- Provide a mix of textual phrase links: -In explicit menus. -Embedded within page text.
- Graphical Images or Icons

- Graphical images or icons may appear in an array in the form of a navigation bar or be individually
- located at relevant points in a page.
- Command Buttons
- Command buttons may appear in an array in the form of a navigation bar or be individually located at
- relevant points in a page.

Selection of Window:

Window Characteristics

- A name or title, allowing it to be identified.
- A size in height & width (which can vary).
- A state, accessible or active or not accessible.
- Visibility–the portion can be seen.
- A location relative to the display boundary.
- Presentation–its arrangement with respect to other windows.
- Management capabilities.
- Highlighting

The function, task or application to which it is dedicated.

1. Attraction of Windows
   - Presentation of Different Levels of Information.
   - Presentation of Multiple Kinds of Information.
   - Sequential Presentation of Levels or Kinds of Information.
   - Access to Different Sources of Information.
   - Combining Multiple Sources of Information.
   - Performing More Than One Task.
   - Reminding.
   - Monitoring.
   - Multiple Representations of the Same Task.
2. Constraints in Window System Design
   - Historical Considerations
   - Hardware Limitations
   - Human Limitations
3. Window Management
   - Single-Document Interface
   - It's a single primary window with a set of secondary windows.
4. Multiple-Document Interface

It's a technique for managing a set of windows where documents are opened into windows.

Contains:

-A single primary window called the parent.

-A set of related document or child windows, each also essentially a primary window.

4. Organizing Window Functions
   - Window Organization–organize windows to support user tasks.
   - Number of Windows–minimize the number of windows needed to accomplish an objective.

5. Window Operations

I. Active window
- A window should be made active with as few steps as possible.
- Visually differentiate the active window from other windows.

ii. Opening a window
- Provide an iconic representation or textual list of available windows.
- If more than one object is selected & opened, display each object in a separate window. Designate the last window selected as the active window.

iii. Sizing windows
- Provide large-enough windows to present all relevant & expected information for the task.

iv. Window placement
- Position the window so it is entirely visible.

v. Window separation
- Crisply, clearly & pleasingly demarcate a window from the background of the screen on which it appears.

vi. Moving a window
- Permit the user to change the position of all windows.

Vii .Resizing a window
- Permit the user to change the size of primary windows

Select the Proper Device-Based Controls
Device-based controls, often called input devices, are the mechanisms through which people communicate their desires to the system.
Identify the characteristics and capabilities of device-based control
• Trackball
• Joystick
• Graphic tablet
• Light pen
• Touch screen
• Voice
• Mouse
• Keyboard
Trackball
• Description
– A ball that rotates freely in all directions in its socket
• Advantages
– Direct relationship between hand and pointer movement in terms of direction and speed
– Does not obscure vision onscreen
– Does not require additional desk space (if mounted on keyboard)
• Disadvantage
– Movement indirect, in plane different from screen
– Requires hand to be removed from keyboard keys
– Requires different hand movements
– May be difficult to control
– May be fatiguing to use over extended time

Joystick
• Advantages
– Direct relationship between hand and pointer movement in terms of direction and speed
– Does not obscure vision on screen
– Does not require additional desk space (if mounted on keyboard)
• Disadvantage
– Movement indirect, in plane different from screen
– Requires hand to be removed from keyboard keys
– Requires different hand movements
– May be difficult to control
– May be fatiguing to use over extended time
– May be slow and inaccurate.
Graphic (Touch) Tablet
• Description
– Pressure-, heat-,light-, or light-blockage-sensitive horizontal surfaces that lie on the desktop or keyboard
– May be operated with fingers, light pen, or objects like pencil
• Advantages
– Direct relationship between hand and pointer movement in terms of direction and speed
– Does not obscure vision of screen
– More comfortable horizontal operating plane
• Disadvantage
Movement is indirect, in a plane different from screen
– Requires hand to be removed from keyboard
– Requires different hand movements to use
– Finger may be too large for accuracy with small objects
Touch Screen
• Advantages
– Direct relationship between hand and pointer movement in terms of direction and speed
– Movement is direct, in the same plane as screen
– Requires no additional desk space
• Disadvantage
– Finger may obscure part of screen
– Finger may be too large for accuracy with small objects
– Requires moving the hand far from the keyboard to use
– Very fatiguing to use for extended period of time
– May Damage the screen
Light Pen
• Description
– A special surface on a screen sensitive to the touch of a special stylus pen
• Advantage
– Direct relationship between hand and pointer movement in terms of direction, distance, and speed
– Movement is direct, in the same plane as screen
– Requires minimal additional desk space
– Stands up well in high-use environments

– More accurate than finger touching
• Disadvantage
– Hand may obscure part of screen
– Requires picking it to use
– Requires moving the hand far from the keyboard to use
– Very fatiguing to use for extended period over time
Messages
• Screen messages is classified into two categories
– System messages:
• Generated by the system to keep the user informed of the system's state and activities
– Instructional messages (prompting message):
• tell the user how to work with, or complete the screen displayed System Messages
• Status messages
– Providing information concerning the progress of a lengthy operation
– Usually contains a progress indicator and a short message
• Informational messages (notification messages)
– This kind of message is usually identified by an "I" icon to the left of the message
• Warning messages
– They are usually identified by an"!"
– The user must determine whether the situation is in fact a problem and may be asked to advise the system whether or not to proceed (A deletion request by a user is any action that commonly generates a warning messages) System Messages
• Critical messages (Action messages)
– Call attention to conditions that require a user action before the system can proceed
– Some products use a "Do Not" symbol while others use a "Stop" sign. An X in a circle used by Microsoft Windows
• Question messages
– A question message asks a question and offers a choice of options for selection
– It is designated by a "?" icon proceeding the message text
Message Box Controls • Command Buttons: – If a message requires no choices to be made, include an OK button – If a message requires a choice to be made • OK and Cancel buttons only when the user has the options continue or cancel • Yes and No buttons when the user must decide how to continue • If these choices are too ambiguous, label with the name of specific actions – If a message describes an interrupted process, provide Stop button – If a message offer a chance to cancel a process, provide a Cancel button – If more details about a message must be presented, provide a Help button – Display only one message box for a specific condition • Close Box: – Enable the title bar Close only if the message includes a Cancel button Instructional Messages • Provide instructional information at the depth of detail needed by the user – Accessing instruction through a Help function is the best solution • Location it at strategic position on the screen
Create Meaningful Graphics, Icons and Images
Creating Images
 Create familiar and concrete shapes
 Create visually and conceptually distinct shapes
 Incorporate unique features of an object
 Do not display within a border

 Clearly reflect object represented
 Simple reflect object represented, avoiding excessive detail
 Create as a set, communicating relationships to one another through common shapes
 Provide consistency in icon type
 Create shapes of the proper emotional tone
Creating Images
• Create familiar and concrete shapes
• Create visually and conceptually distinct shapes
• Incorporate unique features of an object
• Do not display within a border
• Clearly reflect object represented
• Simple reflect object represented, avoiding excessive detail
• Create as a set, communicating relationships to one another through common shapes
• Provide consistency in icon type
• Create shapes of the proper emotional tone
Icons:
• Icons are most often used to represent objects and actions with which users can interact
• Icons may stand alone on a desktop or in a window, or be grouped together in a toolbar
• A secondary use of a icon is to reinforce important information, a warning icon in a dialog message box

Characteristics of Icons
• Syntactic refers to a icon's physical structure
– Shape, Color, Size
– Similar shapes and colors can be used to classify a group of related icons
• Semantics is the icon's meaning
– What does it refer – a file, a waste basket, or some other objects?
• Pragmatics is how the icons are physically produced and depicted
– Is the screen resolution enough to illustrate?
• Syntactic, semantics and pragmatics determine an icon's effectiveness and usability
Influences on Icon Usability
• Provide icons that are
– Familiar
– Clarity
– Simple
– Consistent
– Directness of the meaning
– Efficient
– Discriminate able from others
• Also consider the
– Context in which the icon issued
– Expectancies of users
– Complexity of task
Choosing Icons
• A Successful Icon

– Looks different from all other icons

Window Navigation Schemes:

Windows operating systems typically provide a graphical user interface (GUI) that includes windows for different applications. Users navigate through these windows using a mouse or keyboard.

Windows have various controls such as buttons, menus, and tabs, allowing users to interact with the content and features of applications.

Selection of Devices:

When it comes to selecting devices, Windows supports a wide range of hardware. This includes input devices like mice, keyboards, touchscreens, and output devices such as monitors and printers.

Device drivers play a crucial role in enabling communication between the operating system and various hardware components.

Screen-based Controls:

Screen-based controls refer to the user interface elements displayed on the screen. These can include buttons, sliders, checkboxes, and other interactive elements.

Modern Windows interfaces often use a combination of touch-friendly controls for devices like tablets and traditional controls for desktop users.

Components – Text and Messages:

Text and messages are essential components of any user interface. Windows displays text for labels, buttons, error messages, and other communication with the user.

Consistent and clear messaging helps users understand the actions they are performing and any issues that may arise.

Icons and Images:

Icons are visual representations of actions, applications, or files. They help users quickly identify and interact with different elements.

High-quality and meaningful icons contribute to a user-friendly experience.

Multimedia:

Windows supports multimedia features, including audio and video playback. Multimedia applications and codecs allow users to enjoy a wide range of content.

Colors and Color Selection:

Color plays a significant role in user interface design. Windows allows for the customization of color schemes, and users can choose their preferred color settings.

Designers should consider accessibility and readability when selecting colors to ensure a positive user experience.

Choosing Colors Material:

The choice of colors in user interface design should align with the application's purpose and brand identity.

Consideration should be given to color contrast, accessibility standards, and the emotional impact of different colors on users.

Usability Problems:

Usability problems can arise from inconsistent design, confusing navigation, unclear messaging, or poor color choices.

User testing and feedback play a crucial role in identifying and resolving usability issues.

Multimedia in HCI:

Enhanced Communication:

Multimedia elements, such as images, videos, and audio, can enhance communication by providing additional

channels for conveying information.
They are particularly useful for presenting complex data or instructions in a more intuitive and engaging manner.

Engagement and Interactivity:
Multimedia elements can enhance user engagement and interactivity, making the interaction with the system more dynamic and enjoyable.
Storytelling and Context:
Multimedia can be employed to create a narrative or provide contextual information, aiding in a better understanding of the content or tasks at hand.

Colors in HCI:
Aesthetic Appeal:
Colors contribute to the overall aesthetic appeal of an interface, influencing users' perceptions and emotions.
Visual Hierarchy:
Different colors can be used to establish a visual hierarchy, guiding users' attention to important elements and helping them navigate through the interface more efficiently.
Branding and Consistency:
Consistent use of colors contributes to brand identity and recognition. It helps users associate certain colors with specific actions or elements.
Common Problems in HCI Related to Multimedia and Colors:
Accessibility:
Inappropriate use of colors or relying solely on multimedia can create accessibility issues. It's crucial to consider users with color vision deficiencies or those who rely on screen readers.

Cognitive Load:
Excessive use of multimedia or a wide range of colors can lead to cognitive overload. It's important to strike a balance and ensure that the design doesn't overwhelm users with too much information.
Inconsistent Design:
Inconsistency in color schemes and multimedia elements across different parts of the interface can confuse users. Consistency is key for a seamless user experience.
Clashing Colors:
Poorly chosen color combinations can result in low readability and visual discomfort. Designers need to consider color contrast and legibility to ensure readability for all users.

Choosing Colors in HCI:
Color Psychology:
Understand the psychological impact of colors. For example, warm colors (reds, oranges) can evoke energy and excitement, while cool colors (blues, greens) can promote calmness.
Contrast and Readability:
Ensure sufficient contrast between text and background colors to enhance readability. This is especially important for users with visual impairments.
Consistency:
Establish a consistent color scheme throughout the interface to create a cohesive and user-friendly design.
User Testing:
Conduct user testing to gather feedback on color choices. Different users may have varied preferences and sensitivities to certain colors.

UNIT IV

HCI IN THE SOFTWARE PROCESS:

It is therefore necessary that we go beyond the exercise of identifying paradigms and examine the process of interactive system design. In the previous chapter we introduced some of the elements of a user-centered design process. Here we expand on that process, placing the design of interactive systems within the established frameworks of software development.
Within computer science there is already a large subdiscipline that addresses the management and technical issues of the development of software systems – called software engineering. One of the cornerstones of software engineering is the software life cycle, which describes the activities that take place from the initial concept formation for a software system up until its eventual phasing out and replacement.
This is not intended to be a software engineering textbook, so it is not our major concern here to discuss in depth all of the issues associated with software engineering and the myriad life-cycle models.
The important point that we would like to draw out is that issues from HCI affecting the usability of interactive systems are relevant within all the activities of the software life cycle. Therefore, software engineering for interactive system design is not simply a matter of adding one more activity that slots in nicely with the existing activities in the life cycle. Rather, it involves techniques that span the entire life cycle.

THE SOFTWARE LIFE CYCLE:

One of the claims for software development is that it should be considered as an engineering discipline, in a way similar to how electrical engineering is considered for hardware development.

Activities in the life cycle

A more detailed description of the life cycle activities is depicted in Figure 6.1. The graphical representation is reminiscent of a waterfall, in which each activity naturally leads into the next. The analogy of the waterfall is not completely faithful to the real relationship between these activities, but it provides a good starting point for discussing the logical flow of activity. We describe the activities of this waterfall model of the software life cycle next.

Requirements specification

In requirements specification, the designer and customer try to capture a description of what the eventual system will be expected to provide. This is in contrast to determining how the system will provide the expected services, which is the concern of later activities. Requirements specification involves eliciting information from the customer about the work environment, or domain, in which the final product will function. Aspects of the work domain include not only the particular functions that the software product must perform but also details about the environment in which it must operate, such as the people whom it will potentially affect and the new product's relationship to any other products which it is updating or replacing.
Requirements specification begins at the start of product development. Software product they must be formulated in a language suitable for implementation.

Requirements are usually initially expressed in the native language of the customer. The executable languages for software are less natural and are more closely related to a mathematical language in which each term in the language has a precise interpretation, or semantics.

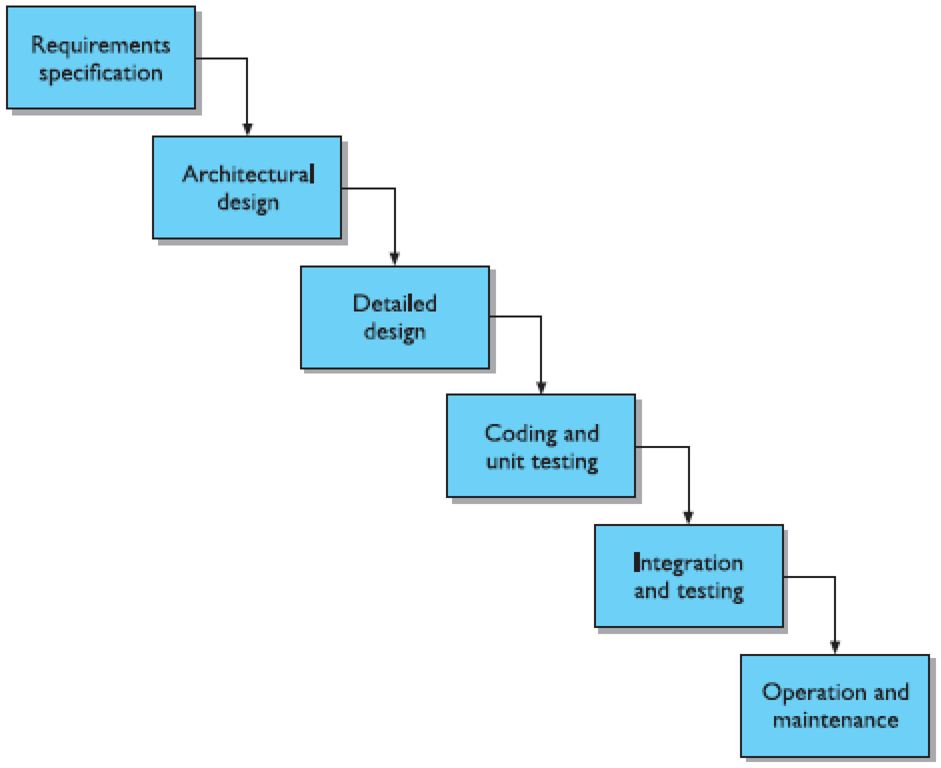


Figure 6.1 The activities in the waterfall model of the software life cycle

Detailed design

The architectural design provides a decomposition of the system description that allows for isolated development of separate components which will later be integrated.

For those components that are not already available for immediate integration, the designer must provide a sufficiently detailed description so that they may be implemented in some programming language. The detailed design is a *refinement* of the component description provided by the architectural design. The behaviour implied by the higher-level description must be preserved in the more detailed description.

Coding and unit testing

The detailed design for a component of the system should be in such a form that it is possible to implement it in some executable programming language. After coding, the component can be tested to verify that it performs correctly, according to some test criteria that were determined in earlier activities. Research on this activity within the life cycle has concentrated on two areas.

Integration and testing

Once enough components have been implemented and individually tested, they must be integrated as described in the architectural design. Further testing is done to ensure correct behavior and acceptable use of any shared resources. It is also possible at this time to perform some acceptance testing with the customers to ensure that the system meets their requirements. It is only after acceptance of the integrated system that the product is finally released to the customer.

Maintenance

After product release, all work on the system is considered under the category of maintenance, until such time as a new version of the product demands a total redesign or the product is phased out entirely.

Validation and verification

Throughout the life cycle, the design must be checked to ensure that it both satisfies the high-level requirements agreed with the customer and is also complete and internally consistent. These checks are referred to as validation and verification, respectively. Boehm [36a] provides a useful distinction between the two, characterizing validation as designing 'the right thing' and verification as designing 'the thing right'.

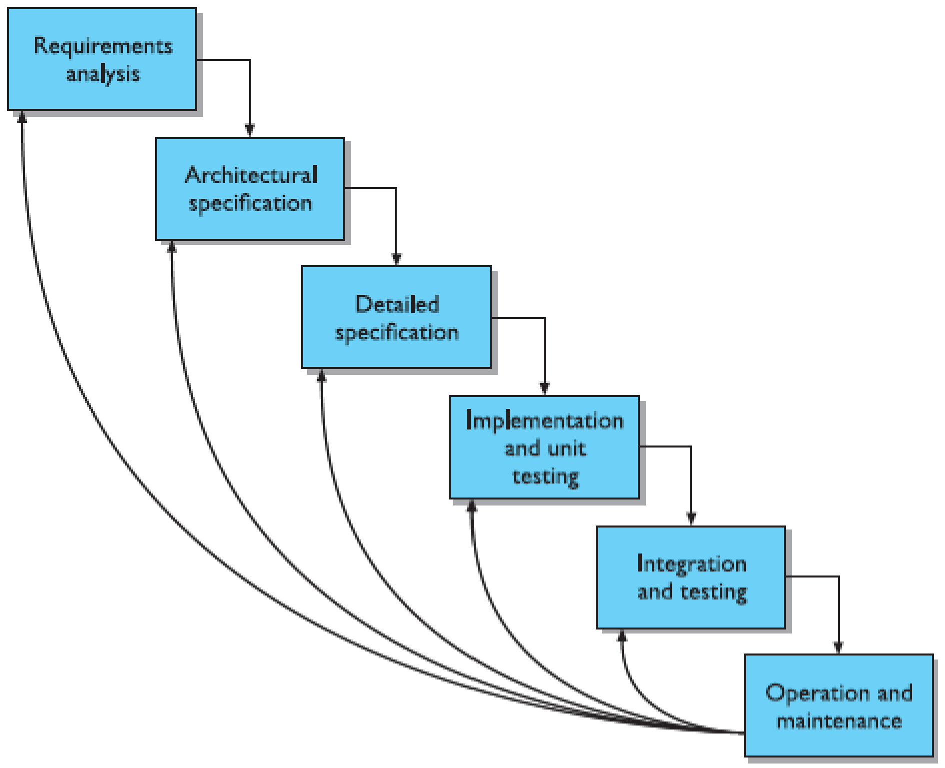

Requirements analysis
Architectural specification
Detailed specification
Implementation and unit testing
Integration and testing
Operation and maintenance

Figure 6.2 Feedback from maintenance activity to other design activities

Validation proofs are much trickier, as they almost always involve a transformation between languages. Furthermore, the origin of customer requirements arises in the inherent ambiguity of the real world and not the mathematical world. This precludes the possibility of objective proof, rigorous or formal. Instead, there will always be a leap from the informal situations of the real world to any formal and structured development process. We refer to this inevitable disparity as the *formality gap*, depicted in Figure 6.3. The formality gap means that validation will always rely to some extent on subjective means of proof. We can increase our confidence in the subjective proof by effective use of real-world experts in performing certain validation chores.

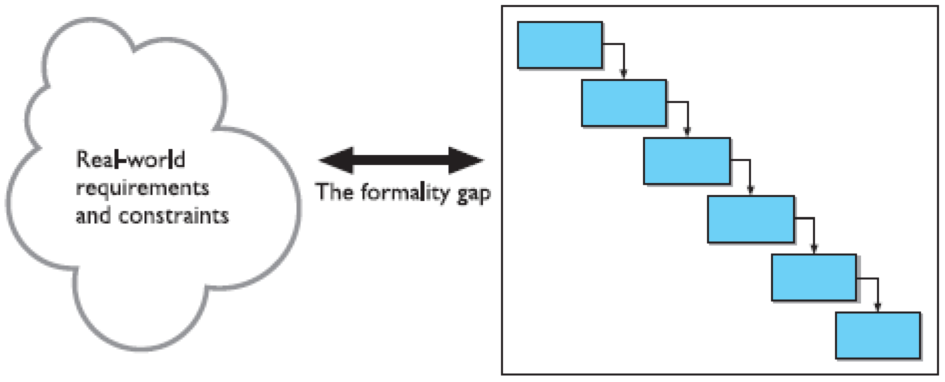


Figure 6.3 The formality gap between the real world and structured design

INTERACTIVE SYSTEMS AND THE SOFTWARE LIFE CYCLE:

The traditional software engineering life cycles arose out of a need in the 1960s and 1970s to provide structure to the development of large software systems. In those days, the majority of large systems produced were concerned with data-processing applications in business. These systems were not highly interactive; rather, they were batch-processing systems. Consequently, issues concerning usability from an enduser's perspective were not all that important. With the advent of personal computing in the late 1970s and its huge commercial success and acceptance, most modern systems developed today are much more interactive, and it is vital to the success of any product that it be easy to operate for someone who is not expected to know much about how the system was designed. The modern user has a great amount of skill in the work that he performs without necessarily having that much skill in software development.
the use of notations and techniques that support the user's perspective of the interactive system. We discussed earlier the purpose of validation and the formality gap.
It is very difficult for an expert on human cognition to predict the cognitive demands that an abstract design would require of the intended user if the notation for the design does not reflect the kind of information the user must recall in order to interact. The same holds for assessing the timing behavior of an abstract design that does not explicitly mention the timing characteristics of the operations to be invoked or their relative ordering. Though no structured development process will entirely eliminate the formality gap, the particular notations

used can go a long way towards making validation of non-functional requirements feasible with expert assistance.

Our models of the psychology and sociology of the human and human cognition, whether in isolation or in a group, are incomplete and do not allow us to predict how to design for maximum usability.

This dearth of predictive psychological theory means that in order to test certain usability properties of their designs, designers must observe how actual users interact with the developed product and measure their performance. In order for the results of those observations to be worthwhile, the experiments must be as close to a real interaction situation as possible.

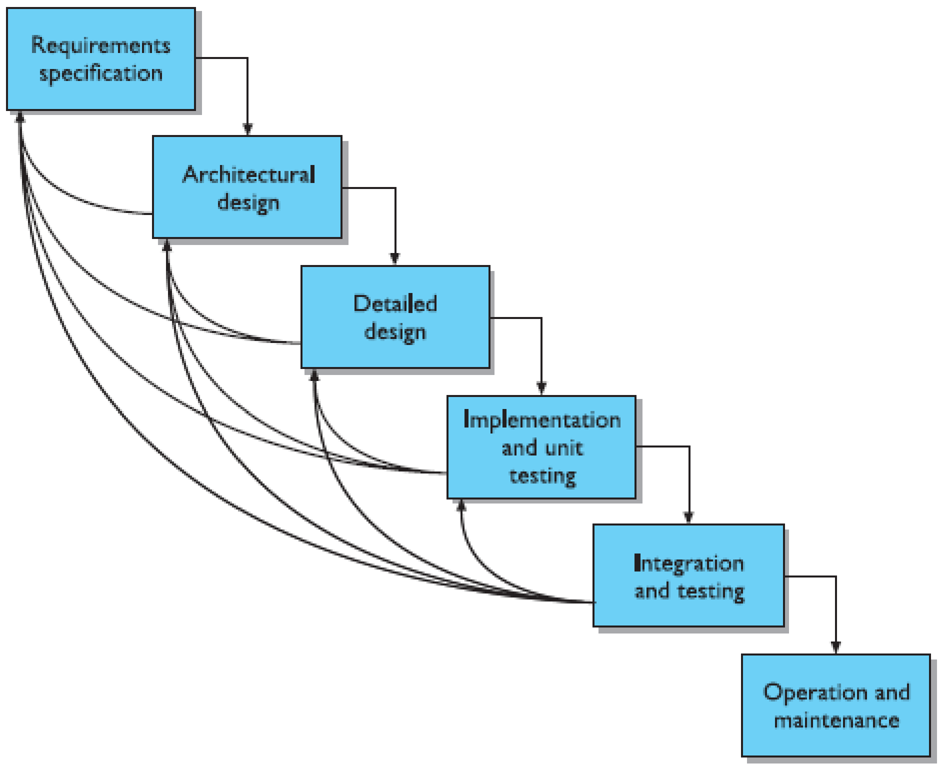

Requirements specification
Architectural design
Detailed design
Implementation and unit testing
Integration and testing
Operation and maintenance

Figure 6.4 Representing iteration in the waterfall model

Usability Engineering:
One approach to user-centered design has been the introduction of explicit usability engineering goals into the design process, as suggested by Whiteside and colleagues at IBM and Digital Equipment Corporation [377] and by Nielsen at Bellcore [260, 261]. Engineering depends on interpretation against a shared background of meaning, agreed goals and an understanding of how satisfactory completion will be judged. The emphasis for usability engineering is in knowing exactly what criteria will be used to judge a product for its usability.

The ultimate test of a product’s usability is based on measurements of users’ experience with it. Therefore, since a user’s direct experience with an interactive system is at the physical interface, focus on the actual user interface is understandable.
The danger with this limited focus is that much of the work that is accomplished in interaction involves more than just the surface features of the systems used to perform that work. In reality, the whole functional architecture of the system and the cognitive capacity of the users should be observed in order to arrive at meaningful measures. But it is not at all simple to derive measurements of activity beyond the physical actions in the world, and so usability engineering is limited in its application.
Sample usability specification for undo with a VCR
Attribute: Backward recoverability
Measuring concept: Undo an erroneous programming sequence
Measuring method: Number of explicit user actions to undo current program
Now level: No current product allows such an undo
Worst case: As many actions as it takes to program in mistake
Planned level: A maximum of two explicit user actions
Best case: One explicit cancel action

Problems with usability engineering
The major feature of usability engineering is the assertion of explicit usability metrics early on in the design process which can be used to judge a system once it is delivered.

ITERATIVE DESIGN AND PROTOTYPING:

A point we raised earlier is that requirements for an interactive system cannot be completely specified from the beginning of the life cycle. The only way to be sure about some features of the potential design is to build them and test them out on real users. The design can then be modified to correct any false assumptions that were revealed in the testing. This is the essence of iterative design, a purposeful design process which tries to overcome the inherent problems of incomplete requirements specification by cycling through several designs, incrementally improving upon the final product with each pass.
The problems with the design process, which lead to an iterative design philosophy, are not unique to the usability features of the intended system. The problem holds for requirements specification in general, and so it is a general software engineering problem, together with technical and managerial issues.
On the technical side, iterative design is described by the use of prototypes, artifacts that simulate or animate some but not all features of the intended system. There are three main approaches to prototyping:

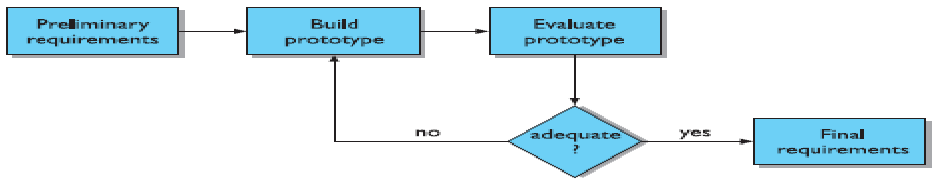


Figure 6.5 Throw-away prototyping within requirements specification

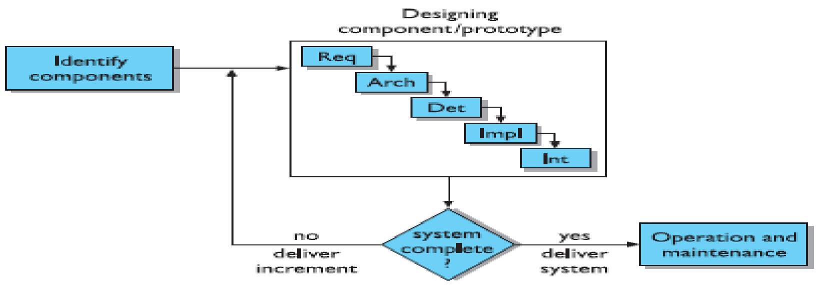


Figure 6.6 Incremental prototyping within the life cycle

Incremental The final product is built as separate components, one at a time. There is one overall design for the final system, but it is partitioned into independent and smaller components. The final product is then released as a series of products, each subsequent release including one more component. This is depicted in Figure 6.6.

Evolutionary Here the prototype is not discarded and serves as the basis for the next iteration of design. In this case, the actual system is seen as evolving from a very limited initial version to its final release, as depicted in Figure 6.7. Evolutionary prototyping also fits in well with the modifications which must be made to the system that arise during the operation and maintenance activity in the life cycle.

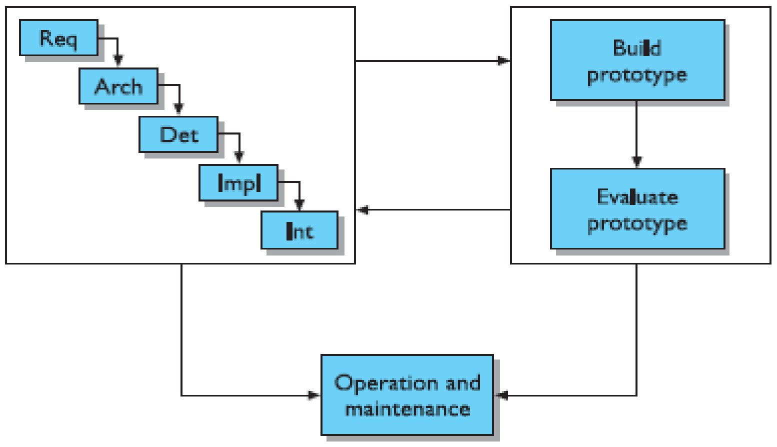


Figure 6.7 Evolutionary prototyping throughout the life cycle

Design Rationale:

In designing any computer system, many decisions are made as the product goes from a set of vague customer requirements to a deliverable entity. Often it is difficult to recreate the reasons, or rationale, behind various design decisions.

In an explicit form, a design rationale provides a communication mechanism among the members of a design team so that during later stages of design and/or maintenance it is possible to understand what critical decisionswere made, what alternatives were investigated (and, possibly, in what order) and the reason why onealternative was chosen over the others. This can help avoid incorrect assumptions later.

Accumulated knowledge in the form of design rationales for a set of products can be reused to transfer what has worked in one situation to another situation which has similar needs. The design rationale can capture the context of a design decision in order that a different design team can determine if a similar rationale is appropriate for their product.

The effort required to produce a design rationale forces the designer to deliberate more carefully about design decisions. The process of deliberation can be assisted by the design rationale technique by suggesting how arguments justifying or discarding a particular design option are formed.

Process-oriented design rationale

Much of the work on design rationale is based on Rittel’s issue-based information system, or IBIS, a style for representing design and planning dialog developed in the 1970s [308]. In IBIS (pronounced ‘ibbiss’), a hierarchical structure to a design rationale is created.

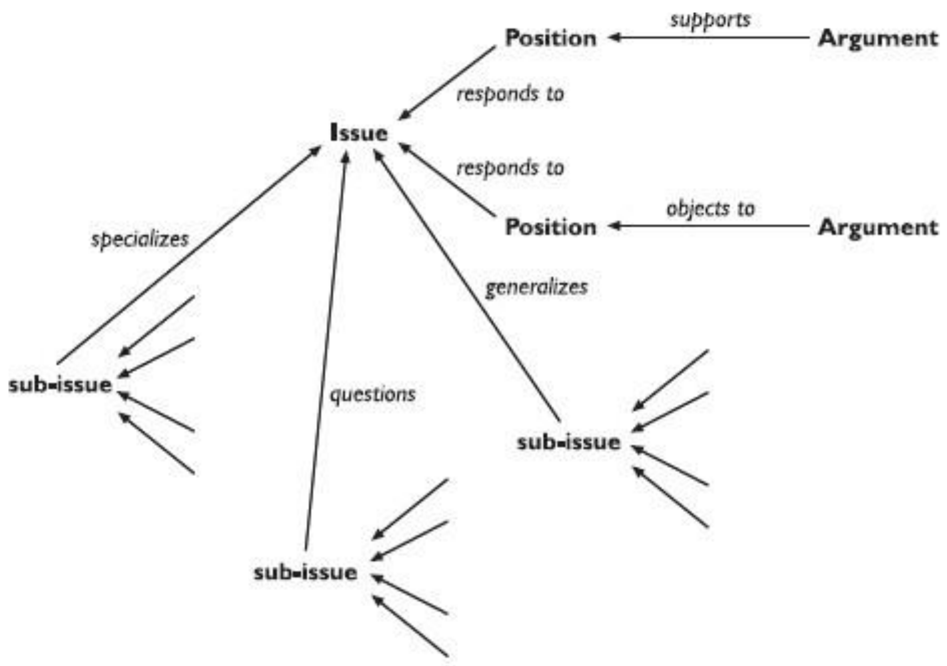


Figure 6.8 The structure of a gIBIS design rationale

Design space analysis

MacLean and colleagues [222] have proposed a more deliberative approach to design rationale which emphasizes a post hoc structuring of the space of design alternatives that have been considered in a design project. Their approach, embodied in the Questions, Options and Criteria (QOC) notation, is characterized as design space analysis.

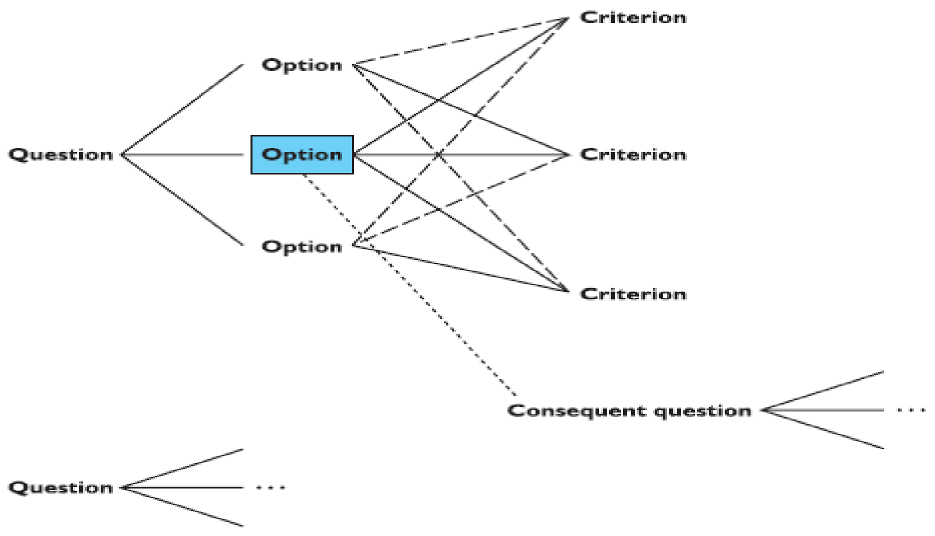


Figure 6.9 The QOC notation

Design Rules:

One of the central problems that must be solved in a user-centered design process is how to provide designers with the ability to determine the usability consequences of their design decisions. We require design rules, which are rules a designer can follow in order to increase the usability of the eventual software product. We can classify these rules along two dimensions, based on the rule's authority and generality.

We will first discuss abstract principles, then go on to consider in more depth some examples of standards and guidelines for user-centered design. Finally, we will consider some well-known heuristics or 'golden rules' which, it has been suggested, provide a succinct summary of the essence of good design. We end the chapter with a discussion of design patterns, a relatively new approach to capturing design knowledge in HCI.

Principles to Support Usability:

The most abstract design rules are general principles, which can be applied to the design of an interactive system in order to promote its usability. In Chapter 4 we looked at the different paradigms that represent the development of interactive systems. Derivation of principles for interaction has usually arisen out of a need to explain why a paradigm is successful and when it might not be. Principles can provide the repeatability which paradigms in themselves cannot provide. In this section we present a collection of usability principles. Since it is too bold an objective to produce a comprehensive catalog of such principles, our emphasis will be on structuring the presentation of usability principles in such a way that the catalog can be easily extended as our knowledge increases.

The principles we present are first divided into three main categories:

Learnability – the ease with which new users can begin effective interaction and achieve maximal performance.

Flexibility – the multiplicity of ways in which the user and system exchange information.

Robustness – the level of support provided to the user in determining successful achievement and assessment of goals.

Table 7.2 Summary of principles affecting flexibility

Principle	Definition	Related principles
Dialog initiative	Allowing the user freedom from artificial constraints on the input dialog imposed by the system	System/user pre-emptiveness
Multi-threading	Ability of the system to support user interaction pertaining to more than one task at a time	Concurrent vs. interleaving, multi-modality
Task migratability	The ability to pass control for the execution of a given task so that it becomes either internalized by the user or the system or shared between them	–
Substitutivity	Allowing equivalent values of input and output to be arbitrarily substituted for each other	Representation multiplicity, equal opportunity
Customizability	Modifiability of the user interface by the user or the system	Adaptivity, adaptability

Table 7.3 Summary of principles affecting robustness

Principle	Definition	Related principles
Observability	Ability of the user to evaluate the internal state of the system from its perceivable representation	Browsability, static/dynamic defaults, reachability, persistence, operation visibility
Recoverability	Ability of the user to take corrective action once an error has been recognized	Reachability, forward/backward recovery, commensurate effort
Responsiveness	How the user perceives the rate of communication with the system	Stability
Task conformance	The degree to which the system services support all of the tasks the user wishes to perform and in the way that the user understands them	Task completeness, task adequacy

GOLDEN RULES AND HEURISTICS:

There are many sets of heuristics, but the most well used are Nielsen's ten heuristics, Shneiderman's eight golden rules and Norman's seven principles. Nielsen's heuristics are intended to be used in evaluation and will therefore be discussed in Chapter 9. We will consider the other two sets here.

7.5.1 Shneiderman’s Eight Golden Rules of Interface Design Shneiderman’s eight golden rules provide a convenient and succinct summary of the key principles of interface design.
1. Strive for consistency in action sequences, layout, terminology, command use and so on.
2. Enable frequent users to use shortcuts, such as abbreviations, special key sequences and macros, to perform regular, familiar actions more quickly.
3. Offer informative feedback for every user action, at a level appropriate to the magnitude of the action.
4. Design dialogs to yield closure so that the user knows when they have completed a task.
5. Offer error prevention and simple error handling so that, ideally, users are prevented from making mistakes and, if they do, they are offered clear and informative instructions to enable them to recover.
6. Permit easy reversal of actions in order to relieve anxiety and encourage exploration, since the user knows that he can always return to the previous state.
7. Support internal locus of control so that the user is in control of the system, which responds to his actions.
8. Reduce short-term memory load by keeping displays simple, consolidating multiple page displays and providing time for learning action sequences. These rules provide a useful shorthand for the more detailed sets of principles described earlier. Like those principles, they are not applicable to every eventuality and need to be interpreted for each new situation. However, they are broadly useful and their application will only help most design projects.
7.5.2 Norman’s Seven Principles for Transforming Difficult Tasks into Simple Ones In Chapter 3 we discussed Norman’s execution–evaluation cycle, in which he elaborates the seven stages of action. Later, in his classic book The Design of Everyday Things, he summarizes user-centered design using the following seven principles:
1. Use both knowledge in the world and knowledge in the head. People work better when the knowledge they need to do a task is available externally – either explicitly or through the constraints imposed by the environment. But experts also need to be able to internalize regular tasks to increase their efficiency. So systems should provide the necessary knowledge within the environment and their operation should be transparent to support the user in building an appropriate mental model of what is going on.
2. Simplify the structure of tasks. Tasks need to be simple in order to avoid complex problem solving and excessive memory load. There are a number of ways to simplify the structure of tasks.

HCI PATTERNS:

As we observed in Chapter 4, one way to approach design is to learn from examples that have proven to be successful in the past: to reuse the knowledge of what made a system – or paradigm – successful. Patterns are an approach to capturing and reusing this knowledge – of abstracting the essential details of successful design so that these can be applied again and again in new situations.
Patterns originated in architecture, where they have been used successfully, and they are also used widely in software development to capture solutions to common programming problems. More recently they have been used in interface and web design.
A pattern is an invariant solution to a recurrent problem within a specific context. Patterns address the problems that designers face by providing a ‘solution statement’. This is best illustrated by example.

Alexander, who initiated the pattern concept, proposes a pattern for house building called 'Light on Two Sides of Every Room'. The problem being addressed here is that When they have a choice, people will always gravitate to those rooms which have light on two sides, and leave the rooms which are lit only from one side unused and empty.

Evaluation Techniques:

Evaluation should not be thought of as a single phase in the design process (still less as an activity tacked on the end of the process if time permits). Ideally, evaluation should occur throughout the design life cycle, with the results of the evaluation feeding back into modifications to the design. Clearly, it is not usually possible to perform extensive experimental testing continuously throughout the design, but analytic and informal techniques can and should be used. In this respect, there is a close link between evaluation and the principles and prototyping techniques we have already discussed – such techniques help to ensure that the design is assessed continually.

Goals of Evaluation:

Evaluation has three main goals: to assess the extent and accessibility of the system's functionality, to assess users' experience of the interaction, and to identify any specific problems with the system.

The system's functionality is important in that it must accord with the user's requirements. In other words, the design of the system should enable users to perform their intended tasks more easily. This includes not only making the appropriate functionality available within the system, but making it clearly reachable by the user in terms of the actions that the user needs to take to perform the task.

Evaluation through Expert Analysis:

Cognitive walkthrough

Cognitive walkthrough was originally proposed and later revised by Polson and colleagues [294, 376] as an attempt to introduce psychological theory into the informal and subjective walkthrough technique.

The origin of the cognitive walkthrough approach to evaluation is the code walkthrough familiar in software engineering. Walkthroughs require a detailed review of a sequence of actions. In the code walkthrough, the sequence represents a segment of the program code that is stepped through by the reviewers to check certain characteristics (for example, that coding style is adhered to, conventions for spelling variables versus procedure calls, and to check that system-wide invariants are not violated). In the cognitive walkthrough, the sequence of actions refers to the steps that an interface will require a user to perform in order to accomplish some known task. The evaluators then 'step through' that action sequence to check it for potential usability problems. Usually, the main focus of the cognitive walkthrough is to establish how easy a system is to learn. More specifically, the focus is on learning through exploration. Experience shows that many users prefer to learn how to use a system by exploring its functionality hands on, and not after sufficient training or examination of a user's manual. So the checks that are made during the walkthrough ask questions that address this exploratory learning. To do this, the evaluators go through each step in the task and provide a 'story' about why that step is or is not good for a new user. To do a walkthrough (the term walkthrough from now on refers to the cognitive walkthrough, and not to any other kind of walkthrough), you need four things:

1. A specification or prototype of the system. It doesn't have to be complete, but it should be fairly detailed. Details such as the location and wording for a menu can make a big difference.
2. A description of the task the user is to perform on the system. This should be a representative task that most users will want to do.
3. A complete, written list of the actions needed to complete the task with the proposed system.
4. An indication of who the users are and what kind of experience and knowledge the evaluators can assume about them.

Given this information, the evaluators step through the action sequence (identified in item 3 above) to critique the system and tell a believable story about its usability. To do this, for each action, the evaluators try to answer the following four questions for each step in the action sequence

Heuristic evaluation

A heuristic is a guideline or general principle or rule of thumb that can guide a design decision or be used to critique a decision that has already been made. Heuristic evaluation, developed by Jakob Nielsen and Rolf Molich, is a method for structuring the critique of a system using a set of relatively simple and general heuristics.

Heuristic evaluation can be performed on a design specification so it is useful for evaluating early design. But it can also be used on prototypes, storyboards and fully functioning systems. It is therefore a flexible, relatively cheap approach. Hence it is often considered a discount usability technique.

Model-based evaluation

A third expert-based approach is the use of models. Certain cognitive and design models provide a means of combining design specification and evaluation into the same framework. These are discussed in detail in Chapter 12. For example, the GOMS (goals, operators, methods and selection) model predicts user performance with a particular interface and can be used to filter particular design options. Similarly, lower-level modeling techniques such as the keystroke-level model provide predictions of the time users will take to perform low-level physical tasks.

Evaluation through User Participation:

The techniques we have considered so far concentrate on evaluating a design or system through analysis by the designer, or an expert evaluator, rather than testing with actual users. However, useful as these techniques are for filtering and refining the design, they are not a replacement for actual usability testing with the people for whom the system is intended:

Styles of evaluation

Before we consider some of the techniques that are available for evaluation with users, we will distinguish between two distinct evaluation styles: those performed under laboratory conditions and those conducted in the work environment or 'in the field'.

Field studies

The second type of evaluation takes the designer or evaluator out into the user's work environment in order to observe the system in action. Again this approach has its pros and cons.

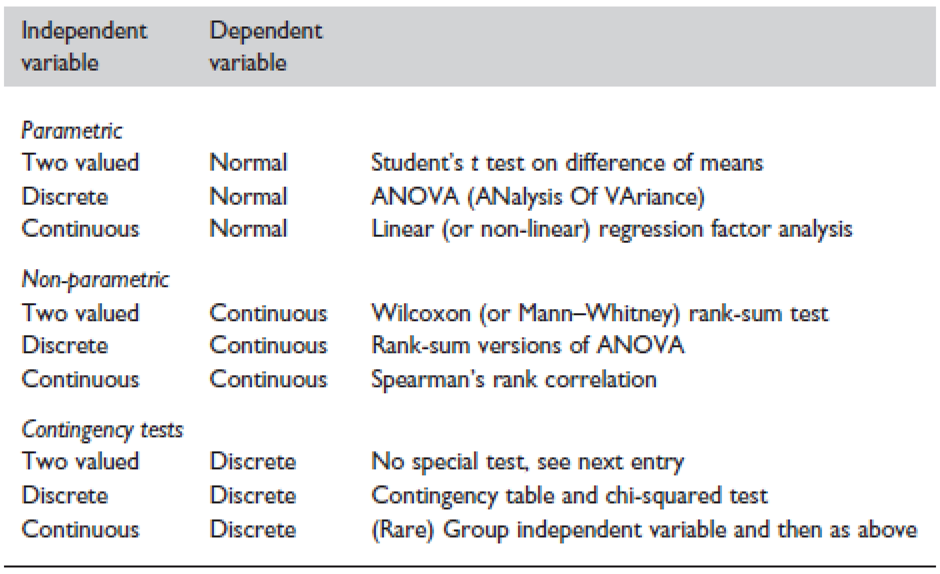
Empirical methods: experimental evaluation:

One of the most powerful methods of evaluating a design or an aspect of a design is to use a controlled experiment. This provides empirical evidence to support a particular claim or hypothesis. It can be used to study a wide range of different issues at different levels of detail.

Any experiment has the same basic form. The evaluator chooses a hypothesis to test, which can be determined by measuring some attribute of participant behavior.

A number of experimental conditions are considered which differ only in the values of certain controlled variables. Any changes in the behavioral measures are attributed to the different conditions. Within this basic form there are a number of factors that are important to the overall reliability of the experiment, which must be considered carefully in experimental design. These include the participants chosen, the variables tested and manipulated, and the hypothesis tested.

Table 9.1 Choosing a statistical technique



Independent variable	Dependent variable	
Parametric		
Two valued	Normal	Student's *t* test on difference of means
Discrete	Normal	ANOVA (ANalysis Of VAriance)
Continuous	Normal	Linear (or non-linear) regression factor analysis
Non-parametric		
Two valued	Continuous	Wilcoxon (or Mann–Whitney) rank-sum test
Discrete	Continuous	Rank-sum versions of ANOVA
Continuous	Continuous	Spearman's rank correlation
Contingency tests		
Two valued	Discrete	No special test, see next entry
Discrete	Discrete	Contingency table and chi-squared test
Continuous	Discrete	(Rare) Group independent variable and then as above

Query techniques

Another set of evaluation techniques relies on asking the user about the interface directly. Query techniques can be useful in eliciting detail of the user's view of a system. They embody the philosophy that states that the best way to find out how a system meets user requirements is to 'ask the user'. They can be used in evaluation and more widely to collect information about user requirements and tasks. The advantage of such methods is that they get the user's viewpoint directly and may reveal issues that have not been considered by the designer.

Evaluation through monitoring physiological responses:

One of the problems with most evaluation techniques is that we are reliant on observation and the users telling us what they are doing and how they are feeling. What if we were able to measure these things directly? Interest has grown recently in the use of what is sometimes called objective usability testing, ways of monitoring physiological aspects of computer use. Potentially this will allow us not only to see more clearly exactly what users do when they interact with computers, but also to measure how they feel. The two areas receiving the most attention to date are eye tracking and physiological measurement.

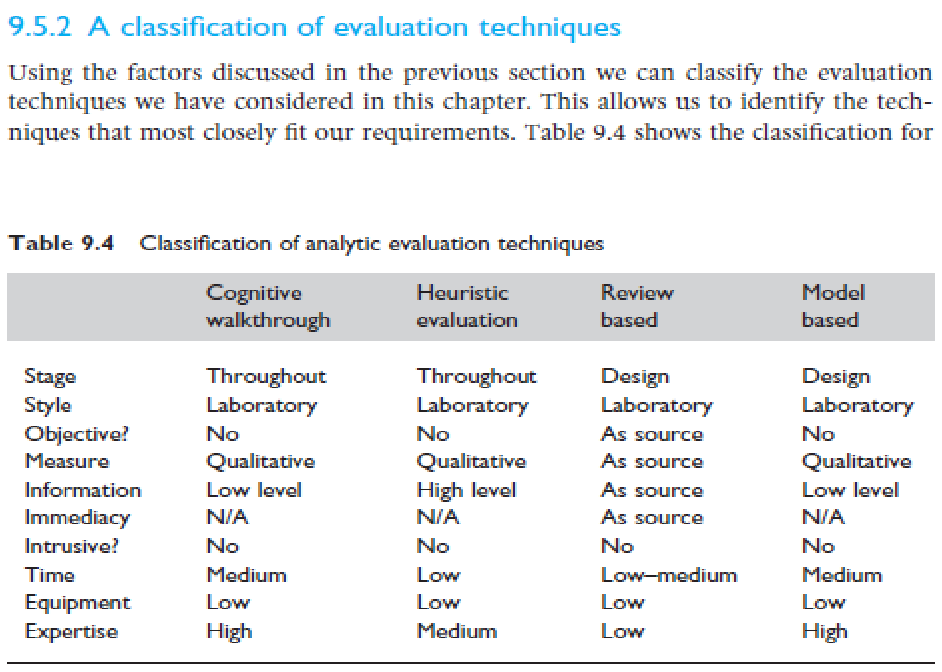
Choosing An Evaluation Method:

As we have seen in this chapter, a range of techniques is available for evaluating an interactive system at all stages in the design process. So how do we decide which methods are most appropriate for our needs? There are no hard and fast rules in this – each method has its particular strengths and weaknesses and each is useful if applied appropriately. However, there are a number of factors that should be taken into account when selecting evaluation techniques. These also provide a way of categorizing the different methods so that we can compare and choose between them. In this final section we will consider these factors.

Factors distinguishing evaluation techniques

We can identify at least eight factors that distinguish different evaluation techniques and therefore help us to make an appropriate choice. These are:

- the stage in the cycle at which the evaluation is carried out
- the style of evaluation
- the level of subjectivity or objectivity of the technique
- the type of measures provided
- the information provided
- the immediacy of the response
- the level of interference implied
- the resources required.



9.5.2 A classification of evaluation techniques

Using the factors discussed in the previous section we can classify the evaluation techniques we have considered in this chapter. This allows us to identify the techniques that most closely fit our requirements. Table 9.4 shows the classification for

Table 9.4 Classification of analytic evaluation techniques

	Cognitive walkthrough	Heuristic evaluation	Review based	Model based
Stage	Throughout	Throughout	Design	Design
Style	Laboratory	Laboratory	Laboratory	Laboratory
Objective?	No	No	As source	No
Measure	Qualitative	Qualitative	As source	Qualitative
Information	Low level	High level	As source	Low level
Immediacy	N/A	N/A	As source	N/A
Intrusive?	No	No	No	No
Time	Medium	Low	Low–medium	Medium
Equipment	Low	Low	Low	Low
Expertise	High	Medium	Low	High

Universal Design:

Universal design is the process of designing products so that they can be used by as many people as possible in as many situations as possible. In our case, this means particularly designing interactive systems that are usable by anyone, with any range of abilities, using any technology platform. This can be achieved by designing systems either to have built in redundancy or to be compatible with assistive technologies. An example of the former might be an interface that has both visual and audio access
to commands; an example of the latter, a website that provides text alternatives for graphics, so that it can be read using a screen reader.

Universal Design Principles:

We have defined universal design as 'the process of designing products so that they can be used by as many people as possible in as many situations as possible'. But what does that mean in practice? Is it possible to design anything so that anyone can use it – and if we could, how practical would it be? Wouldn't the cost be prohibitive? In reality, we may not be able to design everything to be accessible to everyone, and we certainly cannot ensure that everyone has the same experience of using a product, but we can work toward the aim of universal design and try to provide an equivalent experience**.**

Multi-Modal Interaction:

As we have seen in the previous section, providing access to information through more than one mode of interaction is an important principle of universal design.
Such design relies on multi-modal interaction.
As we saw in Chapter 1, there are five senses: sight, sound, touch, taste and smell. Sight is the predominant sense for the majority of people, and most interactive systems consequently use the visual channel as their primary means of presentation, through graphics, text, video and animation.
However, sound is also an important channel, keeping us aware of our surroundings, monitoring people and events around us, reacting to sudden noises, providing clues and cues that switch our attention from one thing to another. It can also have an emotional effect on us, particularly in the case of music. Music is almost completely an auditory experience, yet is able to alter moods, conjure up visual images, evoke atmospheres or scenes in the mind of the listener.
Touch, too, provides important information: tactile feedback forms an intrinsic part of the operation of many common tools – cars, musical instruments, pens, anything that requires holding or moving. It can form a sensuous bond between
individuals, communicating a wealth of non-verbal information. Taste and smell are often less appreciated (until they are absent) but they also provide useful information in daily life: checking if food is bad, detecting early signs of fire, noticing that manure has been spread in a field, pleasure Examples of the use of sensory information are easy to come by (we looked at some in Chapter 1), but the important point is that our everyday interaction with each other and the world around us is multi-sensory, each sense providing different information that informs the whole.

Sound in the interface:

Sound is an important contributor to usability. There is experimental evidence to suggest that the addition of audio confirmation of modes, in the form of changes in keyclicks, reduces errors [237]. Video games offer further evidence, since experts tend to score less well when the sound is turned off than when it is on; they pick up vital clues and information from the sound while concentrating their visual attention on different things. The dual presentation of information through sound and vision supports universal design, by enabling access for users with visual and hearing

impairments respectively. It also enables information to be accessed in poorly lit or noisy environments. Sound can convey transient information and does not take up screen space, making it potentially useful for mobile applications.

Touch in the interface:

We have already considered the importance of touch in our interaction with our environment, in Chapter 1. Touch is the only sense that can be used to both send and receive information. Although it is not yet widely used in interacting with computers, there is a significant research effort in this area and commercial applications are becoming available.

The use of touch in the interface is known as haptic interaction. Haptics is a generic term relating to touch, but it can be roughly divided into two areas: cutaneous perception, which is concerned with tactile sensations through the skin; and kinesthetics, which is the perception of movement and position. Both are useful in interaction but they require different technologies.

Handwriting recognition:

Like speech, we consider handwriting to be a very natural form of communication. The idea of being able to interpret handwritten input is very appealing, and handwriting appears to offer both textual and graphical input using the same tools. There are problems associated with the use of handwriting as an input medium, however, and in this section we shall consider these. We will first look at the mechanisms for capturing handwritten information, and then look at the problems of interpreting it.

Gesture recognition:

Gesture is a component of human–computer interaction that has become the subject of attention in multi-modal systems. Being able to control the computer with certain movements of the hand would be advantageous in many situations where there is no possibility of typing, or when other senses are fully occupied. It could also support communication for people who have hearing loss, if signing could be ‘translated’ into speech or vice versa. But, like speech, gesture is user dependent, subject to variation and co-articulation. The technology for capturing gestures is expensive, using either computer vision or a special dataglove (see Chapter 2). The dataglove provides easier access to highly accurate information, but is a relatively intrusive technology, requiring the user to wear the special Lycra glove. The interpretation of the sampled data is very difficult, since segmenting the gestures causes problems. A team from Toronto [131] has produced a gesture recognition system that translates hand movements into synthesized speech, using five neural networks working in parallel to learn and then interpret different parts of the inputs.

Designing For Diversity:

Designing for users with disabilities

It is estimated that at least 10% of the population of every country has a disability that will affect interaction with computers. Employers and manufacturers of computing equipment have not only a moral responsibility to provide accessible products, but often also a legal responsibility. In many countries, legislation now demands that the workplace must be designed to be accessible or at least adaptable to all – the design of software and hardware should not unnecessarily restrict the job prospects of people with disabilities.

Hearing impairment:

Compared with a visual disability where the impact on interacting with a graphical interface is immediately obvious, a hearing impairment may appear to have little impact on the use of an interface. After all, it is the visual not the auditory channel that is predominantly used. To an extent this is true, and computer technology can actually enhance communication opportunities for people with hearing loss. Email and instant messaging are great levellers and can be used equally by hearing and deaf users alike.

Universal design is about designing systems that are accessible by all users in all circumstances, taking account of human diversity in disabilities, age and culture.

Universal design helps everyone – for example, designing a system so that it can be used by someone who is deaf or hard of hearing will benefit other people working in noisy environments or without audio facilities. Designing to be accessible to screenreading systems will make websites better for mobile users and older browsers.

Multi-modal systems provide access to system information and functionality through a range of different input and output channels, exploiting redundancy.

Such systems will enable users with sensory, physical or cognitive impairments to make use of the channels that they can use most effectively. But all users benefit from multi-modal systems that utilize more of our senses in an involving interactive experience.

For any design choice we should ask ourselves whether our decision is excluding someone and whether there are any potential confusions or misunderstandings in our choice.

UNIT V

COGNITIVE MODELS:

The techniques and models in this chapter all claim to have some representation of users as they interact with an interface; that is, they model some aspect of the user understands, knowledge, intentions or processing. The level of representation differs from technique to technique – from models of high-level goals and the results of problem-solving activities, to descriptions of motor-level activity, such as keystrokes and mouse clicks. The formalisms have largely been developed by psychologists, or computer scientists, whose interest is in understanding user behavior.

Competence models, therefore, represent the kinds of behavior expected of a user, but they provide little help in analyzing that behavior to determine its demands on the user. Performance models provide analytical power mainly by focussing on routine behavior in very limited applications.
Another useful distinction between these models is whether they address the acquisition or formulation of a plan of activity or the execution of that plan. Referring back to the interaction framework presented in Chapter 3, this classification would mean that some models are concerned with understanding the User and his associated task language while others are concerned with the articulation translation between that task language and the Input language. The presentation of the cognitive models in this chapter follows this classification scheme, divided into the following categories:

- hierarchical representation of the user's task and goal structure
- linguistic and grammatical models
- physical and device-level models.

GOAL AND TASK HIERARCHIES:

Many models make use of a model of mental processing in which the user achieves goals by solving subgoals in a divide-and-conquer fashion. We will consider two models, GOMS and CCT, where this is a central feature. Imagine we want to produce a report on sales of introductory HCI textbooks. To achieve this goal we divide it into several subgoals, say gathering the data together, producing the tables and histograms, and writing the descriptive material.
Concentrating on the data gathering, we decide to split this into further subgoals: find the names of all introductory HCI textbooks and then search the book sales database for these books. Similarly, each of the other subgoals is divided up into further subgoals, until some level of detail is found at which we decide to stop. We thus end up with a hierarchy of goals and subgoals. The example can be laid out to expose this structure:
These two questions are issues of granularity, and both of the methods described below leave this to some extent in the hands of the designer. Different design issues demand different levels of analysis. However, both methods operate at a relatively low level; neither would attempt to start with such an abstract goal as 'produce a

report' which will involve real creativity and difficult problem solving. Instead they confine themselves to more routine learned behavior. This most abstract task is referred to as the unit task. The unit task does not require any problem-solving skills on the part of the user, though it frequently demands quite sophisticated problem-solving skills on the part of the designer to determine them.

GOMS:

The GOMS model of Card, Moran and Newell is an acronym for Goals, Operators, Methods and Selection [56]. A GOMS description consists of these four elements: Goals These are the user's goals, describing what the user wants to achieve.

Further, in GOMS the goals are taken to represent a 'memory point' for the user, from which he can evaluate what should be done and to which he may return should any errors occur.

Operators These are the lowest level of analysis. They are the basic actions that the user must perform in order to use the system. They may affect the system (for example, press the 'X' key) or only the user's mental state (for example, read the dialog box). There is still a degree of flexibility about the granularity of operators; we may take the command level 'issue the SELECT command' or be more primitive: 'move mouse to menu bar, press center mouse button . . .'.

Methods As we have already noted, there are typically several ways in which a goal can be split into subgoals. For instance, in a certain window manager a currently selected window can be closed to an icon either by selecting the 'CLOSE' option from a pop-up menu, or by hitting the 'L7' function key. In GOMS these two goal decompositions are referred to as methods, so we have the CLOSE-METHOD and the L7-METHOD:

DESIGN FOCUS:

GOMS saves money

Some years ago the US telephone company NYNEX were intending to install a new computer system to support their operators. Before installation a detailed GOMS analysis was performed taking into account the cognitive and physical processes involved in dealing with a call. The particular technique was rather different from the original GOMS notation as described here. Because an operator performs several activities in parallel a PERT-style GOMS description was constructed [192, 154]. The PERT analysis was used to determine the critical path, and hence the time to complete a typical task. It was discovered that rather than speeding up operations, the new system would take longer to process each call. The new system was abandoned before installation, leading to a saving of many millions of dollars.

Cognitive complexity theory

Cognitive complexity theory, introduced by Kieras and Polson [199], begins with the basic premises of goal decomposition from GOMS and enriches the model to provide more predictive power. CCT has two parallel descriptions: one of the user's goals and the other of the computer system (called the device in CCT). The description of the user's goals is based on a GOMS-like goal hierarchy, but is expressed primarily using production rules. We introduced production rules in Chapter 1 and we further describe their use in CCT below. For the system grammar, CCT uses generalized transition networks, a form of state transition network. This will not be described here, but state transition networks will be discussed in detail in Chapter 16.

The production rules are a sequence of rules: if condition then action where condition is a statement about the contents of working memory. If the condition is true then the production rule is said to fire. An action may consist of one or more elementary actions, which may be either changes to the working memory, or external actions such as keystrokes. The production rule 'program' is written in a LISP-like language.
As an example, we consider an editing task using the UNIX vi text editor. The task is to insert a space where one has been missed out in the text, for instance if we noticed that in the above paragraph we had written 'cognitivecomplexity theory'. This is a reasonably frequent typing error and so we assume that we have developed good procedures to perform the task. We consider a fragment of the associated CCT production rules.

```
(SELECT-INSERT-SPACE
IF (AND (TEST-GOAL perform unit task)
(TEST-TEXT task is insert space)
(NOT (TEST-GOAL insert space))
(NOT (TEST-NOTE executing insert space)) )
THEN ( (ADD-GOAL insert space)
(ADD-NOTE executing insert space)
(LOOK-TEXT task is at %LINE %COL) ))
(INSERT-SPACE-DONE
IF (AND (TEST-GOAL perform unit task)
(TEST-NOTE executing insert space)
(NOT (TEST-GOAL insert space)) )
THEN ( (DELETE-NOTE executing insert space)
(DELETE-GOAL perform unit task)
(UNBIND %LINE %COL) ))
(INSERT-SPACE-1
IF (AND (TEST-GOAL insert space)
(NOT (TEST-GOAL move cursor))
(NOT (TEST-CURSOR %LINE %COL)) )
THEN ( (ADD-GOAL move cursor to %LINE %COL) ))
(INSERT-SPACE-2
IF (AND (TEST-GOAL insert space)
(TEST-CURSOR %LINE %COL) )
THEN ( (DO-KEYSTROKE 'I')
(DO-KEYSTROKE SPACE)
(DO-KEYSTROKE ESC)
(DELETE-GOAL insert space) ))
```

To see how these rules work, imagine that the user has just seen the typing mistake
and thus the contents of working memory (w.m.) are

(GOAL perform unit task)
(TEXT task is insert space)
(TEXT task is at 5 23)
(CURSOR 8 7)

Problems and extensions of goal hierarchies:

The formation of a goal hierarchy is largely a post hoc technique and runs a very real risk of being defined by the computer dialog rather than the user. One way to rectify this is to produce a goal structure based on pre-existing manual procedures and thus obtain a natural hierarchy [201]. To be fair, GOMS defines its domain to be that of expert use, and thus the goal structures that are important are those which users develop out of their use of the system. However, such a natural hierarchy may be particularly useful as part of a CCT analysis, representing a very early state of knowledge.

LINGUISTIC MODELS:

BNF

Representative of the linguistic approach is Reisner's use of Backus–Naur Form (BNF) rules to describe the dialog grammar [301]. This views the dialog at a purely syntactic level, ignoring the semantics of the language. BNF has been used widely to specify the syntax of computer programming languages, and many system dialogs can be described easily using BNF rules. For example, imagine a graphics system that has a
line-drawing function. To select the function the user must select the 'line' menu option. The line-drawing function allows the user to draw a polyline, that is a sequence of line arcs between points. The user selects the points by clicking the mouse button in the drawing area. The user double clicks to indicate the last point of the polyline.

```
draw-line ::= select-line + choose-points
+ last-point
select-line ::= position-mouse + CLICK-MOUSE
choose-points ::= choose-one
| choose-one + choose-points
choose-one ::= position-mouse + CLICK-MOUSE
last-point  ::=  position-mouse  +  DOUBLE-CLICK-MOUSE
position-mouse ::= empty | MOVE-MOUSE + position-mouse
```

The names in the description are of two types: non-terminals, shown in lower case, and terminals, shown in upper case. Terminals represent the lowest level of user behavior, such as pressing a key, clicking a mouse button or moving the mouse.

Non-terminals are higher-level abstractions. The non-terminals are defined in terms of other non-terminals and terminals by a definition of the form

name ::= expression

The '::=' symbol is read as 'is defined as'. Only non-terminals may appear on the
left of a definition. The right-hand side is built up using two operators '+' (sequence)
and '|' (choice). For example, the first rule says that the non-terminal draw-line

is defined to be select-line followed by choose-points followed by lastpoint.
All of these are non-terminals, that is they do not tell us what the basic user actions are. The second rule says that select-line is defined to be position mouse (intended to be over the 'line' menu entry) followed by CLICK-MOUSE. This is our first terminal and represents the actual clicking of a mouse.

Task–action grammar
Measures based upon BNF have been criticized as not 'cognitive' enough. They ignore the advantages of consistency both in the language's structure and in its use of command names and letters. Task–action grammar (TAG) [284] attempts to deal with some of these problems by including elements such as parametrized grammar rules to emphasize consistency and encoding the user's world knowledge (for example, up is the opposite of down).
To illustrate consistency, we consider the three UNIX commands: cp (for copying files), mv (for moving files) and ln (for linking files). Each of these has two possible forms. They either have two arguments, a source and destination filename, or have any number of source filenames followed by a destination directory:

The Challenge of Display-Based Systems:

Both goal hierarchical and grammar-based techniques were initially developed when most interactive systems were command line, or at most, keyboard and cursor based.
There are significant worries, therefore, about how well these approaches can generalize to deal with more modern windowed and mouse-driven interfaces.

Both families of techniques largely ignore system output – what the user sees.
The implicit assumption is that the users know exactly what they want to do and execute the appropriate command sequences blindly. There are exceptions to this.
We have already mentioned how Reisner's BNF has been extended to include assertions about output. In addition, TAG has been extended to include information about how the display can affect the grammar rules [180].
Another problem for grammars is the lowest-level lexical structure. Pressing a cursor key is a reasonable lexeme, but moving a mouse one pixel is less sensible. In addition, pointer-based dialogs are more display oriented. Clicking a cursor at a particular point on the screen has a meaning dependent on the current screen contents.
This problem can be partially resolved by regarding operations such as 'select region of text' or 'click on quit button' as the terminals of the grammar. If this approach is taken, the detailed mouse movements and parsing of mouse events in the context of display information (menus, etc.) are abstracted away.
Physical and Device Models:
Keystroke-level model

Compared with the deep cognitive understanding required to describe problem solving activities, the human motor system is well understood. KLM (Keystroke-Level Model [55]) uses this understanding as a basis for detailed predictions about user performance. It is aimed at unit tasks within interaction – the execution of simple command sequences, typically taking no more than 20 seconds. Examples of this would be using a search and replace feature, or changing the font of a word. It does not extend to complex actions such as producing a diagram. The assumption is that these more complex tasks would be split into subtasks (as in GOMS) before the user attempts to map them into physical actions. The task is split into two phases:
acquisition of the task, when the user builds a mental representation of the task;
execution of the task using the system's facilities.
KLM only gives predictions for the latter stage of activity. During the acquisition
phase, the user will have decided how to accomplish the task using the primitives of
the system, and thus, during the execution phase, there is no high-level mental activity
– the user is effectively expert. KLM is related to the GOMS model, and can be
thought of as a very low-level GOMS model where the method is given.
The model decomposes the execution phase into five different physical motor
operators, a mental operator and a system response operator:
K Keystroking, actually striking keys, including shifts and other modifier keys.
B Pressing a mouse button.
P Pointing, moving the mouse (or similar device) at a target.
H Homing, switching the hand between mouse and keyboard.
D Drawing lines using the mouse.
M Mentally preparing for a physical action.
R System response which may be ignored if the user does not have to wait for it, as
in copy typing.
The execution of a task will involve interleaved occurrences of the various operators.
For instance, imagine we are using a mouse-based editor. If we notice a single character error we will point at the error, delete the character and retype it, and then return to our previous typing point. This is decomposed as follows:
1. Move hand to mouse H[mouse]
2. Position mouse after bad character PB[LEFT]
3. Return to keyboard H[keyboard]
4. Delete character MK[DELETE]
5. Type correction K[char]
6. Reposition insertion point H[mouse]MPB[LEFT]
Notice that some operators have descriptions added to them, representing which
device the hand homes to (for example, [mouse]) and what keys are hit (for example,
LEFT – the left mouse button).

The model predicts the total time taken during the execution phase by adding the component times for each of the above activities. For example, if the time taken for one keystroke is tK, then the total time doing keystrokes is

TK = 2tK

Similar calculations for the rest of the operators give a total time of

Texecute = TK + TB + TP + TH + TD + TM + TR

= 2tK + 2tB + tP + 3tH + 0 + 2tM + 0

In this example, the system response time was zero. However, if the user had to wait for the system then the appropriate time would be added. In many typing tasks, the user can type ahead anyway and thus there is no need to add response times. Where needed, the response time can be measured by observing the system.

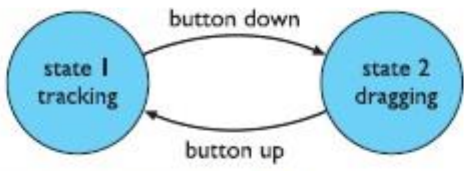


Figure 12.1 Mouse transitions: states 1 and 2

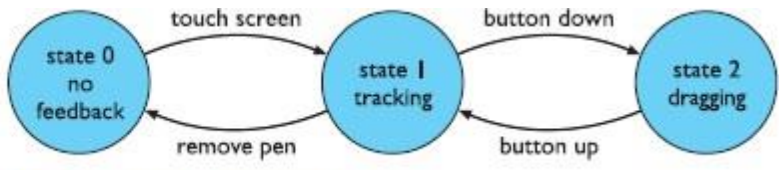


Figure 12.2 Light pen transitions: three states

COGNITIVE ARCHITECTURES:

The formalisms we have seen so far have some implicit or explicit model of how the user performs the cognitive processing involved in carrying out a task. For instance, the concept of taking a problem and solving it by divide and conquer using subgoals is central to GOMS. CCT assumes the distinction between long- and short-term memory, with production rules being stored in long-term memory and 'matched' against the contents of short-term (or working) memory to determine which 'fire'. The values for various motor and mental operators in KLM were based on the Model Human Processor (MHP) architecture of Card, Moran and Newell [56]. Another common assumption, which we have not discussed in this chapter, is the distinction between linguistic levels – semantic, syntactic and lexical – as an architectural model of the user's understanding.

The problem space model

Rational behavior is characterized as behavior that is intended to achieve a specific goal. This element of rationality is often used to distinguish between intelligent and machine-like behavior. In the field of artificial intelligence (AI), a system exhibiting rational behavior is referred to as a knowledge-level system. A knowledge-level system contains an agent behaving in an environment. The agent has knowledge about itself and its environment, including its own goals. It can perform certain actions and sense information about its changing environment. As the agent behaves in its environment, it changes the environment and its own knowledge. We can view the overall behavior of the knowledge-level system as a sequence of environment and agent states as they progress in time. The goal of the agent is characterized as a preference over all possible sequences of agent/environment states.

Interacting cognitive subsystems

Barnard has proposed a very different cognitive architecture, called interacting cognitive subsystems (ICS) [24, 25, 27]. ICS provides a model of perception, cognition and action, but unlike other cognitive architectures, it is not intended to produce a description of the user in terms of sequences of actions that he performs. ICS provides a more holistic view of the user as an information-processing machine. The emphasis is on determining how easy particular procedures of action sequences become as they are made more automatic within the user.

ICS attempts to incorporate two separate psychological traditions within one cognitive architecture. On the one hand is the architectural and general-purpose information-processing approach of short-term memory research. On the other hand is the computational and representational approach characteristic of psycholinguistic research and AI problem-solving literature.

Ubiquitous Computing and Augmented Realities:

There are several ways in which the earliest assumptions of HCI are challenged. For example, we no longer assume there is a single user; rather, we consider groups and larger organizational concerns when discussing interactions. In this chapter, we challenge another assumption concerning the form factor of the computing device. Traditionally, we think of computers as a glass box, a workstation with keyboard, mouse and monitor sitting on a desk that we seek out when we want to do some work. Since the late 1980s, this traditional form factor has been expanded to include a variety of more mobile devices and computing services that are distributed throughout the physical world and more tightly integrated with it. The trend is towards a ubiquitous or pervasive computing experience, in which computing devices become so commonplace that we do not distinguish them from the 'normal' physical surroundings. Improved display technologies give us the ability to augment the physical world with electronic information through head-mounted displays or steerable projection surfaces. We have also seen display technologies that provide complete virtual replacements of the physical world, creating a so-called virtual reality.

Ubiquitous Computing Applications Research:

We first introduced the notion of ubiquitous computing (or pervasive computing) in Chapter 4. The interest in ubiquitous computing has surged over the past few years, thanks to some influential writings and plenty of experimental work. The defining characteristic of ubiquitous computing is the attempt to break away from the traditional desktop interaction paradigm and move computational power into the environment that surrounds the user. Rather than force the user to search out and find the computer's interface, ubiquitous computing suggests that the interface itself can take on the responsibility of locating and serving the user. Mark Weiser is credited with coining the phrase ubiquitous computing (or ubicomp) when he put forth a vision of people and environments augmented with computational resources that provide information and services when and where desired [369]. Though his vision has excited many technologists, it is important to realize that the main motivation behind Weiser's vision was centered on the impact ubicomp would have on the human experience:

Defining the appropriate physical interaction experience

Ubiquitous computing inspires application development that is 'off the desktop'. In addition to suggesting a freedom from a small number of well-defined interaction locales (the desktop), this vision assumes that physical interaction between humans and computation will be less like the current desktop keyboard/mouse/display paradigm and more like the way humans interact with the physical world. Humans speak, gesture and use writing implements to communicate with other humans and alter physical artifacts. The drive for a ubiquitous computing experience has resulted in a variety of important changes to the input, output and interactions that define the human experience with computing. We describe three of those changes in this section.

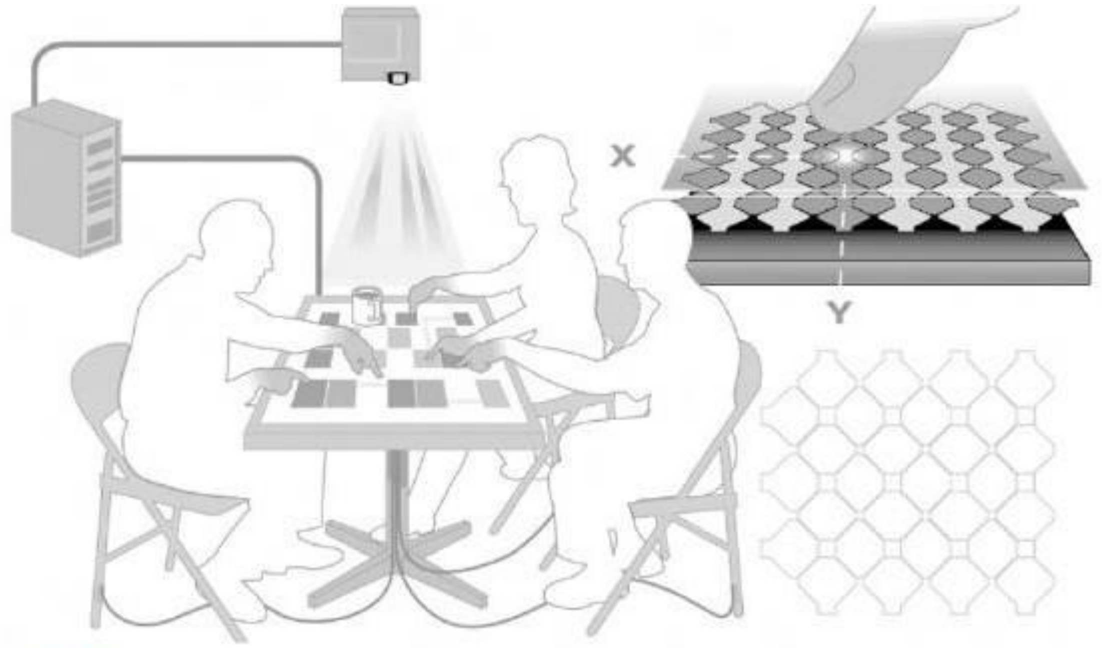


Figure 20.1 The DiamondTouch input technology from Mitsubishi Electric Research Lab (MERL) uses capacitive coupling through humans to provide a large-scale input surface for multiple simultaneous users. See www.merl.com/projects/DiamondTouch/ for more details. Source: Courtesy of Mitsubishi Electric Research Laboratories, Inc.

These recognition technologies are some examples of interpreting meaning from sensed signals of human activity. There are many other ways to infer information about people and environments by sensing a variety of other physical signals. There have been many recent advances in sensing the physical world; the significance here is that sensing and interpretation of human activity provides a more implicit notion of input to an interactive system. For example, many researchers have investigated how simple sensors such as RFID (the technology behind the security tags used on library books, clothes in shops, etc.), accelerometers, tilt sensors, capacitive coupling,infrared range finders and others can be incorporated into artifacts to increase the language of input from the user to control that artefact.

Seamless integration of physical and virtual worlds:
An important feature of ubicomp technology is that it attempts to merge computational artifacts smoothly with the world of physical artifacts. There have been plenty of examples demonstrating how electronic information can be overlaid upon the real world, thus producing an augmented reality [130].

Application themes for ubicomp:
Many applications-focussed researchers in HCI seek the holy grail of ubicomp, the killer app that will cause significant investment in the infrastructure that will then enable a wide variety of ubicomp applications to flourish. It could be argued that person– person communication is such a killer app for ubicomp, as it has caused a large investment in environmental and personal infrastructure that has moved us close (though not entirely) to a completely connected existence. Whether or not personal communication is the killer app, the vision of ubicomp from the human perspective is much more holistic.
Context-aware computing:
Two compelling early demonstrations of ubicomp were the Olivetti Research Lab's Active Badge [361] and the Xerox PARCTab [362], both location-aware appliances.
These devices leverage a simple piece of context, user location, and provide valuable services (automatic call forwarding for a phone system, automatically updated maps of user locations in an office). This technology was also the basis for the Pepys automatic diary system described in Chapter 18 (Section 18.4.1). These simple locationaware appliances are perhaps the first demonstration of linking implicit human activity with computational services that serve to augment general human activity.
Location of identifiable entities (usually people) is a very common piece of context used in ubicomp application development. The most widespread applications have been GPS-based car navigation systems and handheld 'tour guide' systems that vary the content displayed (video or audio) by a handheld unit given the user's physical location in an exhibit area [6, 68]. For example, The Sentient Computing Project uses a 3-D ultrasonic indoor location system to track each worker in a building and so maintain a map of office worker locations that helps coworkers find each other and talk by phone.

Virtual and Augmented Reality:
Virtual reality (VR) refers to the computer-generated simulation of a world, or a subset of it, in which the user is immersed. It represents the state of the art in multimedia systems, but concentrates on the visual senses. VR allows the user to experience situations that are too dangerous or expensive to enter 'in the flesh'. Users may explore the real world at a different scale and with hidden features made visible.

Alternatively, the virtual worlds that are generated may be entirely synthesized: realistic within themselves, but purely a manifestation of electronic structures. The term ‘virtual reality’ conjures up an image of a user weighed down with a helmet or goggles, grasping, apparently blindly, into empty space. The user, isolated within his virtual environment, moves through a simulated landscape, picking up objects on the way. This is fully immersive VR. However, it is only one part of the spectrum of VR, which also includes desktop VR, command and control situations, and augmented reality, where virtuality and reality meet.

VR technology

The technology involved in VR is quite elaborate. The individual devices have been discussed in Chapter 2, but now we shall see how they work together.

Since the user has to ‘see’ a new environment, a headset is usually used in a VR setup. With independent screens for each eye, in order to give a 3D image, the headset is often a large, relatively cumbersome piece of head-mounted gear. However, smaller, lighter VR goggles are now available and may soon become only slightly oversized spectacles.

Immersive VR:

Virtual reality can be used to produce environments that mimic our everyday world. Architects have always used models and sketches to show clients how a building will appear. Now they can use VR to take clients through a virtual tour of the building, fly over it from above, look at it from the streets outside, enter the doors and walk through the corridors. Similar techniques are used to plan kitchens and even gardens.

However, there are also many things that we cannot see, either because they are invisible to the naked eye (heat, magnetism, gravity) or because they are too small or too large. Scientific and data visualization systems make use of VR technology to expose and explore these features. In Section 20.4 we will see one example of this in the virtual wind tunnel, which makes the airflows around an aircraft wing visible using virtual bubble trails (see Figure 20.6). Another example is in the field of protein chemistry, where individual molecules are, of course, too small to see except by electron microscope.

VR on the desktop and in the home:

Virtual reality has been made possible by the advent of very fast high-performance computers. Despite the exponential rise in processor speeds, high-resolution immersive VR is still not available for mass-market applications, and many systems are primarily research projects. Desktop VR is a lower-cost alternative. In desktop VR, 3D images are presented on a normal computer screen and manipulated using mouse and keyboard, rather than using goggles and datagloves. Many readers may have used such systems on personal computers or games consoles: flight simulators, or interactive games such as DOOM or MYST.

Command and control:

In many command and control situations, real users are at one remove from the physical world, seeing it through windows, or cameras. The windows or video screens can be replaced by synthesized pictures. The user operates within an immediate physical environment, with real controls and instruments, but the world outside is virtual.

One such interactive VR application in widespread use is the flight simulator. A full cockpit system is placed in a hydraulically supported container, with large screens replacing the cockpit windows. Images are generated and projected onto the screens, whilst the box can be moved rapidly in any direction by the hydraulic rams.
The visual information and physical motion simulate accurately the conditions encountered by aircraft. Flight simulators are used extensively in pilot training programs.
Landings can be practiced, with the system responding to the commands of the pilot; descending too fast and off to one side, the pilot will have to correct the situation if she wishes to avoid a crash. Emergency situations can also be created, in which aircraft system malfunctions can be artificially created in order to train the pilot to take the correct course of corrective or life-preserving action. With VR, entertainment is never far behind and this kind of system can also be found in many fun fairs!.

AUGMENTED REALITY:

In augmented reality systems electronic images are projected over the real world – virtuality and reality meet. The head-up displays in many aircraft and even some automobiles can be regarded as an example of this, but the data in such displays are not typically connected to the objects seen through them and, hence, the blend between virtuality and reality is quite weak.
A stronger sense of connection can be obtained using semi-transparent goggles. Users can move around in the real world and see real objects, but computer images are reflected off the inside of the glass and overlay the physical objects. Again, this can be used to show unrelated information; for example, some wearable computers allow users to read their email whilst walking around. However, the real sense of two worlds meeting comes when the projected image in some way links or refers to the object it overlays. For example, one experimental system has virtual balls, which can be picked up and thrown by the user [11]. When the virtual ball 'hits' the real wall it bounces off. The balls can even bounce down a real staircase.

INFORMATION AND DATA VISUALIZATION:

Virtual reality and 3D displays can be used to visualize scientific data and other complex information. Whether or not 3D representations are used, animation techniques, especially when under interactive user control, can give a sense of engagement with data, and encourage discovery and pattern formation.

Scientific and technical data:

Three-dimensional representations of scientific and technical data can be classified by the number of dimensions in the virtual world that correspond to physical spatial dimensions, as opposed to those that correspond to more abstract parameters. Perhaps the most engaging images are where all three dimensions have some physical validity. An example of this is the virtual wind tunnel [50]. In a physical wind tunnel, an accurate model of an aircraft is constructed and then subjected to winds that, when appropriately scaled, correspond to realistic situations. The intention is to investigate patterns of air movement and pressure, for example to discover those places where turbulence forms. Of course, air is invisible, so small pieces of ribbon may be attached to the aircraft surface, small bubbles released into the chamber or polarized light used to expose the hidden airflows. In the virtual wind tunnel, air movements are calculated using the equations of fluid dynamics. An engineer can then see the simulated aircraft using VR goggles and can move around a (virtual) baton from which stream (virtual) bubbles (Figure 20.6). By moving the baton to different parts of the aircraft, areas of interest can be investigated.

Structured information:

Scientific data are typically numeric, so can easily be mapped onto a dimension in virtual space. In contrast, the data sets that arise in information systems typically have many discrete attributes and structures: hierarchies, networks and, most complex of all, free text. Examples of hierarchies include file trees and organization charts. Examples of networks include program flow charts and hypertext structures.

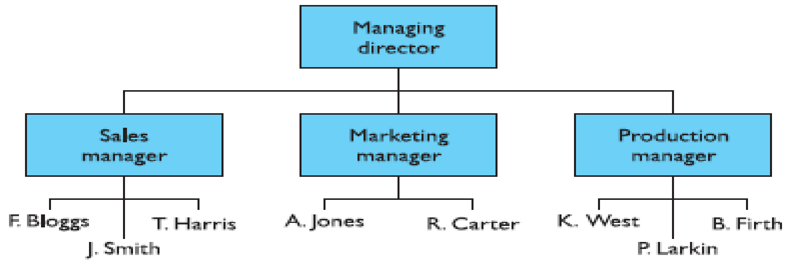


Figure 20.7 Two-dimensional organization chart

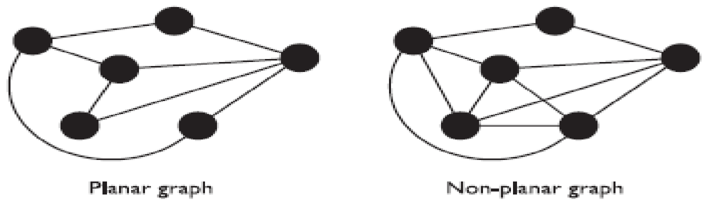


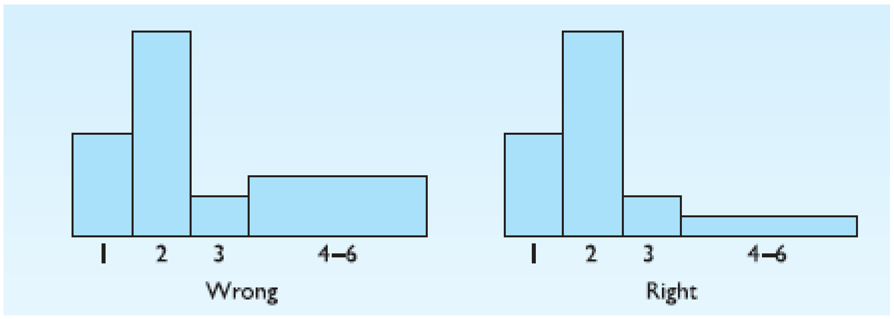
Figure 20.8 Two-dimensional network layout

Time and interactivity:

We can consider time and visualization from two sides. On the one hand, many data sets include temporal values (dates, periods, etc.) that we wish to visualize. On the other hand, the passage of time can itself be used in order to visualize other types of data. In 2D graphs, time is often mapped onto one spatial dimension; for example, showing the monthly sales figures of a company. Where the time-varying data are themselves 2D images, multiple snapshots can be used. Both comic books and technical manuals use successive images to show movement and changes, often augmented by arrows, streamlines or blurring to give an impression of direction and speed. Another type of temporal data is where events occur at irregular intervals. Timelines are often used for this sort of data, where one dimension is used to represent time and the second axis is used to represent the type of activity.

Getting the size right

When faced with a 2D histogram it is the area that we perceive as being the size of a bar, not the height. Survey data of family groups at a fun fair have shown that 25% had only one child, 50% had two children, 10% had three children and the remaining 15% of families had between four and six children.



DESIGN FOCUS:

Getting the size right

If we display the height of the 4–6 group proportional to the percentage of families it makes it look as if this is much more than 15% of the data. This is because the column is three times wider than the rest. We actually perceive the data to be in the ratio 25:50:10:45. The right thing to do is to draw the area of the columns proportional to the number of families, which makes the data look right.

Similar problems arise in 3D representations. If we draw a map with 3D columns proportional to the population rising from each country, we will probably not be able to see anything except a single enormous block for China! We should display the heights of the blocks proportional to the population density. Of course, if we start with density data, we can simply use height to start with.

If data ranges are extremely large, we may use non-linear (for example, logarithmic) scales. Users clearly need to be aware of this, but at least if density data are used, bigger areas/volumes represent bigger data.