SPPU Sem 5 Information Technology

New Syllabus 2021-22

With Typical MCQ's

Compulsory Subject (314444)

Human Computer Interaction

Prof. Savita Vijay Lade

- With Solved Latest UNIVERSITY QUESTION PAPERS.
- Every unit Comprises of Short & Essay Questions with Solutions.
- Exhaustive Coverage of Topics from Examination Point of View.
- All concepts are elaborated with graphical representation for easy understanding.
- FREE DOWNLOAD Sample chapter from our Android App.

e-book

TECH-NEO PUBLICATIONS
Where Authors Inspire Innovation
A Sachin Shah Venture
www.techneobooks.in
info@techneobooks.in

Human Computer Interaction
P5-47B

Price ₹ 245/-

UNIT I

CHAPTER 1

Introduction

Syllabus

What is HCI ? Disciplines involved in HCI, Why HCI study is important ? The psychology of everyday things Donald A. Norman, Principles of HCI, User-centered Design. Measurable Human factors.

1.1 WHAT IS HCI ?

GQ. Explain the term HCI. (5 Marks)

- Human-computer interaction (HCI) is a multidisciplinary field, that focuses on the design and use of computer technology, particularly the interfaces between people (users) and computers.
- HCI is the study of the interaction between people, computers and tasks. It involves the development and application of principals, guidelines and methods to support the design and evaluation of interactive systems.
- HCI brings along the expertise from behavioral science, computer science, cognitive psychology, and design to recognize and initiate a better interface between machines and humans.
- Researchers in the field of human-computer interaction (HCI) study how people engage with computers and develop technologies that allow people to connect with computers in new ways.

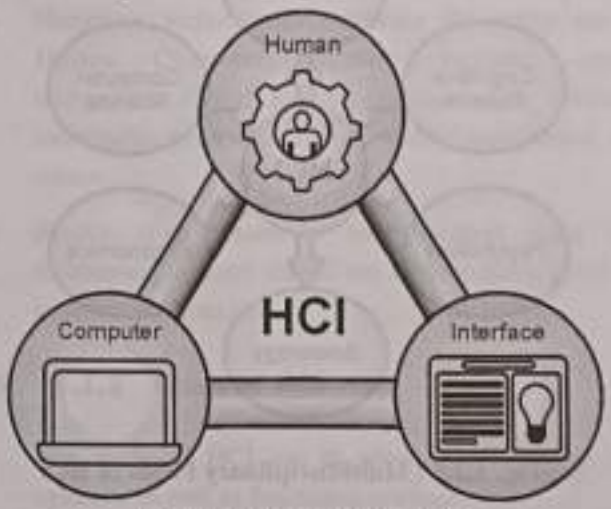
(1A1) Fig. 1.1.1 : Human-Computer Interaction

1.1.1 Human or User

- We may use the term "user" to refer to a particular user or a group of users who collaborate. It's critical to understand how people's sensory systems (sight, hearing, and touch) communicate information.
- Furthermore, various users build distinct conceptions or mental models regarding their interactions, and they absorb and retain information in different ways.

1.1.2 Computer

- When we talk about the computer, we are referring to any technology ranging from desktop computers, to large scale computer systems.
- For example, if we were discussing the design of a Website, then the Website itself would be referred to as "the computer".



(1A2) Fig. 1.1.2 : HCI Interface

1.1.3 Interaction

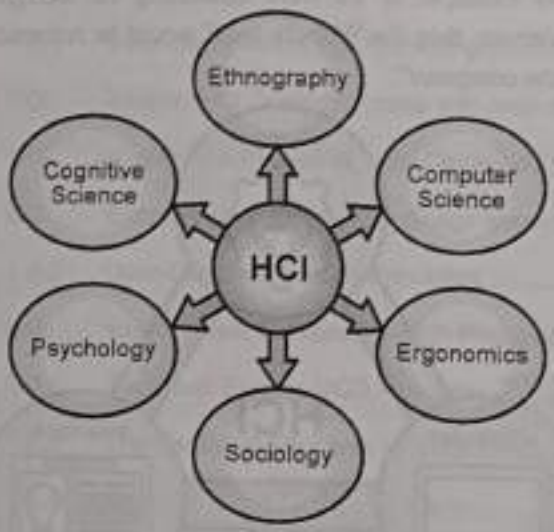
- There are obvious differences between humans and machines. In spite of these, HCI attempts to ensure that they both get on with each other and interact successfully.
- In order to achieve a usable system, you need to apply what you know about humans and computers, and consult with likely users throughout the design process.
- In real-world systems, the schedule and the budget are important, and it is vital to find a balance between what would be ideal for the users and what is feasible in reality.

1.2 DISCIPLINES INVOLVED IN HCI

GQ. What are the different disciplines involved in HCI. (6 Marks)

- HCI has grown to include a variety of fields, including computer science, cognitive science, and human-factors engineering, since its inception.

The following disciplines have made significant contributions to the development of human-computer interaction :



(1A3)**Fig. 1.2.1 : Multidisciplinary Fields of HCI**

Psychology

- Understanding, describing, modeling, predicting and explaining human behavior.

Cognitive Science

- Information processing
- Performance prediction
- Cooperative working and capabilities

Computer Science

- Including graphics, technology, prototyping tools, user interface management systems
- Construction and programming of computers
- Programming language design
- Architectural/algorithmic design of programs

Ergonomics

- Information presentation
- Language design
- Training

Sociology

- Groupware
- Social and organizational structures

Ethnography

- Studying Human behavior

Other Disciplinary contributed to HCI

- Linguistics:
- Including natural language interfaces

Engineering and design

- Engineering principles
- Graphic design

Artificial intelligence

- Intelligent software

Human factors

- Display readability
- Hardware design

Art

- Aesthetic appeal

1.3 WHY HCI STUDY IS IMPORTANT ?

- Today, Human Computer Interface is integrated into our lives than it has ever been. Moreover, it can be found as one of the most important considerations in every aspect of our lives.
- Every day, Human interacts with several technologies having a basic application in their life. Sometimes, it becomes difficult for humans to understand the technology or device. So interaction becomes important at this point.
- HCI helps to make interfaces that **increase productivity**, **enhance user experience**, and **reduce risks** in safety-critical systems.

- Poorly designed machines lead to many unexpected problems. This is why HCI is on the rise. As we become more dependent on technologies, even just the Internet or Smartphone, HCI has become a key part of designing tools that can be used efficiently and safely on a daily basis.

1.3.1 Importance of the Human Computer Interface

GQ. Explain Importance of HCI. (5 Marks)

- HCI is Very important when designing systems which will be usable for people with a varied range of abilities and expertise, and who have not complete knowledge of that.
- HCI takes advantage of our everyday knowledge of the world to make software and devices more understandable and usable for everyone.
- For example, using a graphic of a miniature folder in a computer's interface helps the user understand the purpose of the folder, because everyone is familiar with real paper folders.

(1A4)Fig. 1.3.1 : Design Interaction

- The term "interaction" or "interface" refers to an abstract model of human-machine interaction.
- User interface is an important element of any software. It is responsible for attracting the user to the application.
- Through HCI, one can obtain a better understanding of user experience and how to improve computer equipment and user applications.
- The Human Computer Interface (HCI) is a design that combines the services that the device must give in order to complete a task with a strong interaction between the device and the user.
- Human Computer Interface plays important role while designing several systems, software's or interfaces such as train ticketing system, ATM machines, banking software, aircraft, cars and management software.
- Numerous technologies involving the active use of Human Computer Interface included speech recognition, virtual reality, graphical user interface, multimedia presentation, handwriting recognition, and others.
- Finally, if a system is well-designed using HCI techniques, the user should not have to think about the system's complexities.

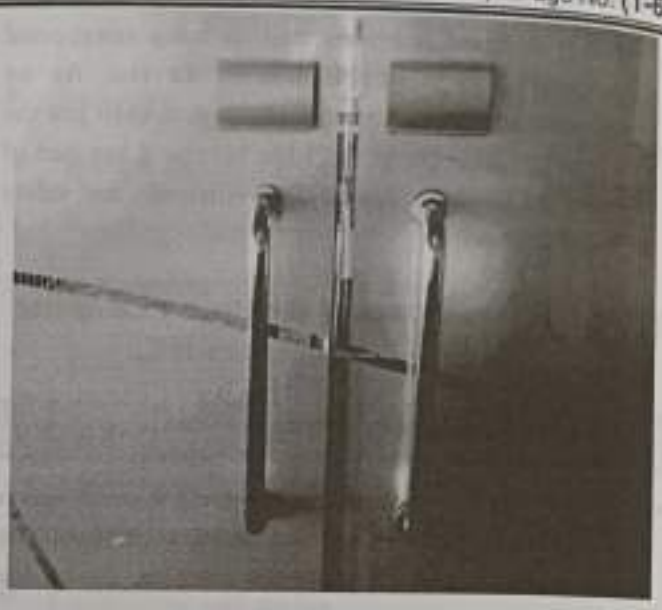
1.3.2 Goals of HCI

- The goals of HCI are to produce usable and safe systems, as well as functional systems.
- In order to produce computer systems with good usability, developers must attempt to:
  - Understand the factors that determine how people use technology, develop tools and techniques to enable building suitable systems, achieve efficient, effective, and safe interaction put people first.
  - Underlying the whole theme of HCI is the belief that people using a computer system should come first. Their needs, capabilities and preferences for conducting various tasks should direct developers in the way that they design systems.
  - People should not have to change the way that they use a system in order to fit in with it. Instead, the system should be designed to match their requirements
- HCI is a broad field that reaches almost every industry. It often overlaps with areas like user-centered design (UCD), user interface (UI) design, and user experience (UX) design.

1.4 THE PSYCHOLOGY OF EVERYDAY THINGS

"Psychology is the study of mental illness"

- Why this psycho title, though? Because design has a lot about psychology.
- People are diverse, you will probably never find two persons that are exactly the same.
- Human behavior is diverse, and people react in different ways. That's why design and user experience are so closely related to psychology.
- The design team have to think about how different people will behavior while interacting with some product, and even making sure that every possible error was taken into consideration.
- A good design is not the one that works pretty well in the expected scenario, but the one that can make the users feel comfortable when unexpected errors occur.
- The psychology of everyday things, emphasizes the understanding of everyday things, things with knobs and dials, controls and switches, lights and meters.
- The instances we have just examined demonstrate several principles, including the importance of visibility, appropriate clues, and feedback of one's actions.
- These principles constitute a form of psychology the psychology of how people interact with things.
- The Design of everyday things makes us crazy. Norman Door is the best example to explain this concept here.
- Imagine you are walking towards the door in the photo below. When you reach it, do you think you should push, or pull to open the door ?



(1A5) **Fig. 1.4.1 : Human-Computer Interaction with Norman Door**

- I am sure we have all encountered a door like this at some point, with a handle that looks like you should pull it, except you actually need to give it the push.
- It's not our fault as the 'user' that we have struggled with the door, here pulling only to find we should be pushing - the design of the door has failed to communicate how we should interact with it. These kinds of ambiguous, poorly designed doors - they are called "*Norman Doors*".
- How can such a simple thing as a door be so confusing? A door would seem to be about as simple a device as possible. There is not much you can do to a door: you can open it or shut it.
- Norman says, The design of the door should indicate how to work it without any need for signs, certainly without any need for trial and error.
- Two fundamentals that he brings to us are the main characteristics of a good design :
  - **Discoverability** : The product must be self explanatory, the user should be able to discover what actions are possible and how to perform them;
  - **Understanding** : Users must be able to understand how the product is supposed to be used and what all the features mean.

- According to the Norman fundamentals, the most relevant components of a product must be visible and communicate the correct message. For everyday products, of course, this is something really intrinsic, but for complex products sometimes an extra layer of information is necessary.
- A good example are instruction manual, or even in-person trainings, that can make the hole experience more complete and relevant. For Norman, this is an aspect that designers usually miss, while focusing on beauty instead of utility.
- Everything in the modern life is designed. When you read everything, you can really think about anything, and you will turn out understanding that it was designed somehow. That's why we have all these different kinds of designers within the design profession. Norman focus in three, specially :
  - **Industrial designers :** Emphasize the form and the material that a product takes;
  - **Interaction designers :** Focus on designing the understandability of a product and its usability;
  - **Experience designers :** Concerned about the whole emotional impact that a product takes in users' lives.

(1A8) **Fig. 1.4.2 : Norman Door Principle**

- With doors that push, the designer must provide signals that naturally indicate where to push. These need not destroy the aesthetics.

1.5 PRINCIPLES OF HCI

- Interaction design is the relationship between user and product and the services they use.
- The purpose of interaction design is to create a great user experience. Most of the UI disciplines require understanding and hands-on experience of interaction design principles.
- Interaction designers attempt to create meaningful relationships between people and the products and services they use. It may include computers, mobile devices, gadgets, appliances, and more.
- There are different HCI principles that HCI researchers use to evaluate an interface.
- These principles were developed by Ben Shneiderman , Don Norman and Jakob Nielsen.

1. Shneiderman's Eight Golden Rules (HCI Principles)

UQ. Explain Shneiderman's Eight Golden Rules of Interface Design. **(SPPU - Q.1(b), May 19, 4 Marks)**

UQ. Explain any 2 of the following HCI principles in brief. **(SPPU – Q.1(b), Dec. 17, Dec. 18, Q.1(a), Dec. 19, 5 Marks)**

(i) Know thy user

(ii) Understand the task

(iii) Reduce Memory Load

(iv) Strive for consistency

(v) Prevent Errors/ Reversal of Action

1. Know thy User or Client

Understanding people and what they do is a difficult and often undervalued process but very critical because of the gap in knowledge, skills, and attitudes existing between system users and developers that build the systems. To create a truly usable system, the designer must always do the following :

- Understand how people interact with computers.
- Understand the human characteristics important in design.
- Identify the user's level of knowledge and experience.
- Identify the characteristics of the user's needs, tasks, and jobs.
- Identify the user's psychological characteristics.
- Identify the user's physical characteristics.
- Employ recommended methods for gaining understanding of users.

2. Understanding the task

- Understanding the task at hand is closely related to the interaction modeling and user analysis.
- A method understanding the task users carry out with a product or system. To analyze the purpose of what people are doing? What are they trying to achieve? Why are they trying to achieve it, and how are they going about it?
- Task analysis is commonly used for breaking down task into subtasks and maintain the hierarchy of goals, operations and plans.

3. Strive for consistency

- Strive for consistency by utilizing familiar icons, colors, menu hierarchy, call-to-actions, and user flows when designing similar situations and sequence of actions.
- Standardizing the way information is conveyed ensures users are able to apply knowledge from one click to another; without the need to learn new representations for the same actions.
- Consistency plays an important role by helping users become familiar with the digital landscape of your product so they can achieve their goals more easily.

4. Enable frequent users to use shortcuts

- Enable frequent users to use shortcuts: With increased use comes the demand for quicker methods of completing tasks.
- For example, both Windows and Mac provide users with keyboard shortcuts for copying and pasting, so as the user becomes more experienced, they can navigate and operate the user interface more quickly and effortlessly.

5. Offer informative feedback

- Offer informative feedback : The user should know where they are at and what is going on at all times. For every action there should be appropriate, human-readable feedback within a reasonable amount of time.
- A good example of applying this would be to indicate to the user where they are at in the process when working through a multi-page questionnaire.
- A bad example we often see is when an error message shows an error-code instead of a human-readable and meaningful message.

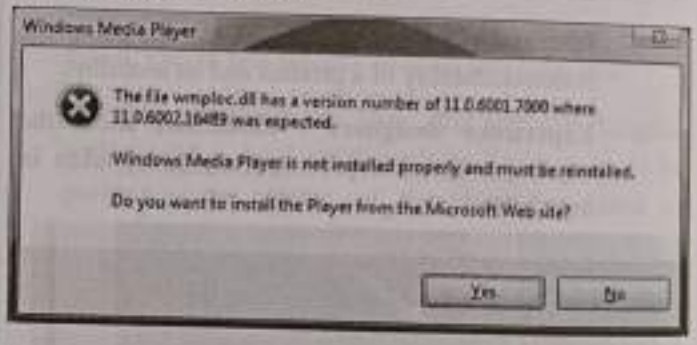


Fig. 1.5.1 : HCI informative feedback

- The Windows Media Player designers should have remembered Ben Shneiderman's golden rule: Offer informative feedback.
- Poorly designed error messages often show an error-code that does not mean anything to the user. As a good designer you should always seek to give human-readable and meaningful feedback.

6. Design dialogs to yield closure

- Design dialogue to yield closure: Don't keep your users guessing. Tell them what their action has led them to.
- For example, users would appreciate a "Thank You" message and a proof of purchase receipt when they've completed an online purchase.

7. Offer error prevention and simple error handling

- Offer simple error handling: No one likes to be told they're wrong, especially your users. Systems should be designed to be as fool-proof as possible, but when unavoidable errors occur, ensure users are provided with simple, intuitive step-by-step instructions to solve the problem as quickly and painlessly as possible.
- For example, flag the text fields where the users forgot to provide input in an online form.

8. Permit easy reversal of actions

- Permit easy reversal of actions: Designers should aim to offer users obvious ways to reverse their actions.
- These reversals should be permitted at various points whether it occurs after a single action, a data entry or a whole sequence of actions. This feature relieves anxiety, since the user knows that errors can be undone; it thus encourages exploration of unfamiliar options.

9. Support internal locus of control

- Support internal locus of control: Allow your users to be the initiators of actions. Give users the sense that they are in full control of events occurring in the digital space.
- Earn their trust as you design the system to behave as they expect.

10. Reduce short-term memory load

- Reduce short-term memory load: Human attention is limited and we are only capable of maintaining around five items in our short-term memory at one time. Therefore, interfaces should be as simple as possible with proper information hierarchy, and choosing recognition over recall.
- Recognizing something is always easier than recall because recognition involves perceiving cues that help us reach into our vast memory and allowing relevant information to surface. For example, we often find the format of multiple choice questions easier than short answer questions on a test because it only requires us to recognize the answer rather than recall it from our memory.

2. Norman's Seven Principles of HCI

UQ. What are the Norman's Seven Principles for Transforming Difficult Tasks into Simple Ones?

(SPPU - Q. 2(a), May 19, 6 Marks)

1. Use both knowledge in the world and knowledge in the head.
2. Simplify the structure of tasks.
3. Make things visible.
4. Get the mappings right.
5. Exploit the power of constraints, both natural and artificial.
6. Design for error.
7. When all else fails, standardize.

(1A7)**Fig. 1.5.2 : HCI Design Principles**

3. Norman's Fundamental Principles of Interaction

GQ. What are the Norman's Fundamental Principles of Interaction. **(10 Marks)**

Norman's seven fundamental design principles can help the user determine the answers to their questions; whether they are using an everyday thing or a product. In summary, here are the principles we observed :

- Discoverability makes it easier to understand where to perform actions
- Feedback communicates the response to our actions
- Conceptual models are a simple explanation of how something works

- Affordance is the perceived action of an object
- Signifiers tell us exactly where to perform an action
- Mapping is the relationship between the controls and effect they have
- Constraints help restrict the kind of interactions that can take place

☞ Discoverability

- Whenever we engage with an everyday thing such as a TV remote control, or a product like a website or an application, we figure out where and how to perform various functions.
- Through good discoverability, we can consider the different options and choose the one that should work to meet our goal. However, we cannot do this if the actions are not discoverable.
- Norman describes good discoverability as: "*it is possible to determine what actions are possible and the current state of the device*".

☞ Affordance

- Affordance is the relationship between what something looks like and how it's used.
- This is considered in terms of design. This indicates the manner in which an object is to be used. Also, humans should be virtually able to see the object. This denotes the things we perceive or think about often unknowingly.

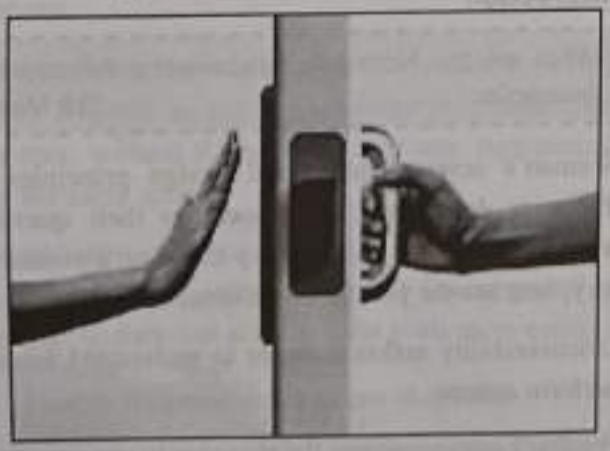


Fig. 1.5.3 : Affordance

- Perceived affordance is what the user understands by looking at the object (Donald Norman introduced).
- For designers, it means that as soon as someone sees something, they have to know how to use it. For example, a mug has high affordance: it's easy to figure out intuitively how to use it.
- For web designers, affordance is even more important. Users need to be able to tell how to access information they want from a website, or else they will just leave.

☞ Signifiers

- A sign's physical form (such as a sound, printed word, or image) as distinct from its meaning.

Fig. 1.5.4 : Signifiers

- Sometimes, affordances are not clear enough to tell the user the functionality of the object. A signifier removes all such ambiguity

Perceivability

- A user should be able to perceive the time or possibility of the action to be carried out.

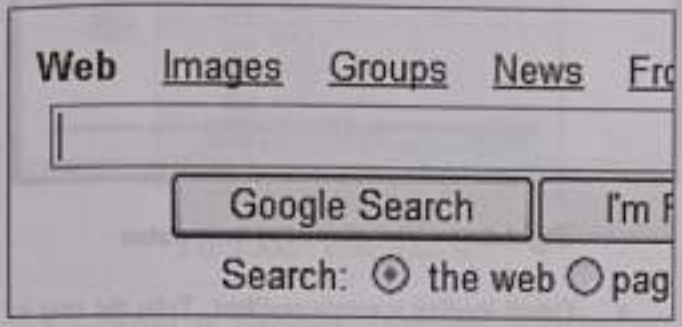


Fig. 1.5.5 : Perceivability

- The design interface should be in a manner to provide the users with information that follows.
- The users should be able to work out the interface without much difficulty. This denotes higher website perceivability.

Visibility

- Users need to know what all the options are, and know straight away how to access them.

Fig. 1.5.6 : Visibility

- For example, use intuitive iconography that clearly indicates there are more options hiding deeper down

Mapping

- Mapping is the relationship between control and effect.

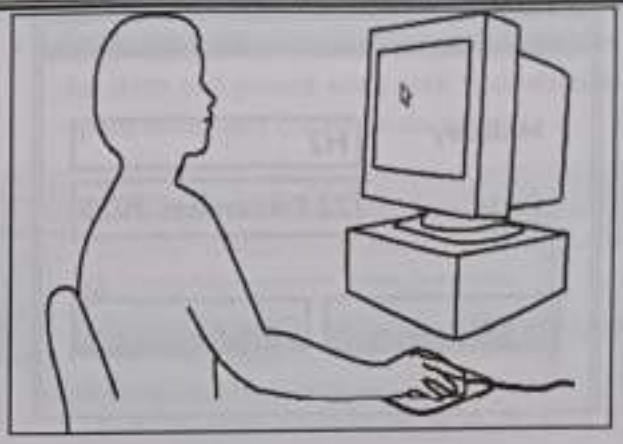


Fig. 1.5.7 : Mapping

- The idea is that with good design, the controls to something will closely resemble what they affect.
- A great example of mapping is the vertical scroll bar. It tells you where you are in a page, and as you drag it down, the page moves down at the same rate; control and effect are closely mapped.

Feedback

- Every action needs a reaction.

Fig. 1.5.8 : Feedback

- There needs to be some indication, like a sound, a moving dial, a spinning rainbow wheel, that the user's action caused something.
- Google Chrome does a great job of this when they're loading pages. The little spinning circle starts as soon as you hit enter, so you know something's happening, and goes faster when the page is about to load, so you know you're about to do something again. It's simple and effective feedback.

Constraints

- Constraints are the limits to an interaction or an interface. Some are really obvious and physical.

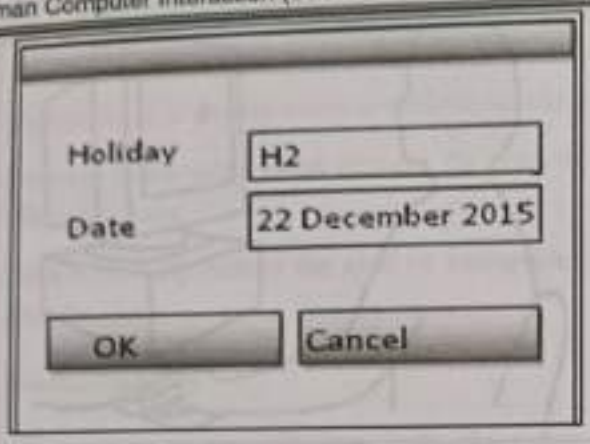


Fig. 1.5.9 : Constraints

- For example the screen size on a phone.

☞ **Consistency**

- The same action has to cause the same reaction, every time.

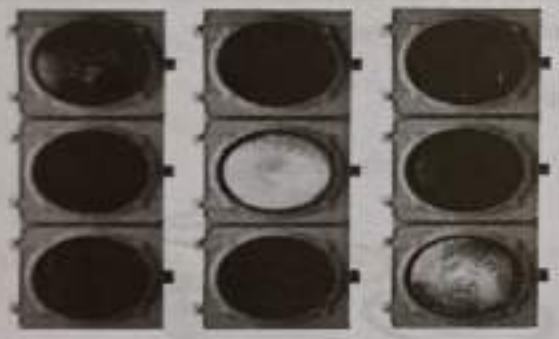


Fig. 1.5.10 : Consistency

- If a website has a back button that sometimes turns a computer off, it becomes very hard to navigate around the web. The same applies for visual consistency.

4. Nielsen's Ten Usability Heuristics

UQ. Explain Nielsen's Ten Heuristics.
(SPPU - Q. 7(b), Dec. 18 , 8 Marks)

1. Visibility of system status

- The system should always keep users informed about current state and actions through appropriate visual cues and feedback within reasonable time.

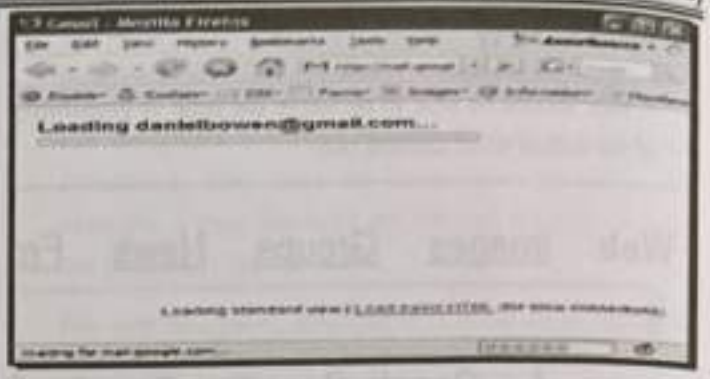


Fig. 1.5.11 : Visibility of system status

- Gmail loading a user's mailbox. Tells the user to wait and Indicates the status of what's going on.

2. Match between system and the real world

- The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms.
- Follow real-world conventions, making information appear in a natural and logical order.

Fig. 1.5.12 : System and real world interface

iBook's, iPad application using the metaphor of wooden book shelf.

3. User control and freedom

- Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

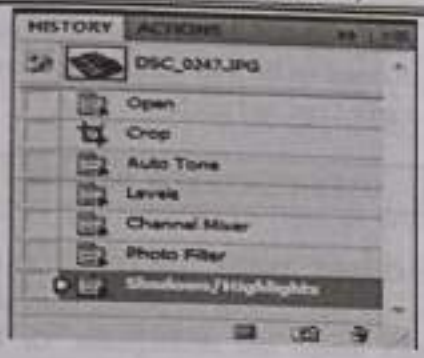


Fig. 1.5.13 : User control

- History in Photoshop helps user in recovering previous steps.

4. Consistency and standards

- Users should not have to wonder whether different words, situations, or actions mean the same thing.
- Follow platform conventions.

Fig. 1.5.14 : Inconsistent Icons

5. Help users recognize, diagnose and recover from errors

- Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

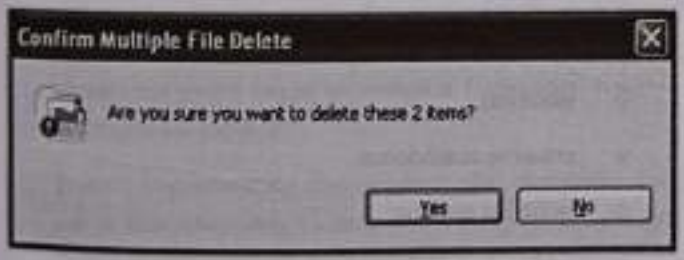


Fig. 1.5.15 : Error Message

6. Error prevention

- Even better than good error messages is a careful design which prevents a problem from occurring in the first place.
- Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Fig. 1.5.16 : Error Prevention

7. Recognition rather than recall

- Minimize the user's memory load by making objects, actions, and options visible.
- The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate

Fig. 1.5.17 : Recognition

8. Flexibility and efficiency of use

- Accelerators –unseen by the novice user --may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users.
- Allow users to tailor frequent actions.

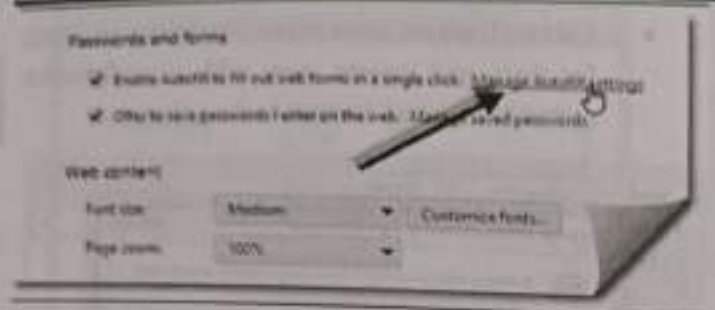


Fig. 1.5.18 : Efficiency of use

9. Aesthetic and Minimalist Design

- Dialogues should not contain information which is irrelevant or rarely needed.
- Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

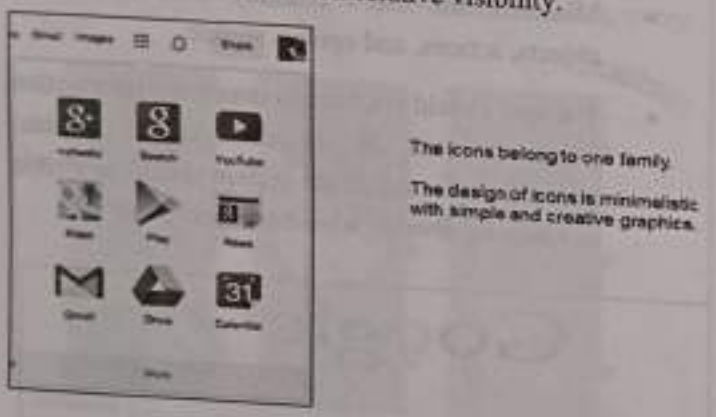


Fig. 1.5.19 : Dialogue design

10. Help and documentation

- Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation.

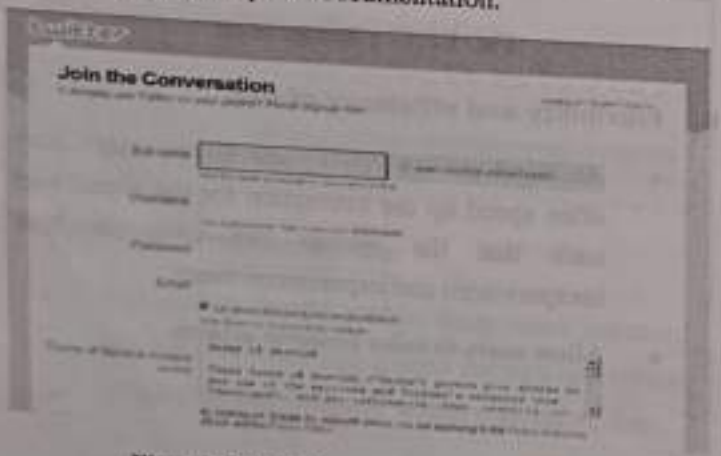
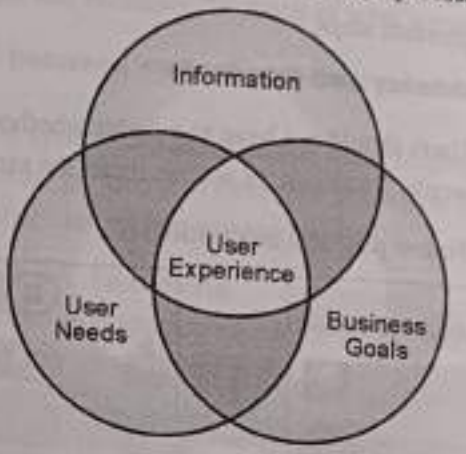


Fig. 1.5.20 : Help and documentation

- Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

1.6 USER-CENTERED DESIGN

- User-Centred Design (UCD) is a collection of processes which focus on putting users at the center of product design and development. Or User-Centered Design (UCD) is an iterative design process in which designers and other stakeholders focus on the users and their needs in each phase of the design process.



(1A8)Fig. 1.6.1 : User-Centered Design

- Dr. Donald Norman, a cognitive science researcher was the first to explain the importance of user-centered design. He said that design decisions should be based on the needs and wants of users. The value system of user-centered design contains :
  - empathy
  - optimism,
  - iteration,
  - creative confidence,
  - belief in making,
  - embracing ambiguity,
  - learning from failure.
- User-centered design creates lots of ideas and make innovative new products rooted in people's actual needs.

- To design in a user-centric way identify the people who will use the product, what they'll use it for, and the conditions under which they will use it. Observe people's lives, hear their hopes and needs, and get smart about your challenge.
- User-centered design provides a common language for scientists, stakeholders and end users. For example, Lunar Rover Mission of NASA integrated user-centered design techniques. NASA also benefits from user interviews, user observations in context, and wireframing.

1.6.1 User-centered Design Process goes through following Phases

UQ. Explain HCI Design Process with neat diagram. **(SPPU - Q. 6(a), Dec. 19, 8 Marks)**

GQ. Explain the Phases of User Centered Design. **(8 Marks)**

UQ. Express your opinion – "A design should be User-Centric" **(SPPU - Q. 1(a), Dec. 18, 8 Marks)**

Specify the use context and user's needs

- Before beginning to develop a product, the designer must research the ideal user and their needs.
- By observing their lives, the designers are able to get a broad picture of some of the challenges these users face.

Specify business requirements

- After interviewing users, the designers have a better sense of what is most desirable to them. During this phase, designers begin to research financially feasible solutions for the user.
- Before implementing the product, the designers may ask a few questions to help them modify the design. Some of the questions may include :
  - What partnerships need to be made?
  - What resources do we need to help develop this project?
  - What is our revenue stream like?

Build design solutions from rough concept to finished design

- This stage of the UCD process involves generating ideas, testing and refining solutions based on user requirement and input. When working through this stage, it is important to keep users involved so that the product can be continuously changed to meet their needs.
- Testing out ideas on users helps designers discover which feasible ideas help the user and which do not. Since the user's needs change over time it is important to keep testing ideas to ensure that they are still relevant solutions.
- This phase will continue until the designers and users are happy with the result.

Evaluate designs with usability testing

- At this point in the UCD process, designers conduct usability testing with actual users of their product. This stage gives the designers insight into how the users would actually interact with the product and understand how to tweak it to better suit them.
- It is recommended that this stage is conducted as soon as possible. The faster that feedback is received from users, the faster the designers can understand their product from the user's point of view.

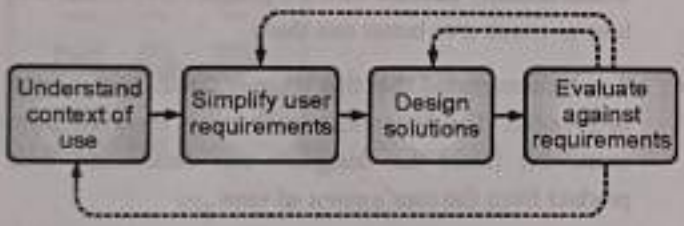
Implementation - develop and deliver the product

- The implementation phase allows designers to finally bring their solution to the market. Within this phase designers focus on a few key steps:
  - Building partnerships
  - Altering their business model
  - Piloting their idea
  - Receiving a reliable solution
- The best way to do this is by researching the user and launching a prototype within their natural setting. This step will look very different depending on the type of product that is being designed.

- For example, the implementation phase for a product that is designed for a younger user may involve user interviews, observing the user in their personal life and community settings.
- By observing the user, the designer is able to put themselves in their shoes. Once this stage is implemented, the design is likely to change to best fit the user's needs and meet business goals.

Deployment - the final product is evaluated, as consumer needs change

- The final stage of designing, before releasing a product, aims to understand if the product will have the desired impact on the user. Assessing this factor will differ depending on the goal of the product.
- If the product is meant to alter user behavior, factors such as user feedback will greatly help the designer understand the outcome of their work.
- Alternatively, the assessment may be as simple as understanding whether or not the product generated revenue.



(1A9)**Fig. 1.6.2 : Phases of User-Centered Design**

1.6.2 User-Centered Design Principles

UQ. Explain User Centered Design Principles?
(SPPU - Q. 1(a), May 19, 6 Marks)

UQ. State and Explain UCD Principles.
(SPPU - Q.1(b), Dec. 19, 5 Marks)

- User-Centered Design (UCD) is a user interface design process that focuses on usability goals, user characteristics, environment, tasks, and workflow in the design of an interface.
- UCD follows a series of well-defined methods and techniques for analysis, design, and evaluation of mainstream hardware, software, and web interfaces.
- The UCD process is an iterative process, where design and evaluation steps are built in from the first stage of projects, through implementation.
- User-centered design is based on a few fundamental principles that can be applied for the product design process:
- Users are involved in the design process from the very beginning. Critical design decisions are evaluated based on how they work for end-users.
- Importance of requirement clarification. The product team always tries to align business requirements with user's needs.
- Introducing user feedback loop in the product life cycle. The product team collects and analyzes feedback from users regularly. This information helps the team to make more user-focused decisions.
- Iterative design process. The product team constantly works on improving user experience; it introduces changes gradually as it gains more understanding about their target audience.
- It is quite simple – if you change the design late in the process, then it will typically cost ten times more than if you changed it during the requirements stage. Analysis and feedback are critical. User-centered design makes sure that you design and develop products right from the beginning, exactly what your clients want.

1.6.3 The Essential Elements of User-Centered Design

- **Visibility :** Users should be able to see from the beginning what they can do with the product, what is it about, how they can use it.
- **Accessibility :** Users should be able to find information easily and quickly. They should be offered various ways to find information for example call to action buttons, search option, menu, etc.

- **Legibility :** Text should be easy to read. As simple as that.
- **Language :** Short sentences are preferred here. The easier the phrase and the words, the better.
- **User-centered design and User Experience (UX) :** User-centered design improves the user experience.

  While it can be applied to almost any product, in this article, we will focus on website or mobile app development. It helps to understand users' needs and preferences regarding features of a product, task, goals, user flows, etc. At the end of the day, it has become one of the most important user experience requirements – that of being user-centered. It should be implemented throughout the entire customer experience, no guessing, no personal opinion. What matters is what your users say and do. Every "touchpoint" that the customer has with the product should be analysed, well design and developed.

1.6.4 User-Centered Design Example

(1A10)**Fig. 1.6.3 : Example of a UCD**

- A great example of a UCD website is Carters.com - a website to shop for children's clothing.
- On the site, navigation helps the user promptly reach the desired section by specifying a child's age (for example, Shoes: New Born - 3 year).
- At the same time, this navigation helps new customers by quickly directing them to the desired section.

1.7 MEASURABLE HUMAN FACTORS IN HCI

Five measurable human factors central to UI evaluation in HCI :

1. Time to learn
2. Speed of performance (define and time benchmark tasks)
3. Rate of user errors
4. Retention of knowledge over time
5. Subjective satisfaction

Importance of each is determined by nature of system, e.g.

1. Life-critical systems
2. Industrial and commercial systems
3. Office, home, and entertainment applications
4. Exploratory, creative (support creative activity), and cooperative systems.

Chapter Ends...

❑❑❑

UNIT II

CHAPTER 2

Understanding the Human and Human Interaction

Syllabus

Input-output channels, Human memory, Human emotions, Individual differences, Psychology.

Ergonomics, Human errors, Models of interaction, Paradigms of Interactions, Interaction styles, Interactivity, Context of interaction, User experience.

2.1 INPUT-OUTPUT CHANNELS

GQ. What are the Input-Output Channels in HCI ?

(8 Marks)

- A user's interaction with the outside world occurs through information being received and sent: input and output. In an interaction with a computer the user receives information that is output by the computer, and responds by providing input to the computer.
- For example, sight may be used primarily in receiving information from the computer, but it can also be used to provide information to the computer, for example by fixating on a particular screen point when using an eye tracking system. Input in the human occurs mainly through the senses and output through the motor control of the effectors.
- There are five major senses: sight, hearing, touch, taste and smell. Of these, the first three are the most important to HCI. Taste and smell do not currently play a significant role in HCI, and it is not clear whether they could be exploited at all in general computer systems, although they could have a role to play in more specialized systems (smells to give warning of malfunction, for example) or in augmented reality systems vision, hearing and touch are central.
- There are a number of effectors, including the limbs, fingers, eyes, head and vocal system. In the interaction with the computer, the fingers play the primary role, through typing or mouse control, with some use of voice, and eye, head and body position.
- Imagine using a personal computer (PC) with a mouse and a keyboard. The application you are using has a graphical interface, with menus, icons and windows. In your interaction with this system you receive information primarily by sight, from what appears on the screen.

2.1.1 Vision

- Human vision is a highly complex activity with a range of physical and perceptual limitations, we can roughly divide visual perception into two stages: the physical reception of the stimulus from the outside world, and the processing and interpretation of that stimulus.
- On the one hand the physical properties of the eye and the visual system mean that there are certain things that cannot be seen by the human; on the other the interpretative capabilities of visual processing allow images to be constructed from incomplete information.
- We need to understand both stages as both influence what can and cannot be perceived visually by a human being, which in turn directly affects the way that we design computer systems.
- We will begin by looking at the eye as a physical receptor, and then go on to consider the processing involved in basic vision.

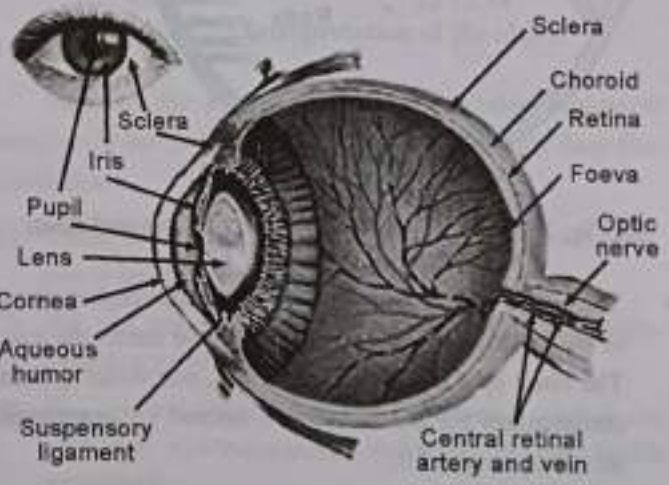
Unit II In Sem.

2.1.2 Human Eye

UQ. The human eye has number of limitations. Give three examples. For one of the limitation identified, describe how this should be taken into an account in the design of indivisible interface.

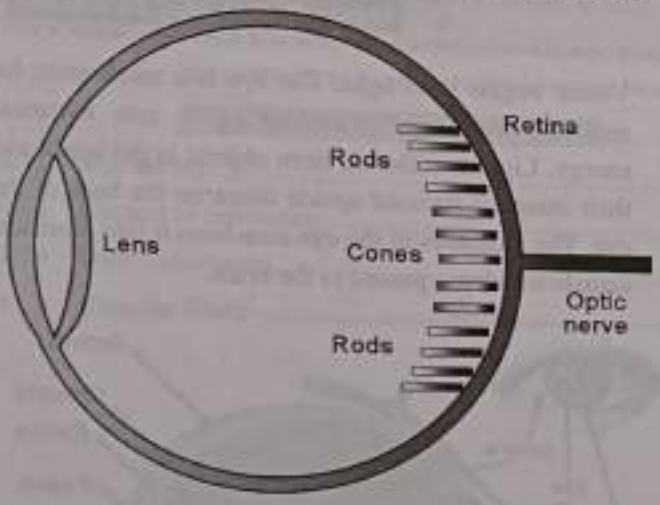
SPPU - Q. 3(a), May 19, 6 Marks

- Vision begins with light. The eye is a mechanism for receiving light and transforming it into electrical energy. Light is reflected from objects in the world and their image is focused upside down on the back of the eye. The receptors in the eye transform it into electrical signals which are passed to the brain.



(181) Fig. 2.1.1 : The Human Eye

- The eye has a number of important components. The cornea and lens at the front of the eye focus the light into a sharp image on the back of the eye, the retina.
- The retina is light sensitive and contains two types of photoreceptor: Rods and Cones. Rods are highly sensitive to light and therefore allow us to see under a low level of illumination. They are unable to resolve fine detail and are subject to light saturation.
- This is the reason for the temporary blindness we get when moving from a darkened room into sunlight: the rods have been active and are saturated by the sudden light. The cones do not operate either as they are suppressed by the rods.
- Cones are the second type of receptor in the eye. They are less sensitive to light than the rods and can therefore tolerate more light. There are three types of cone, each sensitive to a different wavelength of light.
- This allows color vision. The eye has approximately 6 million cones, mainly concentrated on the fovea, a small area of the retina on which images are fixated.

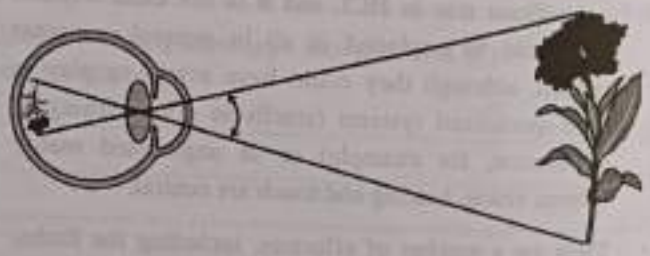


(182)**Fig. 2.1.2 : The Human Eye with Cones and Rods**

- The retina is mainly covered with photoreceptors there is one blind spot where the optic nerve enters the eye. The blind spot has no rods or cones, our visual system compensates for this so that in normal circumstances we are unaware of it.
- The retina also has specialized nerve cells called ganglion cells. There are two types:
- X-cells, which are concentrated in the fovea and are responsible for the early detection of pattern; and Y-cells which are more widely distributed in the retina and are responsible for the early detection of movement.
- The distribution of these cells means that, while we may not be able to detect changes in pattern in peripheral vision, we can perceive movement.

2.1.3 Visual Perception

- How does the eye perceive size, depth and relative distances? To understand this we must consider how the image appears on the retina.
- Reflected light from the object forms an upside-down image on the retina. The size of that image is specified as a visual angle. Fig. 2.1.3 illustrates how the visual angle is calculated.



(183)**Fig. 2.1.3 : Visual Angle**

- If drawing a line from the top of the object to a central point on the front of the eye and a second line from the bottom of the object to the same point, the visual angle of the object is the angle between these two lines.
- Visual angle is affected by both the size of the object and its distance from the eye. Therefore if two objects are at the same distance, the larger one will have the larger visual angle.
- Similarly, if two objects of the same size are placed at different distances from the eye, the furthest one will have the smaller visual angle. The visual angle indicates how much of the field of view is taken by the object.

- The visual angle measurement is given in either degrees or minutes of arc, where 1 degree is equivalent to 60 minutes of arc, and 1 minute of arc to 60 seconds of arc.

☞ Perceiving brightness

- An aspect of visual perception is the perception of brightness. Brightness is in fact a subjective reaction to levels of light. It is affected by luminance which is the amount of light emitted by an object.
- The luminance of an object is dependent on the amount of light falling on the object's surface and its reflective properties. Luminance is a physical characteristic and can be measured using a photometer.
- Contrast is related to luminance: it is a function of the luminance of an object and the luminance of its background.

☞ Perceiving Colour

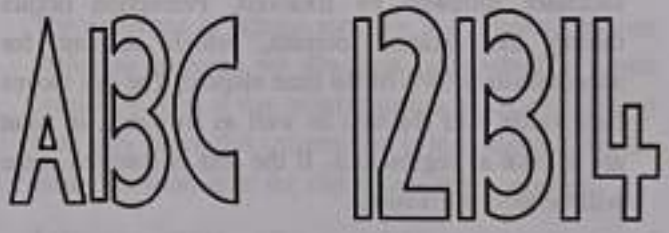
- A factor, that need to consider is perception of colour.
- Colour is usually made up of three components: hue, intensity and saturation.
- Hue is determined by the spectral wavelength of the light. Blues have short wavelengths, greens medium and reds long.
- Approximately 150 different hues can be discriminated by the average person. Intensity is the brightness of the color, and saturation is the amount of whiteness in the color.
- By varying these two, we can perceive in the region of 7 million different colors.

☞ The capabilities and limitations of visual processing

- Visual processing involves the transformation and interpretation of a complete image, from the light that is thrown onto the retina.
- Visual processing compensates for the movement of the image on the retina which occurs as we move around and as the object which we seemoves. Although the retinal image is moving, the image that we perceive is stable.
- Similarly, colour and brightness of objects are perceived as constant, in spite of changes in luminance. This ability to interpret and exploit our expectations can be used to resolve ambiguity.
- For example, consider the image shown in Fig. 2.1.4 is an ambiguous shape.

(184)**Fig. 2.1.4 : Ambiguous shape of object**

- In the Fig. 2.1.5 below. The context in which the object appears allows our expectations to clearly disambiguate the interpretation of the object, as either a B or a 13.



ABC **12 13 14**

(185)**Fig. 2.1.5 : Interpretation of the object**

Although the human eye is a remarkable biological invention, it has some shortcomings that ultimately limit that exploration :

(1) Colour blindness, blind spot, visual acuity (resolution), overly sensitive to motion, etc.

(2) The eye has limited size and therefore limited light-gathering power.

(3) The eye has limited frequency response, since it can only see electromagnetic radiation in the visible wavelengths.

(4) The eye distinguishes a new image multiple times a second, so it cannot be used to accumulate light over a long period in order to intensify a faint image.

Unit II In Sem.

(5) The eye cannot store an image for future reference like a photgraphic plate can.

Address the problem that each limitation may bring, as well as what needs to be considered in the interface. Because a large percentage of the population cannot distinguish between red and green, don't utilise colour just as a factor on buttons or displays, and be aware that red and green next to each other may be interpreted as the same by some, influencing visualisations and so on.

2.1.4 Reading

GQ. Explain Importance of reading in HCI. **(5 Marks)**

- There are several stages in the reading process. First, the visual pattern of the word on the page is perceived. It is then decoded with reference to an internal representation of language. The final stages of language processing include syntactic and semantic analysis and operate on phrases or sentences.
- During reading, the eye makes jerky movements called saccades followed by fixations. Perception occurs during the fixation periods, which account for approximately 94% of the time elapsed. The eye moves backwards over the text as well as forwards, in what are known as regressions. If the text is complex there will be more regressions.
- Adults read approximately 250 words a minute. It is unlikely that words are scanned serially, character by character, since experiments have shown that words can be recognized as quickly as single characters. Instead, familiar words are recognized using word shape. This means that removing the word shape clues (for example, by capitalizing words) is detrimental to reading speed and accuracy
- The speed at which text can be read is a measure of its legibility. Experiments have shown that standard font sizes of 9 to 12 points are equally legible, given proportional spacing between lines . Similarly line lengths of between 2.3 and 5.2 inches (58 and 132 mm) are equally legible.
- However, there is evidence that reading from a computer screen is slower than from a book.
- This is thought to be due to a number offactors including a longer line length, fewer words to a page, orientation and the familiarity of the medium of the page.
- These factors can of course be reduced by careful design of textual interfaces. a negative contrast (dark, characters on a light screen) provides higher luminance and, therefore, increased acuity, than a positive contrast. This will in turn increase legibility.
- Experimental evidence suggests that in practice negative contrast displays are preferred and result in more accurate performance.

2.1.5 Hearing

- The sense of hearing is often considered secondary to sight, but we tend to underestimate the amount of information that we receive through our ears. hearing begins with vibrations in the air or sound waves.
- The ear receives these vibrations and transmits them, through various stages, to the auditory nerves. The ear comprises three sections, commonly known as the Outer ear, middle ear and inner ear.

2.1.6 Touch

- Touch provides us with vital information about our environment. It tells us when we touch something hot or cold, and can therefore act as a warning.
- It also provides us with feedback when we attempt to lift an object, for example. Consider the act of picking up a glass of water. If we could only see the glass and not feel when our hand made contact with it or feel its shape, the speed and accuracy of the action would be reduced.
- This is the experience of users of certain virtual reality games: they can see the computer-generated objects which they need to manipulate but they have no physical sensation of touching them.
- Watching such users can be an informative and amusing experience! Touch is therefore an important means of feedback, and this is no less so in using computer systems.

2.1.7 Movement

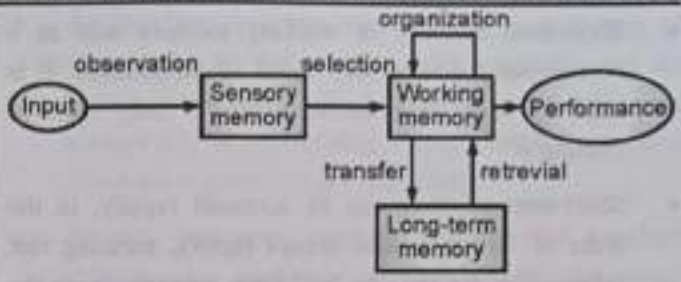
- A simple action such as hitting a button in response to a question involves a number of processing stages. The stimulus (of the question) is received through the sensory receptors and transmitted to the brain.
- The question is processed and a valid response generated. The brain then tells the appropriate muscles to respond. Each of these stages takes time, which can be roughly divided into reaction time and movement time.
- Movement time is dependent largely on the physical characteristics of the subjects: their age and fitness, for example. Reaction time varies according to the sensory channel through which the stimulus is received.
- A person can react to an auditory signal in approximately 150 ms, to a visual signal in 200 ms and to pain in 700 ms.

2.2 HUMAN MEMORY

GQ. Write a Short Note on Human Memory.

Ans. :

- Our memory contains our knowledge of actions or procedures. It allows us to repeat actions, to use language, and to use new information received via our senses.
- It also gives us our sense of identity, by preserving information from our past experiences Memory is the second part of our model of the human as an information-processing system.
- Memory is associated with each level of processing. Bearing this in mind, we will consider the way in which memory is structured and the activities that take place within the system.
- It is generally agreed that there are three types of memory or memory function: sensory buffers, short-term memory or working memory, and long-term memory. These memories interact, with information being processed and passed between memory stores.



(188)Fig. 2.2.1 : The structure of Human Memory

2.2.2 Sensory Memory(ST)

UQ. Explain significant of sensory memory in interface design? SPPU - Q. 4(a), May 19, 5 Marks

- The sensory memories act as buffers for stimuli received through the senses. A sensory memory exists for each sensory channel or significant : iconic memory for visual stimuli, echoic memory for aural stimuli and haptic memory for touch.
- These memories are constantly overwritten by new information coming in on these channels.
- The existence of echoic memory is evidenced by our ability to ascertain the direction from which a sound originates. This is due to information being received by both ears. Since this information is received at different times, we must store the stimulus in the meantime.
- Echoic memory allows brief to play-back of information. Information is passed from sensory memory into short-term memory by attention, thereby filtering the stimuli to only those which are of interest at a given time.

2.2.3 Short-term Memory(STM)/Working Memory

UQ. Discuss RAM and Short- Term memory (STM) SPPU - Q. 4(a), May 19, 5 Marks

UQ. Human memory plays an important role in how well people deal with an interface. Describe differences between STM and LTM. SPPU - Q. 2(a), Dec. 18, 5 Marks

UQ. Design and explain an experiment to investigate the decay aspect of human short-term memory. SPPU - Q. 2(a), Dec. 17, 5 Marks

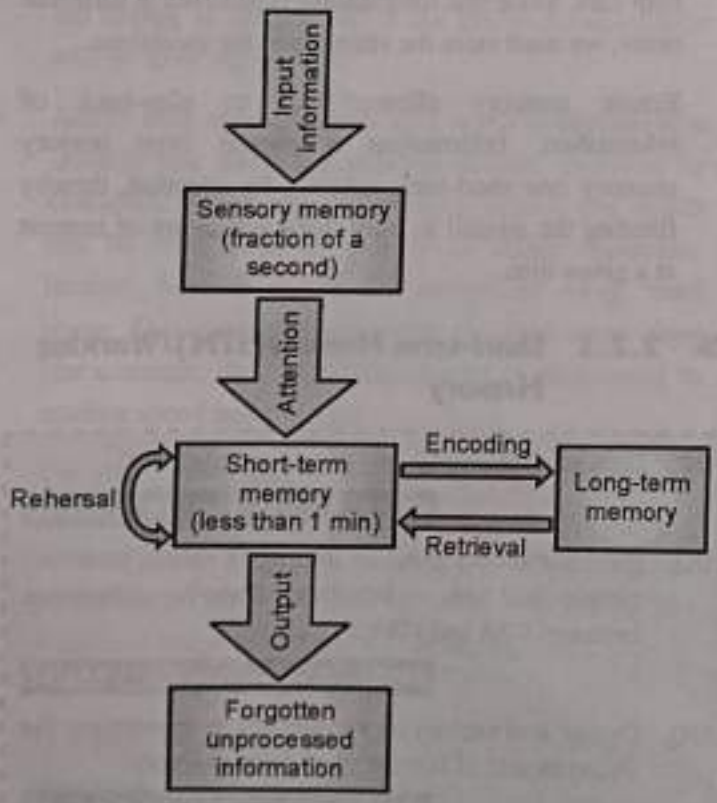
Unit II In Sem.

- Short-term memory or working memory acts as a scratch-pad for temporary recall of information. It is used to store information which is only required fleetingly.
- Short-term memory can be accessed rapidly, in the order of 70 ms. It also decays rapidly, meaning that information can only be held there temporarily, in the order of 200 ms.
- Short-term memory also has a limited capacity.
- There are two basic methods for measuring memory capacity. The first involves determining the length of a sequence which can be remembered in order. The second allows items to be freely recalled in any order.

2.2.4 Long-term memory(LTM)

GQ. Explain Short Term Memory and Long Term Memory in HCI. **(10 Marks)**

- If short-term memory is our working memory or scratch-pad, long-term memory is our main resource.
- Here we store factual information, experiential knowledge, and procedural rules of behaviour – in fact, everything that we know.



(187)**Fig. 2.2.2 : A model structure of Memory**

- It differs from short-term memory in a number of significant ways. First, it has a huge, if not unlimited, capacity. Secondly, it has a relatively slow access time of approximately a tenth of a second. Thirdly, forgetting occurs more slowly in long-term memory, if at all.
- Long-term memory is intended for the long-term storage of information. Information is placed there from working memory through rehearsal. Unlike working memory there is little decay: long-term recall after minutes is the same as that after hours or days.

Long-term memory structure

- There are two types of long-term memory: episodic memory and semantic memory.
- Episodic memory represents our memory of events and experiences in a serial form. It is from this memory that we can reconstruct the actual events that took place at a given point in our lives.
- Semantic memory, on the other hand, is a structured record of facts, concepts and skills that we have acquired. The information in semantic memory is derived from that in our episodic memory.

Long-term memory processes

- This process can be optimized in a number of ways.
- Ebbinghaus performed numerous experiments on memory, using himself as a subject. In these experiments he tested his ability to learn and repeat nonsense syllables, comparing his recall minutes, hours and days after the learning process.
- He discovered that the amount learned was directly proportional to the amount of time spent learning. This is known as the total time hypothesis.
- There are two main theories of forgetting: decay and interference. The first theory suggests that the information held in long-term memory may eventually be forgotten.
- Ebbinghaus concluded from his experiments with nonsense syllables that information in memory decayed logarithmically, that is that it was lost rapidly to begin with, and then more slowly.

- The second theory is that information is lost from memory through interference. If we acquire new information it causes the loss of old information. This is termed retroactive interference.
- A common example of this is the fact that if you change telephone numbers, learning your new number makes it more difficult to remember your old number. This is because the new association masks the old.
- However, sometimes the old memory trace breaks through and interferes with new information. This is called proactive inhibition. Forgetting is also affected by emotional factors.

2.3 HUMAN EMOTIONS

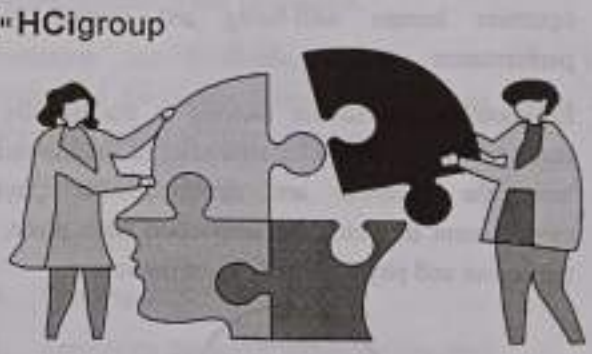
- Emotion is a fundamental component of being human. Joy, hate, anger, and pride, among the variety of other emotions, motivate action and add meaning and richness to virtually all human experience. Traditionally, human–computer interaction.
- (HCI) has been viewed as the "ultimate" exception; users must discard their emotional selves to work efficiently and rationality with computers or machines.
- Emotion plays an important role in our interactions with people and computers in everyday life. Emotions, some believe, are what make our interactions human.
- Studying emotion within the HCI discipline is an inherently interdisciplinary task. The specific areas of interest span recognition and synthesis of emotion in face and body, emotion sensors, speech specifics, and the influence of emotion on information processing and decision-making.

2.4 INDIVIDUAL DIFFERENCES

- Individual Differences among users have not been a major concern of commercial computer interface designers.
- Individual differences among users often have produced descriptive results rather than prescriptions for interface design.
- An interface designers should focus a great deal of attention on the differences among potential users for three reasons. First, individual differences usually play a major role in determining whether humans can use a computer to perform a job effectively.
- Second, personnel selection testing, the standard solution to problems of job-related individual differences, cannot be applied to many settings where humans interact with computers. The third reason for designers to be concerned with individual differences is that the technology has reached the point where it is possible to accommodate more user differences.
- User characteristics and user performance and preferences might be significantly related. User characteristics are classified into four groups: level of experience, personality traits, demographic characteristics and others. User characteristics that may affect user performance

Unit II In Sem.

2.5 PSYCHOLOGY



(188) **Fig. 2.5.1 : A figure of Psychology**

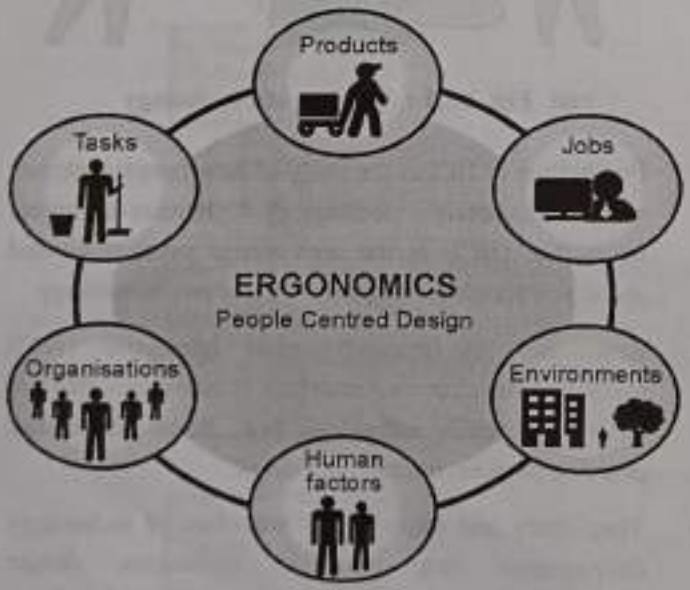
- Psychology in HCI is the study of how peoples interact with computing technology. Human–computer interaction (HCI) is the area where psychology and social science meet computer science and technology.
- Researchers in human-computer interaction (HCI) study and develop user-interface technologies that are tailored to each individual (e.g. three-dimensional pointing devices, interactive video).
- They study and improve the processes of technology development (e.g. usability evaluation, design rationale).

- They develop and evaluate new applications of technology (e.g. computer conferencing, software design environments).
- Through the past two decades, HCI has progressively integrated its scientific concerns with the engineering goal of improving the usability of computer systems and application thus establishing a body of technical knowledge and methodology.
- HCI continues to provide a challenging test domain for applying and developing psychology and social science in the context of technology development and use.

2.6 ERGONOMICS

GQ. Explain the term ergonomics in HCI. **(5 Marks)**

- Ergonomics (or human factors) is the scientific discipline concerned with the understanding of interactions among humans and other elements of a system, and the profession that applies theory, principles, data and methods to design in order to optimize human well-being and overall system performance
- Ergonomics (or human factors) is traditionally the study of the physical characteristics of the interaction: how the controls are designed, the physical environment in which the interaction takes place, and the layout and physical qualities of the screen.



(189) **Fig. 2.6.1 : Ergonomics in HCI.**

- A primary focus is on user performance and how the interface enhances or detracts from this. To evaluate the aspects of the interaction, ergonomics will certainly also touch upon human psychology and system constraints.
- There are three broad domains of ergonomics: physical, cognitive, and organizational. Physical ergonomics is concerned with human anatomical, anthropometric, physiological and biomechanical characteristics as they relate to physical activity.
- Cognitive ergonomics is concerned with mental processes, such as perception, memory, reasoning, and motor response, as they affect interactions among humans and other elements of a system.
- Organizational ergonomics is concerned with the optimization of sociotechnical systems, including their organizational structures, policies, and processes.

Benefits of Ergonomics

(1) Lower costs

(2) Higher productivity

(3) Better product quality

(4) Improved employee engagement

(5) Better safety culture

2.7 HUMAN ERRORS

- The error is any action that violates system, tolerance or some limit of system. Some user action may leads to error in one system but, May not leads to error in other system.
- The Human machine interface can be unsuccessful due to wrong user intervention and human errors. Human error is not limited only to system user. The user's behavioural patterns may lead to some incorrect activities.

Human Errors are :

(a) **Error of exclusion**

The action performed is not required to perform.

(b) **Error of instruction**

The action that should not be performed is actually performed.

Example : The faulty product is passed by quality management team whereas it should not be rejected by him.

(c) **Extraneous Act**

The action performed is to prevent the system goals.

Example : Using Mobile phone to locate unknown place

(d) **Sequential Error**

The action performed is not done in a proper sequence.

Example : First put car in first gear and then start the car.

(e) **Time Error**

The action performed is too much early then the permitted time of action.

Example : Parking a car after the parking time limit exceeds.

2.8 MODELS OF INTERACTION

GQ. What are the different models of Interaction in HCI ? **(8 Marks)**

- Interaction involves at least two participants: the user and the system.
- The interface must therefore effectively translate between them to allow the interaction to be successful. This translation can fail at a number of points and for a number of reasons.
- The use of models of interaction can help us to understand exactly what is going on in the interaction and identify the likely root of difficulties.
- They also provide us with a framework to compare different interaction styles and to consider interaction problems.

1) The terms of interaction

- The purpose of an interactive system is to aid a user in accomplishing goals from some application domain. A domain defines an area of expertise and knowledge in some real world activity.
- Some examples of domains are graphic design, authoring and process control in a factory. A domain consists of concepts that highlight its important aspects. In a graphic design domain, some of the important concepts are geometric shapes, a drawing surface and a drawing utensil.
- Tasks are operations to manipulate the concepts of a domain. A goal is the desired output from a performed task. For example, one task within the graphic design domain is the construction of a specific geometric shape with particular attributes on the drawing surface.
- A related goal would be to produce a solid red triangle centered on the canvas. An intention is a specific action required to meet the goal.

2) The execution-evaluation cycle

GQ. Explain Gulf of Evaluation and Gulf of Execution. **(8 Marks)**

- The interactive cycle can be divided into two major phases: Execution and Evaluation.
- Execution and Evaluation phases can then be subdivided into seven stages. These seven stages in Norman's model of interaction are as follows :

  1. Establishing the goal.

     Execution

  2. Forming the intention (plan)
  3. Specifying the action sequence (specify)
  4. Executing the action (perform)

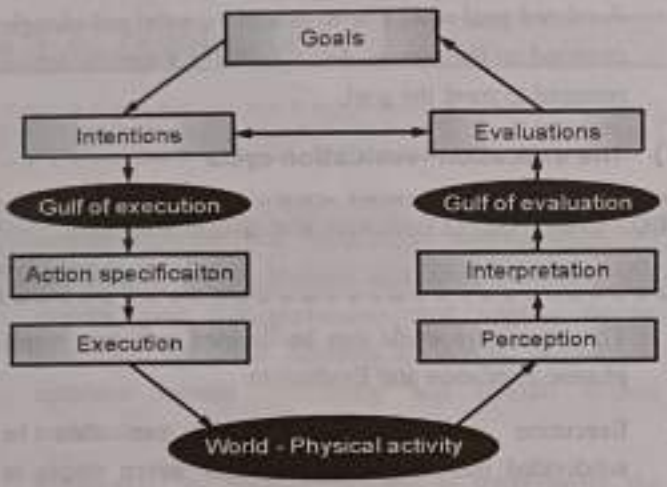
     Evaluation

  5. Perceiving the system state(perceive)
  6. Interpreting the system state (reflect)
  7. Evaluating the output with respect to the goals and intentions (compare)

- It is liable to be imprecise and therefore needs to be translated into the more specific intention, and the actual actions that will reach the goal, before it can be executed by the user.

Unit II In Sem.

- The user perceives the new state of the system, after execution of the action sequence, and interprets it in terms of his expectations.
- If the system state reflects the user's goal then the computer has done what he wanted and the interaction has been successful; otherwise the user must formulate a new goal and repeat the cycle.
- Norman uses this model of interaction to demonstrate why some interfaces cause problems to their users. He describes these in terms of the Gulfs of Execution and the Gulfs of Evaluation.



(1B10)**Fig. 2.8.1: Gulfs of Execution and the Gulfs of Evaluation**

- The Gulf of Execution is the difference between the user's formulation of the actions to reach the goal and the actions allowed by the system.
- If the actions allowed by the system correspond to those intended by the user, the interaction will be effective. The interface should therefore aim to reduce this gulf.
- The Gulf of Evaluation is the distance between the physical presentation of the system state and the expectation of the user.
- If the user can readily evaluate the presentation in terms of his goal, the gulf of evaluation is small.
- The more effort that is required on the part of the user to interpret the presentation, the less effective the interaction.

3) The Interaction Framework

UQ. There are four main translations involved in the interaction framework viz. articulation, performance, presentation and observation.

SPPU - Q 3(a), May 18, 5 Marks

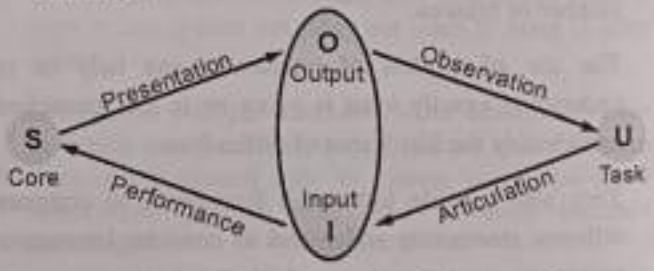
i) The compact disk player has a button for power off. However it's remote control does not have a power off button.

ii) It is difficult in a command line interface to determine the result of copying and moving files in a hierarchical file system.

iii) User is unable to figure out which switches from the bank to turn on to lit the front portion of a class room.

iv) The user is unable to know whether the voice recorder is in playing or recording state.

- Specify each of the above four cases which of the interaction framework translations are ineffective. The interaction framework attempts a more realistic description of interaction by including the system explicitly, and breaks it into four main components.
- The nodes represent the four major components in an interactive system – the System, the User, the Input and the Output. Each component has its own language.
- In addition to the User's task language and the System's core language, which we have already introduced, there are languages for both the Input and Output components. Input and Output together form the Interface.



(1B11)**Fig. 2.8.2 : Interaction framework**

- The general interaction framework Translations between components The System then transforms itself as described by the operations; the execution phase of the cycle is complete and the evaluation phase now begins.

- The System is in a new state, which must now be communicated to the User. The current values of system attributes are rendered as concepts or features of the Output.
- It is then up to the User to observe the Output and assess the results of the interaction relative to the original goal, ending the evaluation phase and, hence, the interactive cycle.
- There are four main translations involved in the interaction: articulation, performance, presentation and observation.

☞ Assessing overall interaction

- The interaction framework is presented as a means to judge the overall usability of an entire interactive system.
- This is not surprising since it is only in attempting to perform a particular task within some domain that we are able to determine if the tools we use are adequate.
- For a particular editing task, one can choose the text editor best suited for interaction relative to the task. The best editor, if we are forced to choose only one, is the one that best suits the tasks most frequently performed.

⏭ 2.9 PARADIGMS OF INTERACTION

GQ. What are the paradigms of Interaction in HCI? **(10 Marks)**

The paradigms of interaction are :

(1) Time sharing
(2) Video display units
(3) Programming toolkits
(4) Personal computing
(5) Window systems and the WIMP interface
(6) The metaphor
(7) Hypertext
(8) Computer-supported cooperative work
(9) The World Wide Web
(10) Ubiquitous computing

(1) Time sharing

- Major contributions to come out of this new emphasis in research were the concept of time sharing, in which a single computer could support multiple users.
- The human (or more accurately, the programmer) was restricted to batch sessions, in which complete jobs were submitted on punched cards or paper tape to an operator who would then run them individually on the computer.
- Time-sharing systems of the 1960s made programming a truly interactive venture and brought about a subculture of programmers known as hackers-single-minded masters of detail who took pleasure in understanding complexity.
- Though the purpose of the first interactive time-sharing systems was simply to augment the programming capabilities of the early hackers, it marked a significant stage in computer applications for human use.
- Rather than rely on a model of interaction as a pre-planned activity that resulted in a complete set of instructions being laid out for the computer to follow, truly interactive exchange between programmer and computer was possible.

(2) Video display units

- In mid-1950s researchers were experimenting with the possibility of presenting and manipulating information from a computer in the form of images on a video display unit (VDU).
- These display screens could provide a more suitable medium than a paper printout for presenting vast quantities of strategic information for rapid assimilation.
- The earliest applications of display screen images were developed in military applications, most notably the Semi-Automatic Ground Environment (SAGE) project of the US Air Force.

(3) Programming toolkits

- Programming toolkits provide a means for those with substantial computing skills to increase their productivity greatly.

Unit II In Sem.

- One of the first demonstrations that the powerful tools of the hacker could be made accessible to the computer novice was a graphics programming language for children called LOGO.
- A child could quite easily pretend they were inside the turtle and direct it to trace out simple geometric shapes, such as a square or a circle.
- By typing in English phrases, such as go forward or Turn left, the child/programmer could teach the turtle to draw more and more complicated figures. The system will always be more powerful if it is easier to use.

(4) Personal computing

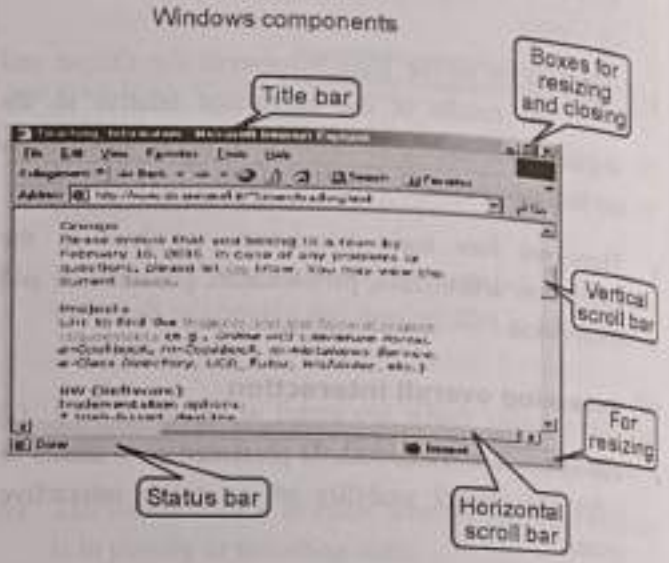
- Humans are able to think about more than one thing at a time, and in accomplishing some piece of work, they frequently interrupt their current train of thought to pursue some other related piece of work.

(1812)**Fig. 2.9.1 : Personal Computer System**

- A personal computer system which forces the user to progress in order through all of the tasks needed to achieve some objective, from beginning to end without any diversions, does not correspond to that standard working pattern.

(5) Window systems and the WIMP interface

- Windows, icons, menus and pointing device (WIMP) denotes a style of computer-human interaction involving the aforementioned elements of the graphical user interface (GUI) which is the most common interaction method being used by desktop computers today.
- It includes both Windows and Macintosh interfaces, as well as other less common operating systems, such as Linux and NeXT.



(1812)**Fig. 2.9.2 : WIMP Interface**

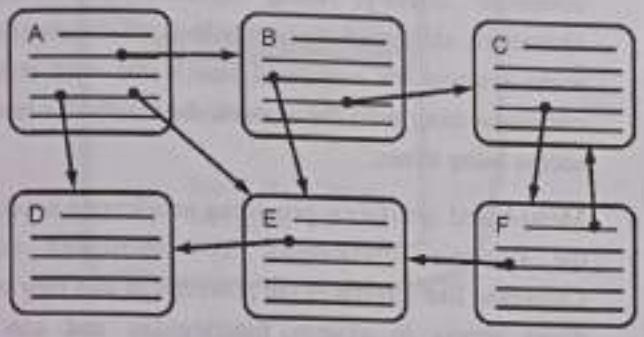
- While the terms GUI and WIMP are sometimes used interchangeably, WIMP is technically a subset of GUIs. This means all WIMP interfaces are GUIs, but not all GUIs are WIMPs.
- WIMP-based systems are designed to be used with a keyboard and mouse, since the mouse controls the pointer (or cursor) and the keyboard is used to enter data.
- Other GUIs may support different types of input, such as a touch screen display.

(6) The metaphor

- Paper used the metaphor of a turtle dragging its tail in the dirt. Children could quickly identify with the real-world phenomenon and that instant familiarity gave them an understanding of how they could create pictures.
- The danger of a metaphor is usually realized after the initial honeymoon period. When word processors were first introduced, they relied heavily on the typewriter metaphor.
- The keyboard of a computer closely resembles that of a standard typewriter, so it seems like a good metaphor from which to start.

(7) Hypertext

- Hypertext is text which is not constrained to be linear. Hypertext is text which contains links to other texts. The term was coined by Ted Nelson around 1965.

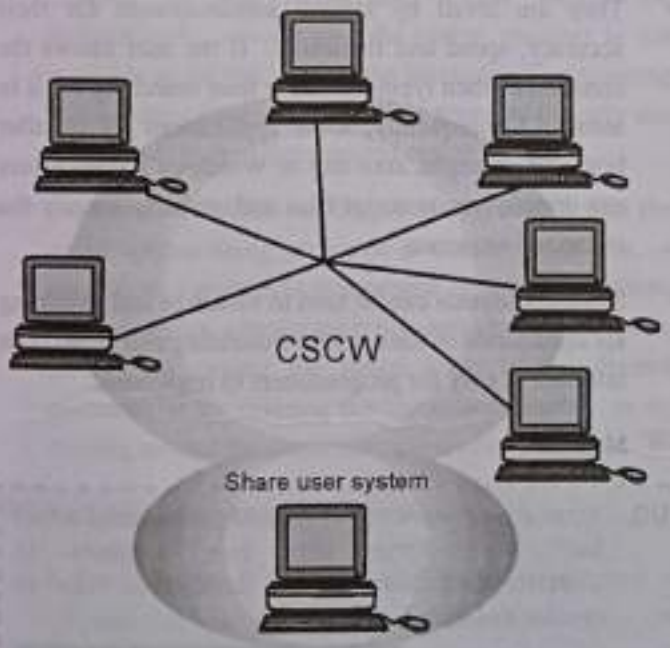


(1813)**Fig. 2.9.3 : Hypertext**

- HyperMedia is a term used for hypertext which is not constrained to be text: it can include graphics, video and sound, for example.

(8) Computer-supported cooperative work

- Personal computing provides individuals with enough computing power so that they were liberated from dumb terminals which operated on a time-sharing system.



(1814)**Fig. 2.9.4 : CSCW Interface**

- It is interesting to note that as computer networks became widespread, individuals retained their powerful workstations but now wanted to reconnect themselves to the rest of the workstations in their immediate working environment, and even throughout the world!
- One result of this reconnection was the emergence of collaboration between individuals via the computer – called computer-supported cooperative work, or CSCW.

(9) The World Wide Web

- WWW or "Web" is a global information medium which users can read and write via computers connected to the Internet.

(1815)**Fig. 2.9.5 : World Wide Web**

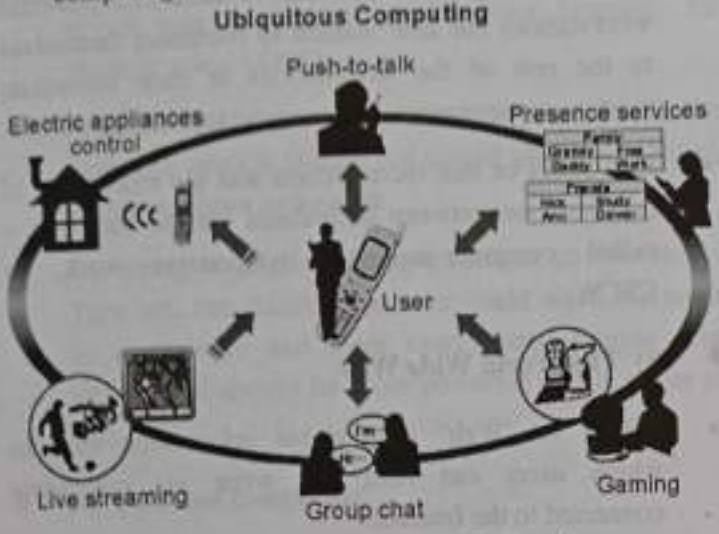
- The term is often mistakenly used as a synonym for the Internet itself, but the Web is a service that operates over the Internet, just as e-mail also does.
- The history of the Internet dates back significantly further than that of the World Wide Web.

(10) Ubiquitous computing

- Ubiquitous computing is a paradigm in which the processing of information is linked with each activity or object as encountered.
- It involves connecting electronic devices, including embedding microprocessors to communicate information.
- Devices that use ubiquitous computing have constant availability and are completely connected.
- Ubiquitous computing focuses on learning by removing the complexity of computing and increases efficiency while using computing for different daily activities.

Unit II In Sem.

- Ubiquitous computing is also known as pervasive computing, every ware and ambient intelligence.



(1818)**Fig. 2.9.6 : Ubiquitous Computing**

2.10 INTERACTION STYLES

GQ. Explain different Interaction Styles in HCI. **(10 Marks)**

- The concept of Interaction Styles refers to all the ways the user can communicate or otherwise interact with the computer system. The concept belongs in the realm of HCI or at least have its roots in the computer medium, usually in the form of a workstation or a desktop computer.
- These concepts do however retain some of their descriptive powers outside the computer medium. Interaction can be seen as a dialog between the computer and the user. The choice of interface style can have a profound effect on the nature of this dialog. There are a number of common interface styles including.

Command line interface

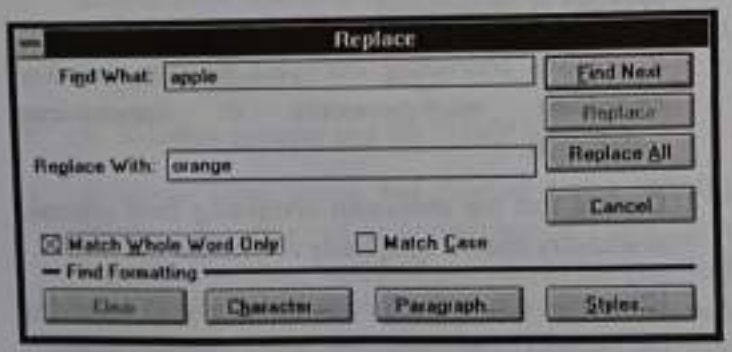


Fig. 2.10.1 : Command line interface

- The command line interface was the first interactive dialog style to be commonly used and, in spite of the availability of menu-driven interfaces, it is still widely used.
- It provides a means of expressing instructions to the computer directly, using function keys, single characters, abbreviations or whole-word commands. In some systems the command line is the only way of communicating with the system, especially for remote access using telnet.
- Menu-based interfaces, providing accelerated access to the system's functionality for experienced users. Command line interfaces are powerful in that they offer direct access to system functionality and can be combined to apply a number of tools to the same data.
- They are also flexible: the command often has a number of options or parameters that will vary its behaviour in some way, and it can be applied to many objects at once, making it useful for repetitive tasks. Flexibility and power brings with it difficulty in use and learning.
- Command line interfaces are the original interfaces for the computer and are still used. Examples of command line interfaces are UNIX operating system commands and the VI editor.
- They are loved by system administrators for their accuracy, speed and flexibility. If the user knows the commands, then typing is faster than searching for it in menus. Consequently, some applications try to offer both; for example, auto cad or Windows Excel. Users can directly create script files and verbally specify the command sequence.
- Some commands can be hard to visualize and searching for commands or files can be frustrating and slow. The interface is easy for programmers to implement.

Menus

UQ. What are differences between Menu bar and a tool bar? Many times user face problems in understanding learning tool bar icons. How to resolve this issue?

SPPU- Q. 2(b), Dec. 18, 5 Marks

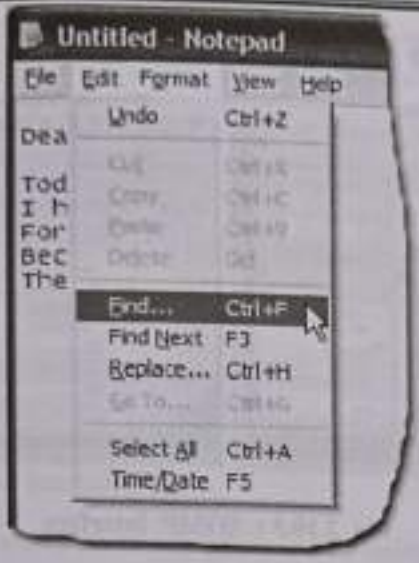


Fig. 2.10.2 : Menu structure

- Menus are a basis of WIMP interfaces and a favourite with inexperienced users. Novice users can easily find commands. Searching the menus, the user builds up a metaphor for the application.
- In a menu-driven interface, the set of options available to the user is displayed on the screen, and selected using the mouse, or numeric or alphabetic keys. Since the options are visible they are less demanding of the user, relying on recognition rather than recall.
- Menu options still need to be meaningful and logically grouped to aid recognition. Often menus are hierarchically ordered and the option required is not available at the top layer of the hierarchy. The grouping and naming of menu options then provides the only cue for the user to find the required option.
- Such systems either can be purely text based, with the menu options being presented as numbered choices, or may have a graphical component in which the menu appears within a rectangular box and choices are made, perhaps by typing the initial letter of the desired selection, or by entering the associated number, or by moving around the menu with the arrow keys.

☞ Natural language

- Natural language Users, unable to remember a command or lost in a hierarchy of menus, may long for the computer that is able to understand instructions expressed in everyday words! Natural language understanding, both of speech and written input, is the subject of much interest and research.
- The ambiguity of natural language makes it very difficult for a machine to understand. Language is ambiguous at a number of levels. First, the syntax, or structure, of a phrase may not be clear. If we are given the sentence. The boy hit the dog with the stick‖
- The most common example of natural language interface are the interface of search engines. They are hard to implement, but can be very flexible.

Unit II In Sem.

☞ Question/answer and query dialog

Fig. 2.10.3 : Question and Answer dialog

- Question and answer dialog is a simple mechanism for providing input to an application in a specific domain. The user is asked a series of questions (mainly with yes/no responses, multiple choice, or codes) and so is led through the interaction step by step.
- Dialogs or "question and answer" boxes are another old interface style. They are windows that pop up asking for information, fields to be filled or buttons to be pressed. Dialogue windows have been around before WIMP interfaces, they first appeared in database entry application programs.
- They are rather inflexible form of interface. Dialog boxes are easy to understand but not very flexible or fast. They are easy to program.

☞ Form-fills and spreadsheets

- Form-filling interfaces are used primarily for data entry but can also be useful in data retrieval applications.

- The user is presented with a display resembling a paper form, with slots to fill in Often the form display is based upon an actual form with which the user is familiar, which makes the interface easier to use.
- The user works through the form, filling in appropriate values. The data are then entered into the application in the correct place. Most form-filling interfaces allow easy movement around the form and allow some fields to be left blank.
- They also require correction facilities, as users may change their minds or make a mistake about the value that belongs in each field. The dialog style is useful primarily for data entry applications and, as it is easy to learn and use, for novice users.

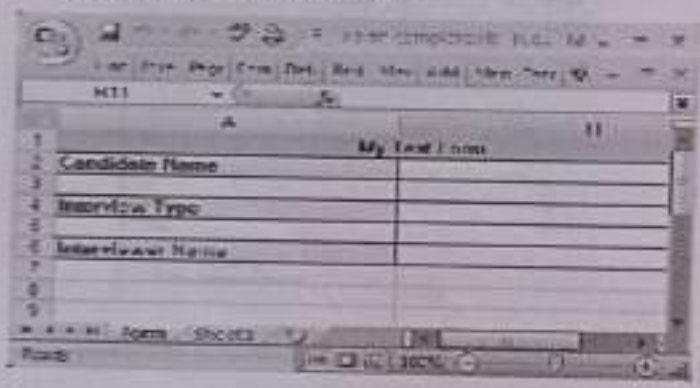


Fig. 2.10.4 : Spreadsheets

- Spreadsheets are a sophisticated variation of form filling. The spreadsheet comprises a grid of cells, each of which can contain a value or a formula. The formula can involve the values of other cells (for example, the total of all cellsin this column).
- The user can enter and alter values and formulae in any order and the system will maintain consistency amongst the values displayed, ensuring that all formulae are obeyed.
- The user can therefore manipulate values to see the effects of changing different parameters. Spreadsheets are an attractive medium for interaction: the user is free to manipulate values at will and the distinction between input and output is blurred, making the interface more flexible and natural.

WIMP

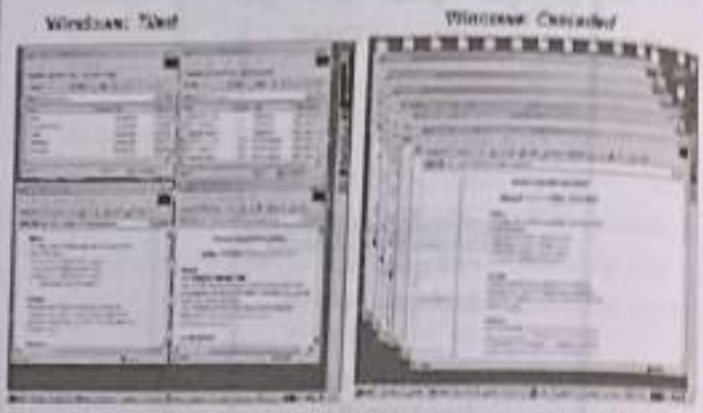


Fig. 2.10.5 : WIMP Interface

- WIMP stands for windows, icons, menus and pointers (sometimes windows, icons, mice and pull-down menus), and is the default interface style for the majority of interactive computer systems in use today, especially in the PC and desktop workstation arena. Examples of WIMP interfaces include Microsoft Windows for IBM PC compatibles, MacOS for Apple Macintosh compatibles and various X Windows-based systems for UNIX.

Point and click

- Point and click interfaces were made popular by the web. They were very suitable for the initial web browsers, when web pages were all text.
- Users knew to interpret the underscore as a link to another web page.
- Now, links are hidden, for examples in images. Icons on the desktop is another example of point and click style interface.

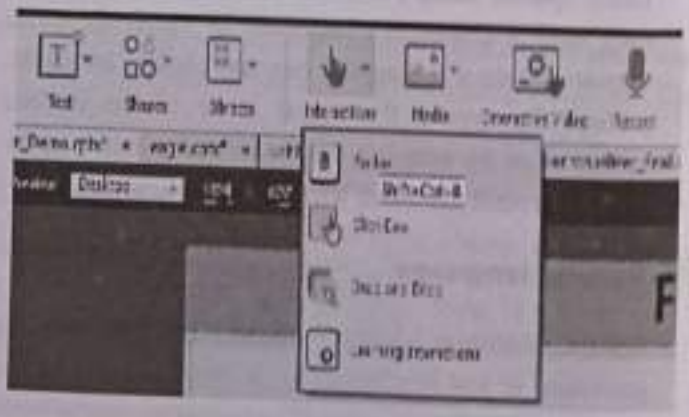


Fig. 2.10.6 : Point and Click

- The notion of point and click is a short interaction that results in a very specific result. Because the user must move the mouse, this interface style is slow.
- It is flexible because many different kinds of UI objects can be pointed at. Short key interaction is a point and click interaction style without the point.
- They are generally easy to implement.

☞ Three-dimensional interfaces

- There is an increasing use of three-dimensional effects in user interfaces.
- The most obvious example is virtual reality, but VR is only part of a range of 3D techniques available to the interface designer.
- The simplest technique is where ordinary WIMP elements, buttons, scroll bars, etc., are given a 3D appearance using shading, giving the appearance of being sculpted out of stone. By unstated convention, such interfaces have a light source at their top right.
- Where used judiciously, the raised areas are easily identifiable and can be used to highlight active areas.

⏭ 2.11 INTERACTIVITY

- Dialog design is focused almost entirely on the choice and specification of appropriate sequences of actions and corresponding changes in the interface state.
- It is typically not used at a fine level of detail and deliberately ignores the semantic level of an interface: for example, the validation of numeric information in a forms-based system.

Fig. 2.11.1: Interactivity

- It is worth remembering that interactivity is the defining feature of an interactive system. This can be seen in many areas of HCI. For example, the recognition rate for speech recognition is too low to allow transcription from tape, but in an airline reservation system, so long as the system can reliably recognize yes and no it can reflect back its understanding of what you said and seek confirmation.
- Speech-based input is difficult, speech-based interaction easier. Also, in the area of information visualization the most exciting developments are all where users can interact with visualization in real time, changing parameters and seeing the effect. Interactivity is also crucial in determining the feel of a WIMP environment.

Unit II In Sem.

⏭ 2.12 CONTEXT OF INTERACTION

- In Human Computer Interaction studies, the context describes the actual conditions under which the software system is used. Determining the context of the system means describing how the software system interacts with the user in normal day to day situations.
- It is important to carry out usability tests, prototyping sessions, meetings, user studies and other "user-dependent sessions" in the correct context of the system to get the most accurate results from your findings.
- In context-aware software systems, determining the context of use can allow the application to modify it's current behaviour to better interact with the user.
- Context information will typically include anything that can be used to characterize the situation of the user, system or any other relevant entities.
- Context can be decomposed into disjoint categories or types to help define the context of the software system.
- Although these context types may differ by opinion, the most commonly recognized are the User Context, the Time Context, the Physical Context and the Computing Context. These four contexts are described below.

User Context

- The user context (also known as personal context) represents information about the end-user, which interacts with the system.
- This includes information such as the user profile (age, preferences), the user's location (e.g. absolute position, indoors, outdoors) and orientation, nearby objects, the people nearby and the social situation.

Time Context

The time context covers relevant information related to time such as absolute time, date, day of the week and season.

Physical Context

- The physical context includes everything which is measurable in the environment of the system with which the user interacts. This includes temperatures, noise levels, lighting situations, traffic conditions, etc.
- In the simulator presented to the right, the physical world is affected by the computer simulation. The actuated chassis responds to the user's input through the computer.

Computing Context

- The computing context contains everything related to computational resources.
- This can include things such as available networks, network bandwidth, communication costs and nearby computational resources such as printers or fax machines.

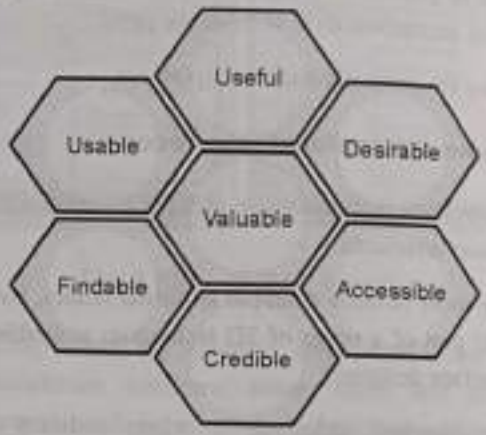
Examples of HCI Contexts

(1) Consumer Devices (2) Mobile Devices

(3) Business Applications (4) Games

2.13 USER EXPERIENCE

- User experience (UX) focuses on having a deep understanding of users, what they need, what they value, their abilities, and also their limitations.
- It also takes into account the business goals and objectives of the group managing the project. UX best practices promote improving the quality of the user's interaction with and perceptions of your product and any related services.



(1817)**Fig. 2.13.1: User Experience**

He notes that in order for there to be a meaningful and valuable user experience, information must be:

(1) **Useful :** Your content should be original and fulfil a need.

(2) **Usable :** Site must be easy to use

(3) **Desirable :** Image, identity, brand, and other design elements are used to evoke emotion and appreciation.

(4) **Findable :** Content needs to be navigable and locatable onsite and offsite

(5) **Accessible :** Content needs to be accessible to people with disabilities.

(6) **Credible :** Users must trust and believe what you tell them

The more practical and design-based aspects of HCI, which tend to be more customer-facing, fall into the field of user experience (UX). As the name suggests, UX focuses on how a user experiences their interaction with a type of technology.

Chapter Ends...

❑❑❑

UNIT III

CHAPTER 3

HCI Models and Theories

Syllabus

User Profiles, categorization of users, Goal and task hierarchy model, Linguistic model, Physical and device models, GOMS, Norman's 7 stage model, Cognitive architectures, Hierarchical task analysis (HTA), Uses of task analysis, Diagrammatic dialog design notations.

3.1 USER PROFILES

A user profile is a representation of information about an individual user that is essential for intelligent application. User profile is a popular term widely employed during product design processes by industrial companies. Such a profile is normally intended to represent real users of a product.

The ultimate purpose of a user profile is actually to help designers to recognize or learn about the real user by presenting them with a description of a real user's attributes, for instance; the user's gender, age, educational level, attitude, technical needs and skill level.

How to Create User Profile : Example Travel Agent

Example

Travel Agent (Primary) Characteristic Ranges		
Age	:	25-40 years (Average : 32 years)
Gender	:	80% female
Job Titles	:	Travel agenet, Travel specialist, Travel associate
Experience Level	:	0-10 years (Typical : 3 years)
Work Hours	:	40 hours per week; days and times depend on the company
Education	:	High school to Bachelors degree (Typical : some college)
Location	:	Anywhere in the U.S. (Predominantly mid-west)
Income	:	$25,000 – $50,000 / year ; depends on experience level and location (Average : $35000/year)
Technology	:	Some computer experience; high speed internet connection
Disabilities	:	No specific limitations
Family	:	Single or married (Predominantly married with 1 child)

Fig. 3.1.1 : User Profile of Travel Agent

- User profiling involves gaining an understanding and building a profile of the final users of the system in terms of age, gender, socio economic background, abilities, knowledge, skill sets, frequency, interest and any other relevant information.
- It included the process of establishing knowledge about the users to find out who users are, such as children, elderly, professional, male or female all the possibilities.
- It is important when your interface, website, or object has multiple audiences, profiling will reveal potential clashes of interest. For instance, an educational website when viewed by parents, teachers and children will have to cater for each groups' different perspectives.
- Furthermore, you have a place from which to start making decisions or trade-offs. For example, to apply for leave, check schedule, fill in from, and get approval from administrators.
- Other than this features, such as results from all the students or examination questions that only available for lecturers to view, the website should make the availability for every perspective clearly.
- User profiling is where to recognize the tasks that have to be performed for the satisfaction of users. User

Unit III End Sem.

profiling is a necessity to identify the attributes of users, physical characteristics (height, physical abilities or disabilities), background (education, social, religious), skills (task experience), or preferences (efficiency) in order to reduce the failure on the system when you meet the needs of specific users.

- Moreover, when a system does not meet the user's requirements, the loss of huge amount of money will caused due to the unaccepted result from user, and the rebuild of new system will consume more time to complete where you basically loss the time and cash simultaneously.

3.2 CATEGORIZATION OF USERS

There are 3 main types of user

(1) Novice

(2) Knowledgeable / intermittent user

(3) Expert / frequent user

Users can however, be classified in any other way that is appropriate to the system being built.

- Some users may have keyboard skills, others not.
- Some users may have knowledge of other similar systems, others not.

The advantages of classification mean that generalizations can be made about users and their needs. This doesn't necessarily mean that the best system has been designed for every individual. It means that the system has been designed to fit the generalizations for each user group.

1) The Novice

For the novice user of a system, progress is slow because of the limitations of working memory. Chunking is almost entirely absent.

Systems used by novices require more feedback and more opportunities for closure.

Guidelines for novice users

- All initiatives should come from the computer – the novice may not know what is to be done
- Each required input should be brief – the shorter it is the more likely it is to be remembered
- Input procedures should be consistent with user expectations – humans search for patterns and will generalize
- No special training should be necessary – especially true in the case of web or multimedia where the user is 'on their own'
- All system messages should be clear – in the language of the user, not the designer
- User decision should be made from a small set of options – the more of a selection you offer, the harder it is to choose
- Users should control the pace of interaction - they need to understand the system and feel that they can control it, and not the reverse
- User decision making should be a response to a specific request for action – save - y/n?
- Help should always be available – tutor / book / online. There should be sufficient feedback – closure

2) Knowledgeable / intermittent users

These users need consistent structures, good help facilities, good documentation.

3) Expert users

These users have fast response time and will require brief feedback. Experts organize their knowledge according to a higher conceptual structure.

They can recall more than novices because their knowledge is chunked. Expert users will look for keyboard shortcuts, abbreviated sequences. Experts can find constant confirmation screens irritating - Use these only when important.

Examples

Logging on according to the expert's view and the actual steps

Expert steps for logging on

(1) Input username (2) Input password

(3) Actual steps for logging on system

(4) Press any key to activate screen

(5) Click into input box

(6) Input username

(7) Press tab or click into second input box

(8) Input password

(9) Press return

Wait for welcome message or error message ,if welcome message then tasks ends, if error message then repeat 2-7 or toggle 'caps-lock' and repeat 2-7 or ask for help

3.3 GOAL AND TASK HIERARCHY MODEL

GQ. Difference b/w goal and task? (6 Marks)

3.3.1 Goals, Tasks and Actions

- **Goal :** state of a system that the human wishes to achieve using some device. Goal is an intention what you would like to be true
- **Tasks :** activities required to achieve a goal. The device selected to achieve the desired goal determines the tasks and their level of complexity. Actions how to achieve goal
- **Actions :** physical interaction with a device. Task involving no problem solving

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.

- **Example :** Sales report

produce report

gather data.

find book names

..... do keywords search of names database

..... further sub-goals

..... sift through names and abstracts by hand

..... further sub-goals.

search sales database

- further sub-goals

layout tables and histograms

- further sub-goals

write description

- further sub-goals

- Where do we stop? We can go on decomposing tasks until we get down to the individual hand and eye movements of the user, or we can stop at a more abstract level. Where do we start?
- In a similar way, we can start our analyses at different points in the hierarchy of goals. At the extreme we could extend our analysis to larger and larger goals: 'light hob' is a subgoal of 'boil peas' and so on to goals such as 'have my dinner', 'feed' and 'stay alive'.
- 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 HCI 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.
- What do we do when there are several ways of solving a problem, or if the solutions to two subgoals interact? Users will often have more than one way to achieve a goal and there must be some way of representing how they select between competing solutions.

Unit III End Sem.

3.4 LINGUISTIC MODEL

3.4.1 Cognitive Modeling

Cognitive modeling is an area of computer science that deals with simulating human problem-solving and mental processing in a computerized model. Such a model can be used to simulate or predict human behavior or performance on tasks similar to the ones modeled and improve human-computer interaction.

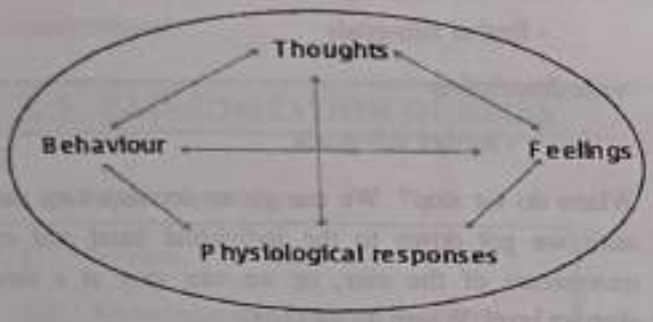


Fig. 3.4.1 : Cognitive Modeling

Cognitive Models

- The field of human-computer interaction, whose goal is to make computers support human activity in much more satisfying ways than they currently do, has three main uses for cognitive modeling.
- A cognitive model can substitute for a human user to predict how users will perform on a system before it is implemented or even prototyped.
- A system can generate a cognitive model of the user currently interacting with the system in order to modify the interaction to better serve that user.

Finally, cognitive models can substitute directly for people so groups of individuals can be simulated in situations that require many participants, e.g., for training or entertainment.

- A system can generate a cognitive model of an end user currently interacting with the design and modify the interaction of the user with respect to the design to understand different outcomes.
- Finally, cognitive models in design can replace a groups of individual users and can act in those situations that require many participants.

Consider your designing a banking application which include multiple personas and multiple users using it, in such huge applications asking multiple questions to multiple users to derive the best possible user flow and understanding how will the user react to the flow is crucial.

Cognitive models represent users of interactive systems.

- Hierarchical models represent a user's task and goal structure. Linguistic models represent the user–system grammar.
- Physical and device models represent human motor skills.
- Cognitive architectures underlie all of these cognitive models

GQ. What is BNF and give example. **(9 Marks)**

BNF is a Very common notation from computer science. A purely syntactic view of the dialogue

- **Terminals :** lowest level of user behaviour.

  e.g. CLICK-MOUSE, MOVE-MOUSE
- **Nonterminals :** ordering of terminals , higher level of abstraction. E.g. select-menu, position-mouse

The user's interaction with a computer is often viewed in terms of a language, so it is not surprising that several modeling formalisms have developed centered around this concept. BNF grammars are frequently used to specify dialogs.

The models here, although similar in form to dialog design notations, have been proposed with the intention of understanding the user's behavior and analyzing the cognitive difficulty of the interface. Understanding the user's behaviour and cognitive difficulty based on analysis of language between user and system.

Backus – Naur Form (BNF)

- Representative of the linguistic approach is Reisner's use of Backus–Naur Form (BNF) rules to describe the dialog grammar. 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.
- Backus–Naur Form (BNF) is:
  - Very common notation from computer science
  - A purely syntactic view of the dialogue
  - Terminals – lowest level of user behaviour – E. g. CLICK-MOUSE, MOVE-MOUSE
  - Nonterminals – ordering of terminals – higher level of abstraction – e. g. select-menu, position-mouse
- Example of BNF
  - Basic syntax : – nonterminal : : = expression
  - An expression – contains terminals and nonterminals – combined in sequence (+) or as alternatives (I)

    draw-line::= select-line + choose-points + last-point

    select-line::= position-mouse + CLICK-MOUSE

    choose-points::= choose-one I choose-one + choose-points

    choose-one ::= position-mouse + CLICK-MOUSE

    last-point ::= position-mouse + DOUBLE-CLICK-MOUSE

    position-mouse ::= empty I MOVE-MOUSE + position-mouse
- The aims 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 'I'(choice).
- For example, the first rule says that the nonterminal draw-line is defined to be select-line followed by choose-points followed by last point. 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. Position-mouse is, we look at the last rule. This tells us that there are two possibilities for position-mouse (separated by the 'I' symbol).
- One option is that position-mouse is empty – a special symbol representing no action. That is, one option is not to move the mouse at all. The other option is to do a MOVE-MOUSE action followed by position-mouse.
- This rule is recursive, and this second position-mouse may itself either be empty or be a MOVE-MOUSE action followed by position-mouse, and so on. That is, position-mouse may be any number of MOVE-MOUSE actions whatsoever.
- Similarly, choose-points is defined recursively, but this time it does not have the option of being empty. It may be one or more of the non-terminal choose-one which is itself defined to be (like select-line) position-mouse followed by CLICK-MOUSE.
- The BNF description of an interface can be analyzed in various ways. One measure is to count the number of rules. The more rules an interface requires to use it, the more complicated it is.
- This measure is rather sensitive to the exact way the interface is described. For example, we could have replaced the rules for choose points and choose-one with the single definition choose-points ::= position-mouse + CLICK-MOUSE I position-mouse + CLICK-MOUSE + choose-points.

Task-Action Grammar (TAG)

GQ. What is TAG ? (6 Marks)

- Task – action grammar (TAG) attempts to deal with some of these problems by including elements such as parameterized grammar rules to emphasize consistency and encoding the user's world knowledge
- 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) is:
  - Making consistency more explicit
  - Encoding user's world knowledge
  - Parameterised grammar rules
  - Nonterminals are modified to include additional semantic features
  - Consistency in TAG
- In BNF, three UNIX commands would be described as:
  - copy : : = cp + filename | cp + filenames + directory
  - move: : = mv + filename | mv + filenames + directory
  - link : : = ln + filename | ln + filenames + directory
- No BNF measure could distinguish between this and a less consistent grammar in which

  link: : = ln + filename | ln + directory + filenames
- consistency of argument order made explicit using a parameter, or semantic feature for file operations
- Feature Possible values Op = copy; move; link
- Rules file-op[Op] : : = command[Op] + filename | command[Op] + filenames + directory
  - command[Op = copy] : : = cp
  - command[Op = move] : : = mv
  - command[Op = link] : : = ln

3.5 PHYSICAL AND DEVICE MODELS

3.5.1 Keystroke-Level Model (KLM)

- The model provides a quantitative tools. To predict time to accomplish a task with a given method on an interactive computer system. The model is based on counting keystrokes and low-level operations, including user's mental operations and the system responses
- This models appears as simple, accurate enough and flexible to be applied in practical design and evaluation situations. One of the earliest and most comprehensive models in HCI. Developed specifically for predicting human performance with interactive computing systems. Predicts expert error-free task completion times

3.5.2 Elements of a KLM Prediction

- Task (or a series of tasks)
- Method used
- Command language of the system
- Motor skill parameters of the user
- Response time parameters of the system

The human motor system is well understood. KLM (Keystroke-Level Model) 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.

- 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 Keystroke - Level Model (KLM) predicts how long it will take an expert user to accomplish a routine task without errors using an interactive computer system. The actions are termed keystroke level if they are at the level of actions like pressing keys, moving the mouse, pressing buttons.
- There is a standard set of operators for use in the KLM, whose execution times have been estimated from experimental data.

The following is a step-by-step description of how to apply the KLM to estimate the execution time required by a specified interface design :

1. Choose one or more representative task scenarios.
2. Have the design specified to the point that keystroke-level actions can be listed for the specific task scenarios.
3. For each task scenario, figure out the best way to do the task, or the way that you assume users will do it.
4. List the keystroke-level actions and the corresponding physical operators involved in doing the task.
5. If necessary, include operators for when the user must wait for the system to respond
6. Insert mental operators for when user has to stop and think.
7. Look up the standard execution time to each operator.
8. Add up the execution times for the operators.
9. The total of the operator times is the estimated time to complete the task.

The model decomposes the execution phase into five different physical motor operators, a mental operator and a system response operator:

- Physical motor operators

  K – Key stroking, 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.

- A mental operator

  M Mentally preparing for a physical action.

- A system response operator

  R System response which may be ignored if the user does not have to wait for it, as in copy typing.

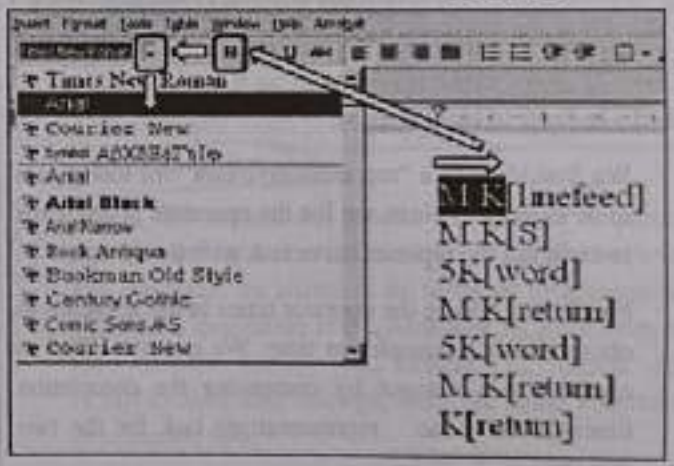


Fig. 3.5.1 : Keystroke Level Model

- 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]

Unit III End Sem.

5. Type correction K[char]
6. Reposition insertion point H[mouse] MPB[LEFT]

UQ. What is a mental operator in the Keystroke Level Model (KLM)? How it is different from physical operators? **SPPU - Q. 9(a), May 19, 9 Marks**

☑ **Ans. :**

Hint : The physical operators indicate the time required to perform physical (motor) tasks, such as moving hand between mouse and keyboard, dragging mouse pointer to a target and so on. However, the mental operator indicates the time required for cognitive (thinking) tasks, such as decision making, problem solving etc., which are internal activities of a user.

UQ. Discuss the steps involved in estimating task execution time using KLM. How we can use KLM to compare competing designs? **(SPPU - Q. 10(a), May 19, 9 Marks)**

☑ **Ans. :**

Hint : In order to build a KLM,

- We first identify a "representative task" for the system to be evaluated. Next, we list the operators in sequence to carry out the representative task with the system.
- Finally, we add up the operator times in the sequence to obtain the task completion time. We can use KLM to compare two designs by comparing the completion times of the same representative) task for the two designs.

The following is a step-by-step description of how to apply the KLM to estimate the execution time required by a specified interface design:

(1) Choose one or more representative task scenarios.

(2) Have the design specified to the point that keystroke-level actions can be listed for the specific task scenarios.

(3) For each task scenario, figure out the best way to do the task, or the way that you assume users will do it.

(4) List the keystroke-level actions and the corresponding physical operators involved in doing the task.

(5) If necessary, include operators for when the user must wait for the system to respond

(6) Insert mental operators for when user has to stop and think.

(7) Look up the standard execution time to each operator.

(8) Add up the execution times for the operators.

(9) The total of the operator times is the estimated time to complete the task.

☞ Operators and Times

The following are the standard operators and estimated times for each operator.

- **K** - Keystroke (.12 - 1.2 sec; .28 recommended for most users). This operator is pressing a key or button on the keyboard. Pressing the SHIFT or CONTROL key counts as a separate keystroke. Different experience levels have different times for the K operator:
  - Expert typist (90 wpm): .12 sec
  - Average skilled typist (55 wpm): .20 sec
  - Average nonsecretarial typist (40 wpm): .28 sec
  - Worst typist (unfamiliar with keyboard): 1.2 sec
- The average nonsecretarial typist (.28 sec) is a good design point for characterizing the typical computer user; these are people familiar enough with the keyboard to use it fluently, but are not professional-grade typists.
- **T(n)** - Type a sequence of n characters on a keyboard (n x K sec). This operator is simply a shorthand for a series of K operators, and would normally be used only when the user is typing a string of characters that is a single "chunk," such as a filename.
- **P** - Point with mouse to a target on the display (1.1 sec). This operator represents the action of moving the mouse to point the cursor to a desired place on the screen.
- For typical situations, it ranges from .8 to 1.5 sec, with an average of 1.1 sec. If great accuracy is not required, or the movement distances or target sizes are not unusual, this average can be used instead of more precise times.

- **B** - Press or release mouse button (.1 sec). This is a highly practiced, very rapid movement. .1 sec for pushing the button down or letting it up.
- **BB** - Click mouse button (.2 sec). Pushing and releasing the mouse button rapidly, as in a selection click, counts as two B operators, for a total of .2 sec.
- **H** - Home hands to keyboard or mouse (.4 sec). Since the targets are pretty large, and the movement well practiced, moving the hand between keyboard and mouse, and vice-versa, is relatively fast.
- **M** - Mental act of routine thinking or perception (.6 - 1.35 sec; use 1.2 sec). Of course, how long it takes to perform a mental act depends on what cognitive processes are involved, and is highly variable from situation to situation or person to person. This operator is based on the fact that when reasonably experienced users are engaged in routine operation of a computer, there are pauses in the stream of actions that are about a second long and that are associated with routine acts such as remembering a filename or finding something on the screen.
- The **M** operator is intended to represent this routine thinking, not complex, lengthy, problem-solving, racking the brain, or creative meditations. In a variety of routine computer usage tasks such as word processing and spreadsheet usage, these routine pauses are fairly uniform in length, justifying the simplifying assumption that all Ms take the same amount of time, around one sec.
- Based on the available results (Olson & Olson, 1990), a good overall estimate for the duration of an M is 1.2 sec.
- Choosing how many Ms are involved, and where they appear, is the hardest part of using the KLM.
- **W(t)** - Waiting for the system to respond (time t must be determined). This is the time that the user must wait on the system before he or she can proceed. Notice that it is not necessarily the same as the time required by the system, because the user may be able to overlap other activities while the system is working.
- In many cases with modern computers, the waiting time is essentially negligible. In other cases, such as a database retrieval or website usage, the delay might be substantial, but still the same regardless of which interface design is under consideration.
- In such cases, it may not be necessary to include this operator, either because it takes near-zero time, or it has the same value in all of the alternative designs, and so affects only the absolute, not the relative, task times.

UQ Discuss the key differences between KLM and (CMN)GOMS. **(SPPU - Q. 9(b), May 19, 9 Marks)**

☑ **Ans. :**

- CMN-GOMS is the original GOMS model proposed by Stuart Card, Thomas P. Moran and Allen Newell. CMN stands for Card, Moran and Newell and it takes the KLM as its basic and adds subgoals and selection rules. This model can predict operator sequence as well as execution time. A CMN-GOMS model can be represented in program form, making it amenable to analysis as well as execution. CMN-GOMS has been used to model word processors and CAD systems for ergonomic design.

Unit III End Sem.

- The CMN method can predict the operator sequence and the execution time of a task on a quantitative level and can focus its attention on methods to accomplish goals on a qualitative level. Although KLM belongs to the GOMS family of models in which the (CMN)GOMS also belongs, there are some difference between the two.
- (CMN)GOMS is much more robust in modeling human cognitive behavior than KLM, which is a very simple and sequential cognitive model. In (CMN) GOMS, cognitive process is modeled as a hierarchy of goals, thus revealing more complexities than KLM.
- From the hierarchy, we can reason about the extent of memory recall and decision making involved. This gives us an extra way to compare to designs along with the task completion time measure. KLM (Keystore - Level) Model predicts expert error - free task completion time (human performance) with interactive computing systems. Total predicted time for a task is given by equation.

$$tEXECUTE = tK + tP + tH + tD + tM + tR.$$

UQ. What does each of the above timing represent ? Develop a KLM model and predict time for the completion of the task "Change font and style for the word "KLM" to bold, Arial" using mouse only.

(SPPU - Q. 9(a), May 18, 9 Marks)

☑ **Ans. :**

- A task is broken into a series of subtasks. Total predicted time is the sum of the subtask times :

  tEXECUTE = tK + tP + tH + tD + tM + tR

  Operators :

  K – keystroking, P – pointing, H - homing

  D – drawing, M - mental prep, R - system response

- **Task :** Change the font and style for word "KLM" to bold, Arial.

  Operations :

Mouse Subtasks K	KLM Operators	tP(s)
Drag across text to select' "KLM"	M P[2.5, 0.5]	0.686
Move pointer to Bold button and click	M P[13, 1]	0.936
Move pointer to Font drop-down button and click	M P[3.3, 1]	0.588
Move pointer down list to Arial and click	M P[2.21]	0.501
	Σ tP =	2.71

Prediction :

$$tEXECUTE = 4 \times tM + \Sigma tP$$
$$= 4 \times 1.35 + 2.71 = 8.11 \text{ seconds}$$

Operations :

- Keyboard Subtasks
- Select text
- Convert to boldface
- Activate Format menu and enter Font Sub-menu
- Type a ("Arial" appears at top of list)
- Select "Arial"

$$tEXECUTE = 4 \times tM + 12 \times tK$$
$$= 4 \times 1.35 + 12 \times 0.75 = 14.40 \text{ seconds}$$

Use "typing complex codes" (tK = 0.75 s)

- **Limitations keystroke - level model**
  - It measures only one aspect of performance : time, which means execution time and not the time to acquire or learn a task.
  - It considers only expert users. Generally, users differ regarding their knowledge and experience of different systems and tasks, motor skills and technical ability.
  - It considers only routine unit tasks. The method has to be specified step by step. The execution of the method has to be error-free.
  - The mental operator aggregates different mental operations and therefore cannot model a deeper representation of the user's mental operations. If this is crucial, a GOMS model has to be used.

3.6 GOMS

- The GOMS is an acronym for Goals, Operators, Methods and Selection. GOMS is a task analysis technique. GOMS is family of user interface modeling techniques.
- A GOMS model is composed of methods that are used to achieve specific goals. These methods are then composed of operators at the lowest level.
- The operators are specific steps that a user performs and are assigned a specific execution time. If a goal can be achieved by more than one method, then selection rules are used to determine the proper Method.
- The GOMS model has four components: goals, operators, methods and selection rules.
  1. **Goals -** Tasks are deconstructed as a set of goals and subgoals. GOMS the goals are taken to represent a 'memory point' for the user

2. **Operators** - Tasks can only be carried out by undertaking specific actions. Example : to decide which search engine to use.
3. **Methods** - It represent ways of achieving a goal. Example : drag mouse over field.
4. **Selection Rules** - The method that the user chooses is determined by selection rules.

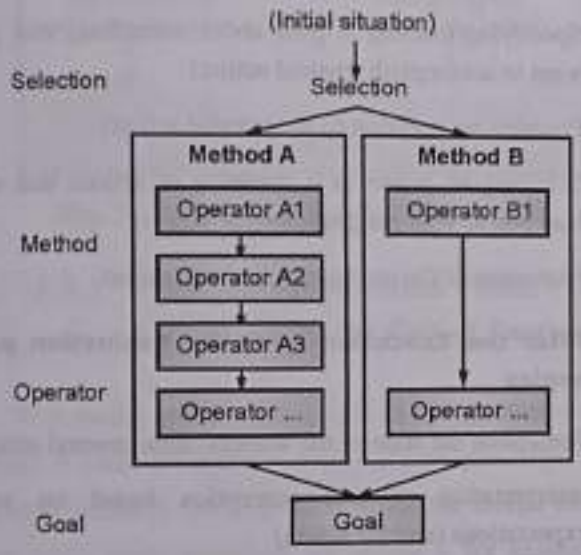


Fig. 3.6.1 : GOMS

3.6.1 Example

UQ. Create a GOMS description of the task of photocopying a paper from a Journal. Discuss the issue of closure in terms of your GOMS description.

(SPPU - Q. 10(a), May 18, 9 Marks)

Ans. :

GOMS for photocopying a paper from journal

- One possible GOMS description of the goal hierarchy for this task is given below. Answers will vary depending on assumptions about the photocopier used as the model for the exercise.
- In this example, we will assume that the article is to be copied one page at a time and that a cover over the imaging surface of the copier has to be in place before the actual copy can be made.
  - Goal : PHOTOCOPY-PAPER
  - Goal : LOCATE-ARTICLE
  - Goal : PHOTOCOPY-PAGE repeat until no more pages
  - [Select Goal : SELECT-PAGE --> CHOOSE-PAGE-TO-COPY]
  - Goal : ORIENT-PAGE
  - OPEN - COVER
  - POSITION-PAGE
  - CLOSE-COVER
  - PRESS-BUTTON
  - Goal: VERIFY-COPY
  - LOCATE-OUT-TRAY
  - EXAMINE-COPY
  - Goal : COLLECT-COPY
  - LOCATE-OUT-TRAY
  - REMOVE-COPY (outer goal satisfied!)
  - Goal: RETRIEVE-JOURNAL
  - OPEN-COVER
  - REMOVE-JOURNAL
  - CLOSE-COVER

Selection rules exist if a spoiled copy was printed. Consider the following :

- **Rule 1 :** SELECT-PAGE if last page was copied successfully or start of article.

  Note : The goal SELECT-PAGE is only valid if we are at the start of the article or the last copy was successful. If the last copy was spoiled the we must recopy the current page, so only a re-orientation would be required.
  - Goal : PHOTOCOPY-PAPER
  - Goal : LOCATE-ARTICLE
  - Goal : PHOTOCOPY-PAGE repeat until no more pages
  - [Select Goal : SELECT-PAGE --> CHOOSE-PAGE-TO-COPY]
  - Goal : ORIENT-PAGE
  - OPEN -COVER

Unit III End Sem.

- POSITION-PAGE
- CLOSE-COVER
- PRESS-BUTTON
- Goal : VERIFY-COPY
- LOCATE-OUT-TRAY
- EXAMINE-COPY
- Goal : RETRIEVE-JOURNAL
- OPEN-COVER
- REMOVE-JOURNAL
- CLOSE-COVER
- Goal : COLLECT-COPY
- LOCATE-OUT-TRAY
- REMOVE-COPY (outer goal satisfied!)

Closure to Outer Goal, must force user to collect copy last.

3.6.2 Advantages and Disadvantages of GOMS

Advantages

1. Easy to construct a simple GOMS model and saves time.
2. Helps discover usability problems.
3. Gives several qualitative and quantitative measures.
4. Less work than usability study.

Disadvantages

1. Only work for goal directed tasks.
2. Not for the novice user.
3. Not ideal for leading edge technology systems.
4. Not as easy as heuristics analysis, guidelines

3.7 NORMAN'S 7 STAGE MODEL

- One of the topics that usability researcher Donald Norman discusses in his book, "The Design of Everyday Things", is a model of how people interact with the real world. This model is called the seven stages of action, or Norman's action cycle.
- The model belongs to one of the most famous Interaction theories that have been used to model user behavior, evaluation, and to set up policies like to create user-friendly interfaces.
- The model can be divided into an execution phase and a phase of the evaluation.

It starts with the Execution part

- Specifying/forming a goal about something that you want to accomplish (mental action)
- Forming an intention to act (mental action)
- Selecting an action or a sequence of actions that will lead you to your set goal (mental action)
- Execution of the action(s) (physical action)

After the Execution part, the Evaluation part begins

- Perception the state of the world/system (mental action)
- Interpretation of this perception based on your expectations (mental action)
- Evaluating the outcome: Is the evaluation successful the problem is solved , goal reached and so on (mental action)

An example

(1) My goal is going to a cafe across the street

(2) I form the intention of crossing the street to reach the cafe

(3) I select the action to press the button of the crosswalk signal for crossing the street

(4a) I actually do press the button of the crosswalk signal (physical action)

(4b) I do cross the street (physical action)

(5) I perceive that I'm now across the street

(6) I interpret this perception and..

(7) Evaluate the outcome – was this action successful in moving closer to my goal? (Y/N)

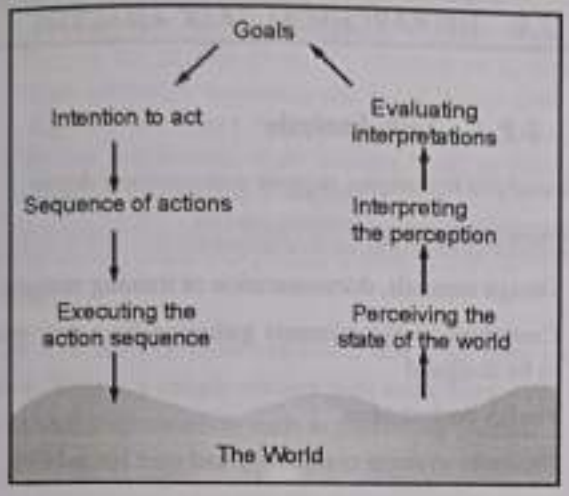


Fig. 3.7.1 : Seven stages of Action Model

3.7.1 Seven Stages of Action Model : Gulf of Evaluation and Gulf of Execution

This model can give us insight into ways that systems can fail or cause problems in supporting peoples larger goals and actions and it can help us to adapt the design towards users goals when designing for people – for example by making options visible (which helps with the execution part) and giving proper feedback (which will help with the evaluation part). Norman refers to problems with those parts either as "Gulf of Execution" or "Gulf of Evaluation".

Seven stages of Action

- Forming the goal
- Forming the intention(plan)
- Specifying an action(specify) } Execution
- Executing the action(perform)
- Perceiving the state of the world(perceive)
- Interpreting the state of the world(reflect) Evaluation
- Evaluating the outcome(compare)

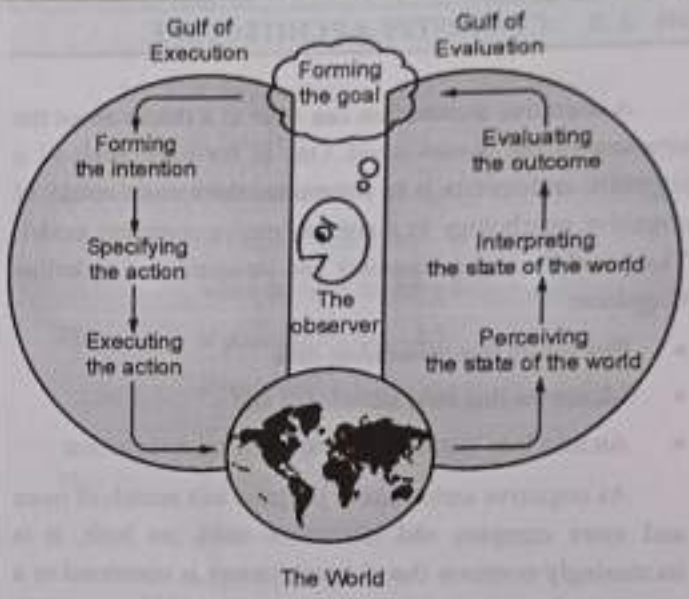


Fig. 3.7.2 : Gulf of Evaluation and Gulf of Execution

Unit III End Sem.

The Gulf of Execution

- The difference between the intentions and the allowable actions.
- The Gulf of Execution defines the gap/gulf between a user's goal and the device to implement that goal. One of the principal objective of Usability is to diminish this gap by removing barriers and follow steps to minimize the user's distraction from the intended task that would prevent the flow of the work.

The Gulf of Evaluation

- The Gulf of evaluation reflects the amount of effort that the person must exert to interpret the physical state of the system and to determine how well the expectations and intentions have been met.
- The Gulf of Evaluation is the representation of expectations that the user has interpreted from the system in a design.
- As per Donald Norman, the gulf is small when the system provides information about its state in a form that is easy to get, is easy to interpret, and matches the way the person thinks of the system.

3.8 COGNITIVE ARCHITECTURE

A cognitive architecture can refer to a theory about the structure of the human mind. One of the main goals of a cognitive architecture is to summarize the various results of cognitive psychology in a comprehensive computer model. The possibility of mechanism and structure that underline Cognition:

- Processors that manipulate data
- Memories that hold knowledge and
- An interface that hold interact with an environment

As cognitive architectures progress and models of more and more complex and interactive tasks are built, it is increasingly common that the architecture is connected to a complex simulation of the environment in which the task is performed. In some cases, the cognitive architecture interacts directly with the actual software that humans use to perform the task.

In other cases some form of connecting software must be constructed. Overall, a model of a task generally has three components: The architecture, task knowledge, and a dynamic task environment with which the model interacts. The output of this system is, as mentioned, a time stamped behavior stream, as depicted in following Fig. 3.8.1.

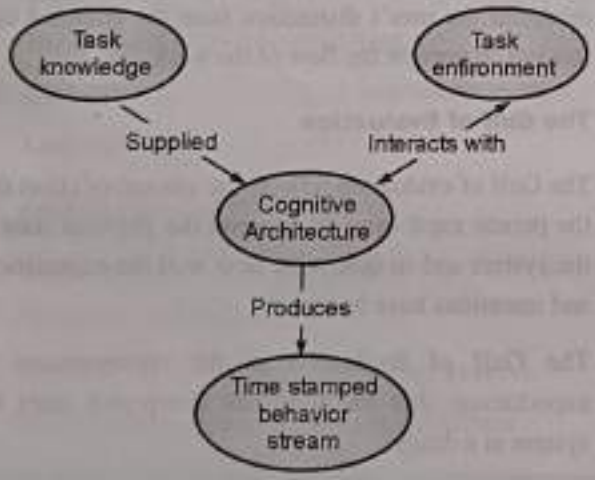


Fig. 3.8.1 : Structure of a model based on a cognitive architecture

Cognitive psychology is replete with theories about particular aspects of human cognition; a cognitive architecture is an attempt to build an integrated theory that encompasses a broad spectrum of what is known about human cognition and performance.

3.9 HIERARCHICAL TASK ANALYSIS

3.9.1 Task Analysis

Task analysis techniques support user-centered design

They are used to:

- Design manuals, documentation or training material
- Contribute to requirements gathering for a new system to be designed
- Predict performance
- Measures systems complexity and user learnability

What is a TASK?

Human actions that contributes to a useful objective, aiming at the system, is a task. Task analysis defines performance of users, not computers. Task Analysis plays an important part in User Requirements Analysis.

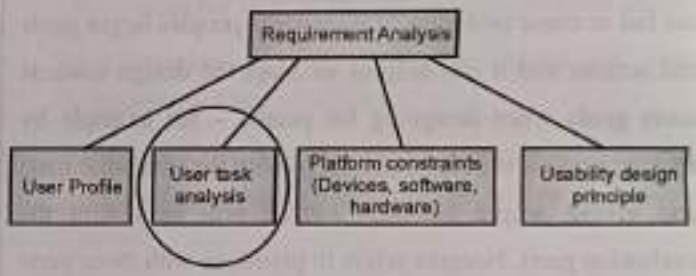


Fig. 3.9.1 : Task Analysis

Task analysis is the procedure to learn the users and abstract frameworks, the patterns used in workflows, and the chronological implementation of interaction with the GUI. It analyzes the ways in which the user partitions the tasks and sequence them.

3.9.2 Task Analysis Techniques

- Task decomposition approach

  Hierarchical Task Analysis (HTA)
- Knowledge based approach: Cognitive Task Analysis

  Procedural knowledge: "how to do it"

UQ. Consider the activity of making a telephone call. Record the actions in an HT diagram or textually. Start off simply, assuming you know the number to dial, but then add more complicated situations : finding the number in an address book, or what to do when the number is engaged.

(SPPU - Q. 10(b), Dec. 19, 9 Marks)

☑ **Ans. :**

As with most of these exercises, this is an open ended question. Here is a simple version with some expansion, but one can look at alternatives such as public pay phones.

0. Make phone call 1. pick up receiver
2. dial number 3. wait for reply
4. talk 5. replace receiver

Plan 0 : 1 –2–3

when answered – 4

when finished – 5

- We now add looking up the number. The form this takes depends on whether we find the number in an address book or a telephone directory.
- If both fail, say if the call is long distance to someone not in a local directory, the telephone operator must be consulted.
- If students have some form of online telephone list, possibly in a palmtop computer, they could include the analysis of the lookup procedure. The next exercise, dictionary lookup, is very similar to looking up a number in a directory, and so further analysis of these branches is not included here.
- Note that contacting the telephone operator involves making a phone call, but the steps for this are not repeated in full!

0. make phone call
1. find number
   - 1.1 look in address book
   - 1.2 look in phone directory
   - 1.3 ask operator
     - 1.3.1 pickup receiver ...
2. Actually call
   - 2.1 pick up receiver 2.2 dial number
   - 2.3 wait for reply 2.4 talk
   - 2.5 replace receiver

Plan 0 : if number unknown – 1

when number found – 2

Plan 1 : if phoning friend – 1.1

if local call – 1.2

if 1.1 or 1.2 fail – 1.3

Plan 2 : 2.1 – 2.2 – 2.3

when answered – 2.4

when finished – 2.5

Finally, we add the case when the phone is engaged. The simplest way to do this is simply to change

Plan 2 : Plan 2: 2.1 – 2.2 – 2.3

if answered — 2.4 then when finished 2.5

if engaged — 2.5

However, looking at the second line it might suggest that we modify 2.4 to have two parts:

0. make phone call ...
2. actually call ...
   - 2.4 successful call
     - 2.4.1 talk
     - 2.4.2 replace receiver
   - 2.5 failed call
     - 2.5.1 replace receiver

Plan 2: 2.1 – 2.2 – 2.3

if answered – 2.4

if engaged – 2.5

Plan 2.4: 2.4 then when finished 2.5

3.9.3 Hierarchical Task Analysis

Hierarchical task analysis (HTA) is a widely used type of Task analysis where a high-level task is decomposed into a hierarchy of subtasks. An HTA is sometimes referred to as a hierarchical decomposition.

Unit III End Sem.

Hierarchical Task Analysis is the procedure of disintegrating tasks into subtasks that could be analyzed using the logical sequence for execution. This would help in achieving the goal in the best possible way. A hierarchical task analysis provides an understanding of the tasks users need to perform to achieve certain goals.

☞ Advantages of HTA

1. HTA provides extensive information about a task (goals, plans, operations), can be highly detailed.
2. HTA can be used for subsequent analyses such as error analysis.
3. HTA can be easily portrayed/interpreted through diagrammatic formats.
4. HTA is a simple and flexible method that does not depend on a methodological context.
5. HTA enables the representation of a task hierarchy that could be further detailed.
6. Although HTA is task oriented and to some extent user oriented it still maintains a strong relationship with traditional software engineering.
7. HTA provides information, inefficiencies in tasks, that can be used for developing product requirements

☞ Disadvantages of HTA

1. HTA can be time and resource intensive.
2. This technique provides more descriptive information than analytical information.
3. There are no strict rules for creating an HTA diagram so different analysts will generate inconsistent hierarchies at varying levels of detail.
4. HTA requires both training and experience. It is not a tool that can be applied immediately.
5. HTA is not a predictive tool. It focuses on existing tasks. HTA diagrams can become quite complex.

☞ Outcome is a hierarchy of

- Tasks
- Sub-tasks
- Actions
- Plans (in what order and under what conditions sub-tasks are performed)
- Based on structure chart notation
- Can be represented textually as well

UQ. In order to clean the house

1. get the vacuum cleaner out
2. fix the appropriate attachment
3. clean the rooms
   - (3.1) clean the hall
   - (3.2) clean the living rooms
   - (3.3) clean the bedrooms
4. Empty the dust bag
5. put the vacuum cleaner and attachments away

Plan 0 : do 1 - 2 - 3 - 5 in that order when the dust bag gets full do 4

Plan 1 : do any of 3.1, 3.2 or 3.3 in any order depending on which rooms need cleaning.

For this HT description of vacuum cleaning, present the same information in a diagrammatic form.

(SPPU - Q. 10(a), Dec. 19, 9 Marks)

☑ **Ans. :**

There are 2 ways to represent Hierarchical Task Analysis – Graphically or in Text:

1. **Graphical Example :** The example below is a Hierarchical Task Analysis graphical representation of vacuum cleaning

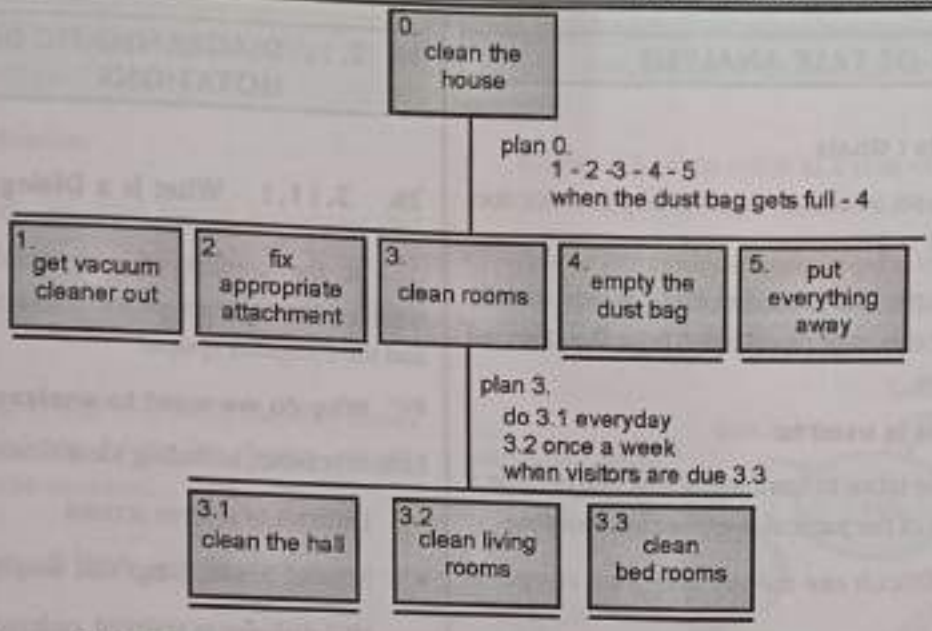


Fig. 3.9.2 : A graphical Hierarchical Task Analysis (HTA) modelling the task of vacuum cleaning

2. **Textual Example :** The above example can be represented using plain text as follows

(0) Clean the house (1) Get vacuum cleaner out (2) Fix appropriate attachment

(3) Clean rooms : 3.1 Clean the hall 3.2 Clean living rooms 3.3 Clean bedrooms

(4) Empty the dust bag

(5) Put everything away

Plan 0. 1 – 2 – 3 – 5. When the dustbag gets full do 4

Plan 3. Do 3.1 every day 3.2 once a week when visitors are due do 3.3

Hierarchical task analysis other example : for make a cup of tea

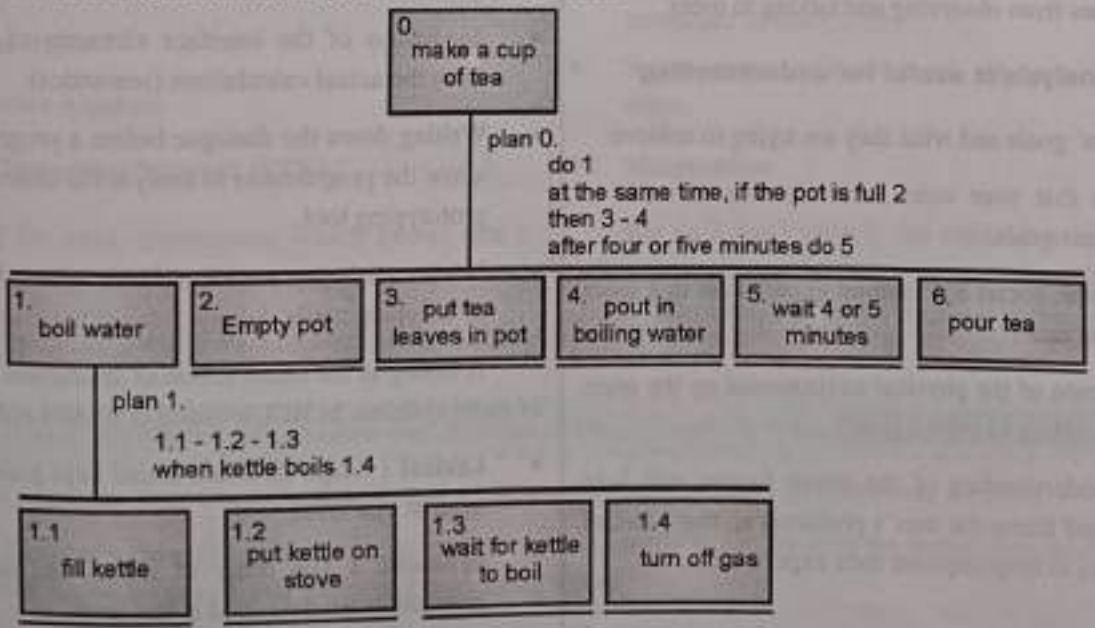


Fig. 3.9.3 : Hierarchical Task Analysis for make a cup of tea

3.10 USES OF TASK ANALYSIS

Task analysis : Goals

- Verify that the set of actions employed by the user does accomplish the task.
- Explicitly describe the procedure that the user actually employs since this may be different from the expected series of actions.

Task analysis is used to

1. Predict the time taken to learn a new task and become a proficient user of the particular application /machine
2. Reveal how difficult one method is to learn compared to another
3. Improve the delivery of information to the user. This involves identifying any problems with the delivery of information to the user and the consideration of possible solutions.
4. Task descriptions (stories, scenarios, use cases) are often used to envision new systems or devices.
5. Task analysis is used mainly to investigate an existing situation
   - An existing system Or a process.
   - How does a user currently do a job?
   - Comes from observing and talking to users.

A task analysis is useful for understanding

- Your users' goals and what they are trying to achieve
- The steps that your users currently take in order to achieve their goals.
- The personal, social and cultural experiences that users bring to the tasks
- The influence of the physical environment on the users while attempting to meet a goal

A clear understanding of the above factors will help you to define and frame the user's problems so that you can then ideate ways to help improve their experiences.

3.11 DIAGRAMMATIC DIALOG DESIGN NOTATIONS

3.11.1 What is a Dialogue?

In the context of user interface design, dialogue signifies the structure of the conversation between the user and the computer system

Why do we want to analyze the dialogue?

Lots of reasons, including identification of:

- Difficult to reverse actions
- Missing items (things that simply can't be achieved)

HCI dialogue is scripted, ordered and structured.

Dialogue Notations

Notations are formal ways of representing the dialogue-structure of the conversation.

Why we need separate dialogue notations?

- Explore, evaluate design options,
  - Discover if flow of events and structure of conversation is logical or too involved or complex.
  - If user has to traverse through too many system sates.
- Separation of the interface elements of the program from the actual calculations (semantics).
- Writing down the dialogue before a program is written allow the programmer to analyse the structure and use a prototyping tool.
- Hence, dialogue notation forms part of the overall prototyping methodology.

A dialog is the construction of interaction between two or more systems. In HCI, a dialog is studied at three levels –

- **Lexical :** Shape of icons, actual keys pressed, etc., are dealt at this level.
- **Syntactic :** The order of inputs and outputs in an interaction are described at this level.

- **Semantic :** At this level, the effect of dialog on the internal application/data is taken care of.

Dialog Representation

To represent dialogs, we need formal techniques that serves two purposes –

- It helps in understanding the proposed design in a better way.
- It helps in analyzing dialogs to identify usability issues. E.g., Questions such as "does the design actually support undo?" can be answered.

3.11.2 Classes of Dialogue Notation

Diagrammatic

- State Transition Networks (STNs)
- Hierarchical STNs
- Flow Charts
- Jackson Structured Design (JSD)
- Petri Nets
- State Charts

Textual

- Grammars
- Production Rules
- Event/Process Algebras

(1) State Transition Network (STN)

STNs are the most spontaneous, which knows that a dialog fundamentally denotes to a progression from one state of the system to the next.

The syntax of an STN consists of the following two entities –

- **Circles :** A circle refers to a state of the system, which is branded by giving a name to the state.
- **Arcs :** The circles are connected with arcs that refers to the action/event resulting in the transition from the state where the arc initiates, to the state where it ends.

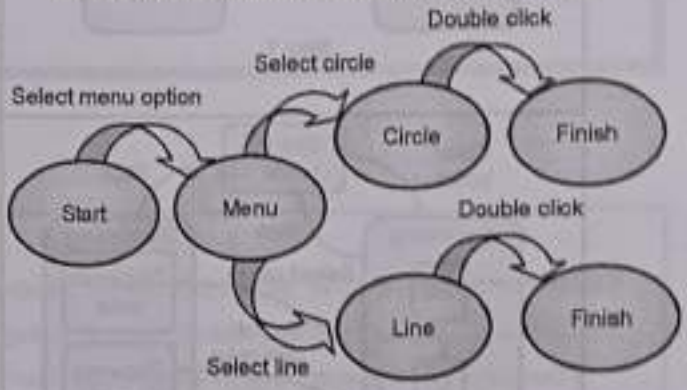


Fig. 3.11.1 : STN Diagram

(2) State Charts

State Charts represent complex reactive systems that extends Finite State Machines (FSM), handle concurrency, and adds memory to FSM. It also simplifies complex system representations. StateCharts has the following states –

- **Active state :** The present state of the underlying FSM.
- **Basic states :** These are individual states and are not composed of other states.
- **Super states :** These states are composed of other states.

Illustration

For each basic state b, the super state containing b is called the ancestor state. A super state is called OR super state if exactly one of its sub states is active, whenever it is active.

UQ. Draw a State Chart diagram of a machine that dispense bottles on inserting coins. **(SPPU-Q. 10(a), Dec. 18, 9 Marks)**

Ans. : Let us see the State Chart Construction of a machine that dispense bottles on inserting coins.

- Fig. 3.11.2 explains the entire procedure of a bottle dispensing machine. On pressing the button after inserting coin, the machine will toggle between bottle filling and dispensing modes.
- When a required request bottle is available, it dispenses the bottle. In the background, another procedure runs where any stuck bottle will be cleared. The 'H' symbol in Step 4, indicates that a procedure is added to history for future access.

Unit III End Sem.

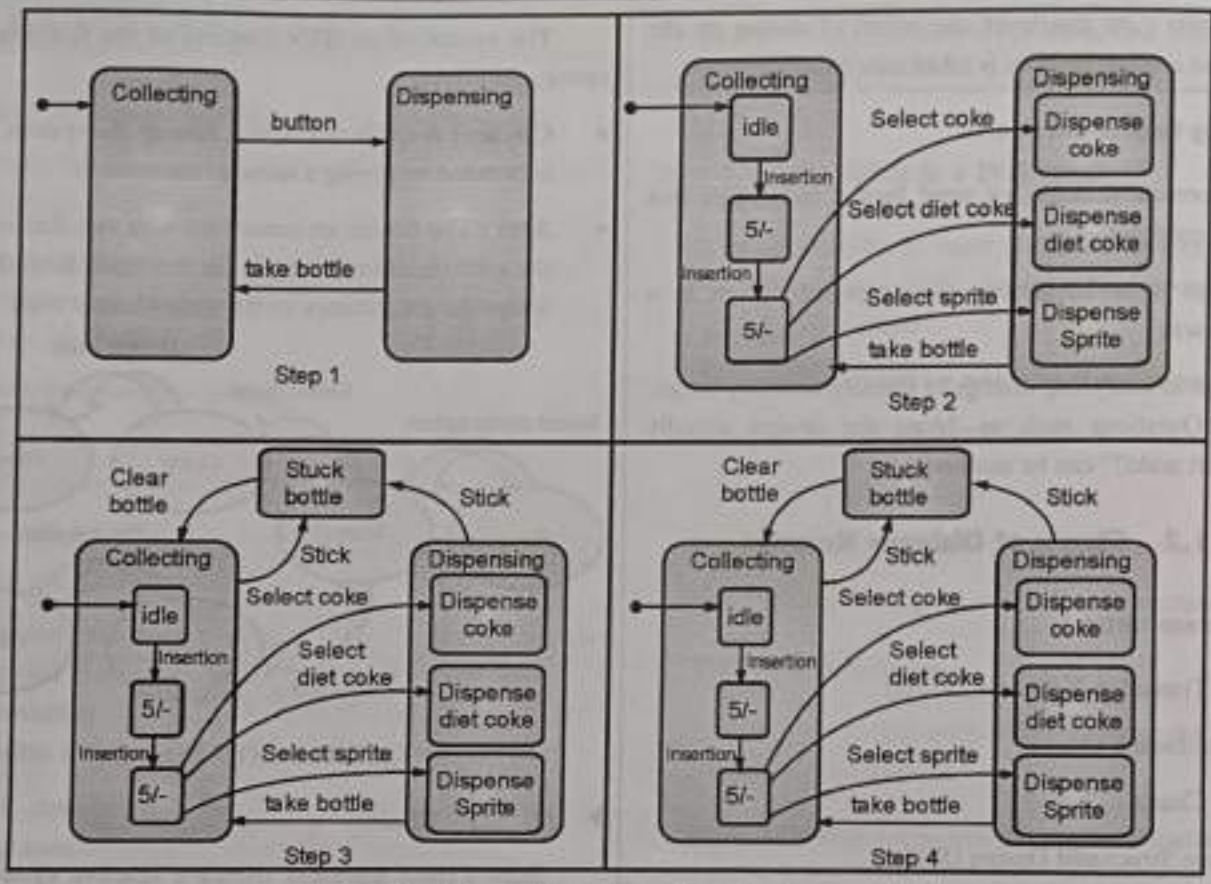


Fig. 3.11.2 : State Chart Construction

(3) Petri Nets

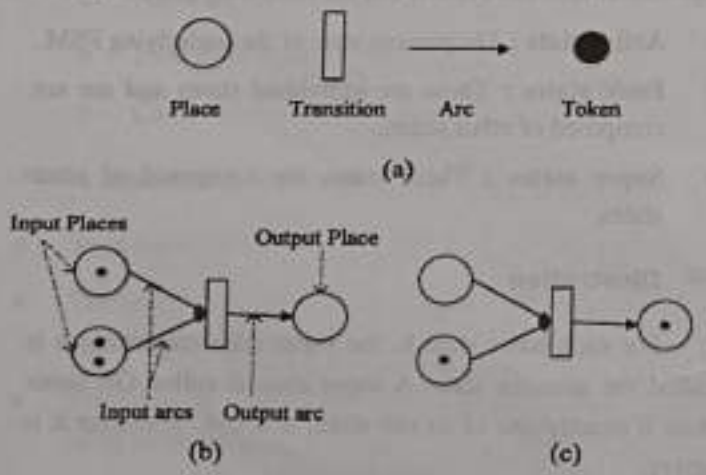


Fig. 3.11.3 : Petri Nets Notations

Petri Net is a simple model of active behavior, which has fourbehavior elements such as – places, transitions, arcs and tokens.

Petri Nets provide a graphical explanation for easy understanding.

- **Place :** This element is used to symbolize passive elements of the reactive system. A place is represented by a circle.
- **Transition :** This element is used to symbolize active elements of the reactive system. Transitions are represented by squares/rectangles.
- **Arc :** This element is used to represent causal relations. Arc is represented by arrows.
- **Token :** This element is subject to change. Tokens are represented by small filled circles.

Chapter Ends...

□□□

UNIT IV

CHAPTER 4

Design Process

Syllabus

Design Rules : Principles that support usability, Design standards, Design Guidelines, What is interaction design?, The software design process, User focus, Scenarios, Navigation Design, Screen Design, Prototyping techniques, Wire-Framing, Understanding the UI Layer and Its Execution Framework, Model-View-Controller(MVC) Framework.

4.1 DESIGN RULES

- Design rules (or usability rules) are rules that a designer can follow in order to increase the usability of the system/product e.g., principles, standards, guidelines.
- Designing for maximum usability is the goal of interactive systems design.
- Abstract principles offer a way of understanding usability in a more general sense, especially if we can express them within some coherent catalog.
- Design rules in the form of standards and guidelines provide direction for design, in both general and more concrete terms, in order to enhance the interactive properties of the system.
- The essential characteristics of good design are often summarized through golden rules or heuristics.
- Design patterns provide a potentially generative approach to capturing and reusing design knowledge.

4.1.1 Usability

UQ. What is the definition of usability as per ISO 9241 standard? Effective applications are both consistent within themselves and consistent with one another. Discuss this in context of Microsoft Office products. **(SPPU - Q. 7(a), Dec. 18, 8 Marks)**

- Usability related to useful and useable. Useful means that the system does what it should. Usable means that it is easy to do it.
- But there is no usability without usefulness and no usefulness without usability. So the distinction is not so clear. This illustrates difficulty with defining usability.
- The difficulty is what is usable to one user may not be to another user. Consider the command line interface for controlling the operating system.
- It is very usable by system administrators, but unusable by common users. So usability should always be considered in context with the user.
- Usability is a measure of how well a specific user in a specific context can use a product/design to achieve a defined goal effectively, efficiently and satisfactorily.
- Designers usually measure a design's usability throughout the development process from wireframes to the final deliverable to ensure maximum usability. A design's usability depends on how well its features accommodate user's needs and contexts.
- The ISO 9241-11 standard on usability describes it as :

  "The extent to which a product can be used by specified users to achieve specified goals, with effectiveness, efficiency and satisfaction in a specified context of use" – ISO 9241 Ergonomics of Human System Interaction
- Usability is hence more than just about whether users can perform tasks easily (ease-of-use). It also deals with user satisfaction for a website to be usable, it has to be engaging and aesthetically pleasing, too.

4.1.2 Goals of Usability

UQ. Describe any four usability goals of Internet Explorer. **(SPPU - Q. 7(a), May 19, 8 Marks)**

1. **Effectiveness :** It supports users in completing actions accurately.
2. **Efficiency :** Users can perform tasks quickly through the easiest process.
3. **Engagement :** Users find it pleasant to use and appropriate for its industry/topic.
4. **Error Tolerance :** It supports a range of user actions and only shows an error in genuine erroneous situations. You achieve this by finding out the number, type and severity of common errors users make, as well as how easily users can recover from those errors.
5. **Ease of Learning :** New users can accomplish goals easily and even more easily on future visits.

When they first encounter an interface, users should be able to find their way about easily enough to achieve objectives without relying on outside/expert knowledge.

Unit IV End Sem.

A design with high usability guides users through the easiest and least labor-intensive route. So, you must leverage a deep understanding of users' contexts. To do that, you must accommodate their limitations, such as their environment, likely distractions and cognitive load.

(1) Effectiveness

- Effectiveness is about the high degree of accuracy under which users can complete their goals. The product has to be able to support the user while performing tasks.
- For example, validating each field of a form accordingly (the postal code field has to be 5 characters long and only contain numbers) and be informative while doing it so, this can reduce data entry errors and help the user finish the task correctly.
- It is also important to choose the right language to communicate and give instructions to the user. The clearer and simpler the language is you're increasing the probability of understandability and also making the right impact on the user.
- Nordstrom, for example, has a category of "Women" clothing that you can find in the main navigation menu, but they also show it with a picture and a link underneath to browse in the same category.

(101)**Fig. 4.1.1 : Effectiveness**

(2) Efficiency

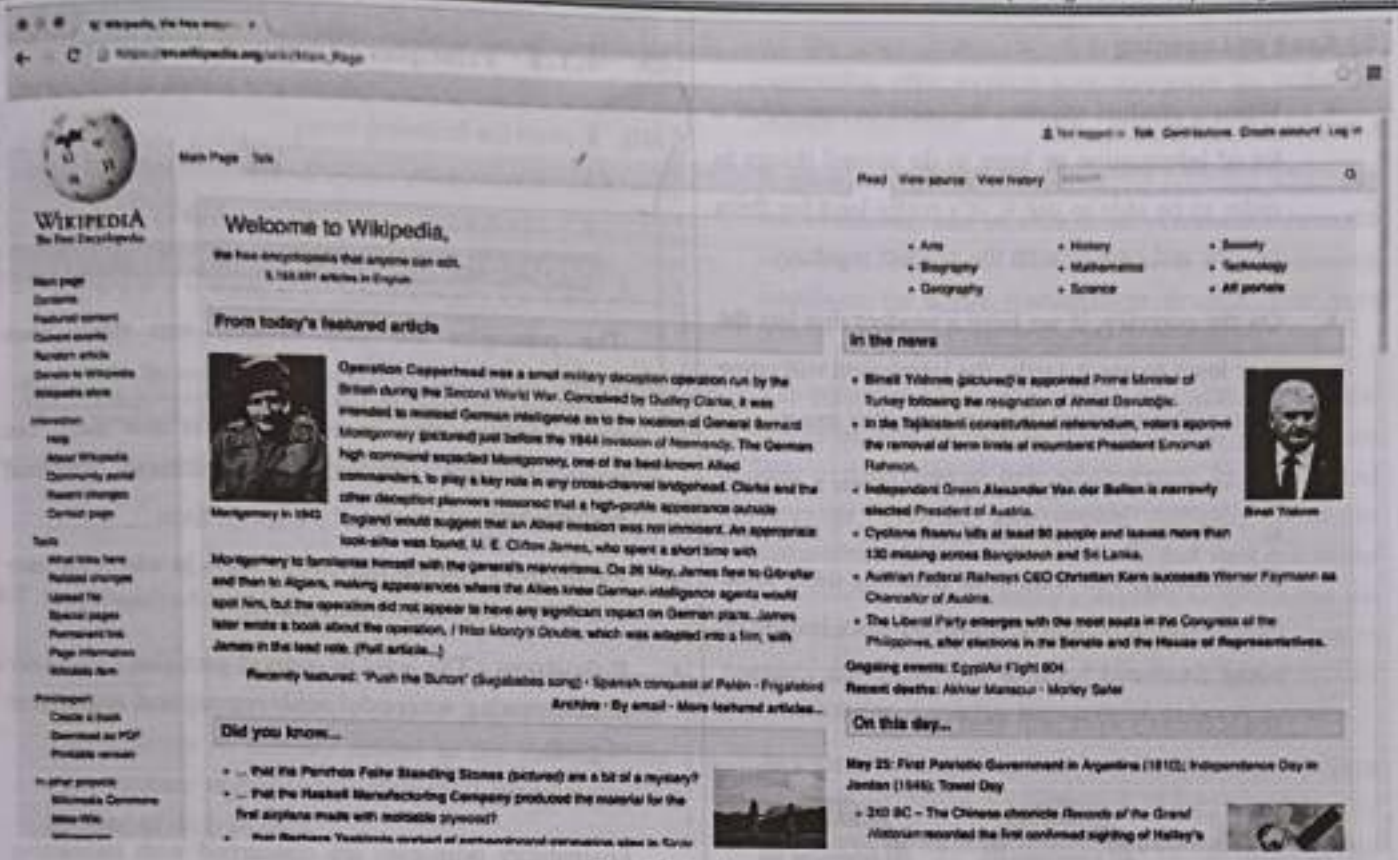
- Efficiency must not be mistaken for effectiveness as they are quite different and our goal is aiming to have both of them. Effectiveness, as we covered above, is about the accuracy of the user to complete a task, while efficiency is how fast can the user finish the task. It's all about speed!
- When you're reviewing efficiency you have to look at the numbers of steps (clicks/keystrokes) needed to achieve the objective. If they can be reduced (without compromising the effectiveness of the task) it can help develop a more efficient process; for example, labelling buttons very clearly so the user can be sure of what to do next or creating shortcuts.

(102)**Fig. 4.1.2 : Efficiency**

- It's also important to understand the environment under which the users are commonly interacting with your product. The navigation, layout, and shortcuts are very different both visually and interactively if they're using a smartphone vs a desktop computer.

(3) Engagement

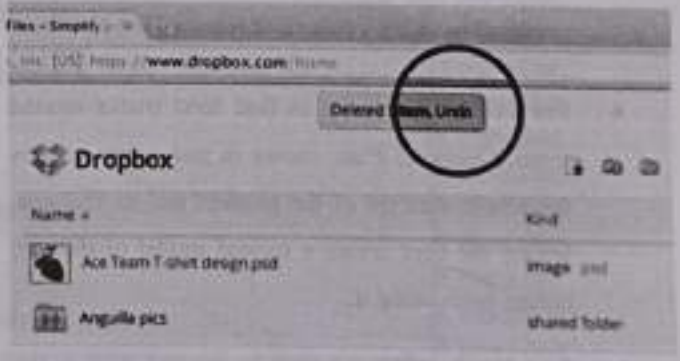
- Engagement happens when the user finds your product enjoyable and satisfactory to use. Yes, aesthetics and great UI elements start to have relevance here, but they're not the only factors implicated in creating a gratifying product that users like to interact with.
- Engagement is not only about looking nice; it's also about looking right. Proper layouts, readable typography and ease of navigation all come together to deliver the right interaction for the user and make it engaging. Looking nice isn't everything, as Wikipedia (famous for its ultra-basic design) proves.



(103)Fig. 4.1.3 : Engagement

(4) Error Tolerance

- Not one single interface ever created can be clean of errors as you can't control the whole ecosystem under which the product is being used and certainly human error is only natural.
- However, what we can do when designing a product, is trying to minimize errors from occurring but if an error does occur make sure the users can quickly and easily recover from it and get back to what they were doing. In Human-Computer Interaction (HCI) this is known as error tolerance.
- Being tolerant to error means to make everything in our power to design a product in which is easy to achieve tasks and without letting the users get confused and do the wrong thing, for example:
- Making all the navigation elements clear and visible so the users can know where they're at and where to go next.
- The right language comes into play again: communicate everything in a simple language.
- The actions performed have to be consistent throughout the product to reduce the probability of mistakes. Limit the options to only correct choices.



(104)Fig. 4.1.4 : Error Tolerance

- Dropbox has an undo function, in case users accidentally delete items in their folders.

Unit IV End Sem

(5) Ease of Learning

- When a product requires the users to remember a lot of information or learn to do several things in order to be able to use it, it's really hard for them to stick and engage with the product regularly.
- On the contrary, if we have a product that lets the user learn to use it easily, the interaction will come as something natural the next time they use it.
- The ease of learning also applies when a product is releasing new features or renewing functionality, you want your returning users to be happy with the improvements you make instead of being frustrated because everything has changed and it doesn't work as it used to.

Fig. 4.1.5 : Ease of Learning

- For example, the app to find food trucks around town, Diamond Plate shows in just a few screens the whole concept of the product and its features, letting the user create a mental model of the app before even using it.
- As a result, when it's time to interact with it, the users can easily navigate and make the best use of the app by reaching their **goals.**

4.1.3 Principles to Support Usability

UQ. Explain the following terms
1. Predictability
2. Synthesizability
3. Familiarity
4. Consistency

(SPPU - Q. 7(a), May 19, Q. 8(b), Dec 18, 8 Marks)

The principles are first divided into three main categories:

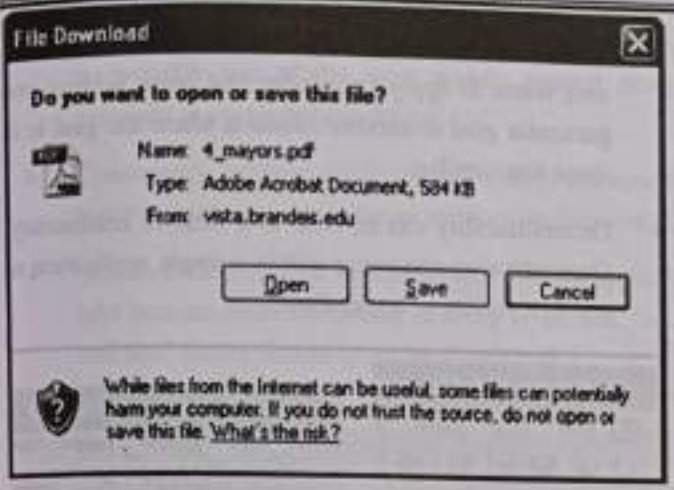
- **Learnability :** The ease with which new users can begin effective interaction and achieve maximal performance.
- **Flexibility :** The multiplicity of ways in which the user and system exchange information.
- **Robustness :** The level of support provided to the user in determining successful achievement and assessment of goals.

1. Learnability

Learnability principles are concerned with interactive system features, which aid novice users to learn quickly and also allows steady progression to expertise. The principles discussed below support the learnability design principle.

Predictability

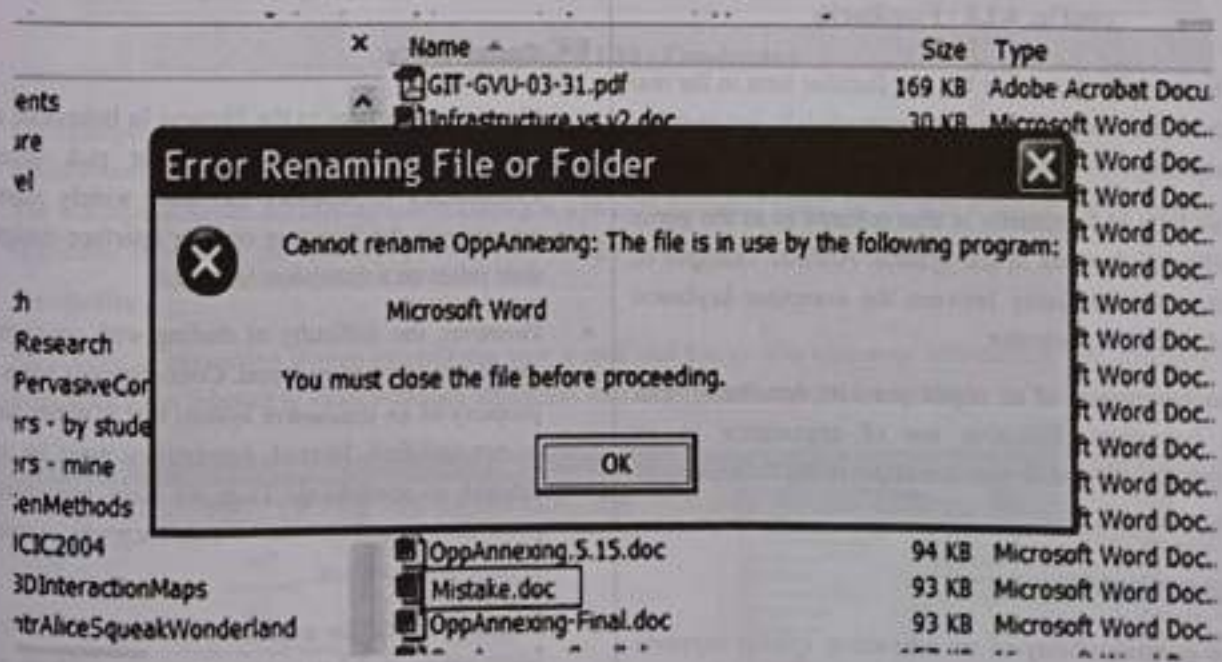
- Predictability of an interactive system is distinguished from deterministic behavior of the computer system alone.
- Most computer systems are ultimately deterministic machines, so that given the state at any one point in time and the operation which is to be performed at that time, there is only one possible state that can result.
- Predictability is a user-centered concept; it is deterministic behavior from the perspective of the user. It is not enough for the behavior of the computer system to be determined completely from its state, as the user must be able to take advantage of the determinism.
- For example, closing a document should always allow the user to save changes not saved already.



(106)Fig. 4.1.6 : Predictability

Synthesizability

- When an operation changes some aspect of the internal state, it is important that the change is seen by the user. The principle of honesty relates to the ability of the user interface to provide an observable and informative account of such change.
- In the best of circumstances, this notification can come immediately, requiring no further interaction initiated by the user.
- At the very least, the notification should appear eventually, after explicit user directives to make the change observable.
- A good example of the distinction between adjacency and eventuality can be seen in the comparison between command language interfaces and visual desktop interfaces for a file management system. You have moved a file from one directory to another.
- The principle of honesty implies that after moving the file to its new location in the file system you are then able to determine its new whereabouts. In a command language system, you would typically have to remember the destination directory and then ask to see the contents of that directory in order to verify that the file has been moved (in fact, you would also have to check that the file is no longer in its original directory to determine that it has been moved and not copied).
- In a visual desktop interface, a visual representation (or icon) of the file is dragged from its original directory and placed in its destination directory where it remains visible (assuming the destination folder is selected to reveal its contents). In this case, the user need not expend any more effort to assess the result of the move operation. The visual desktop is immediately honest.



(107)Fig. 4.1.7 : Synthesizability

Unit IV End Sem.

Familiarity

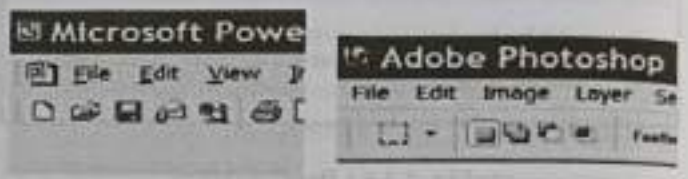
- New users of a system bring with them a wealth of experience across a wide number of application domains. This experience is obtained both through interactions in the real world and through interaction with other computer systems.
- For a new user, the familiarity of an interactive system measures the correlation between the user's existing knowledge and the knowledge required for effective interaction.

(108)Fig. 4.1.8 : Familiarity

- For example, A recycle bin is a familiar item in the real world and recycle bin icon immediately suggests its function.
- This type of familiarity is also referred to as the guess ability of features in the system. Another example of this is the similarity between the computer keyboard and that of a typewriter.
- The appearance of an object provides familiarity with its behaviour. Effective use of appearance in an interactive system design can improve the familiarity of the system.

Generalizability

- The generalizability of an interactive system supports this activity, leading to a more complete predictive model of the system for the user.
- We can apply generalization to situations in which the user wants to apply knowledge that helps achieve one particular goal to another situation where the goal is in some way similar.
- Generalizability can be seen as a form of consistency. Generalization can occur within a single application or across a variety of applications.



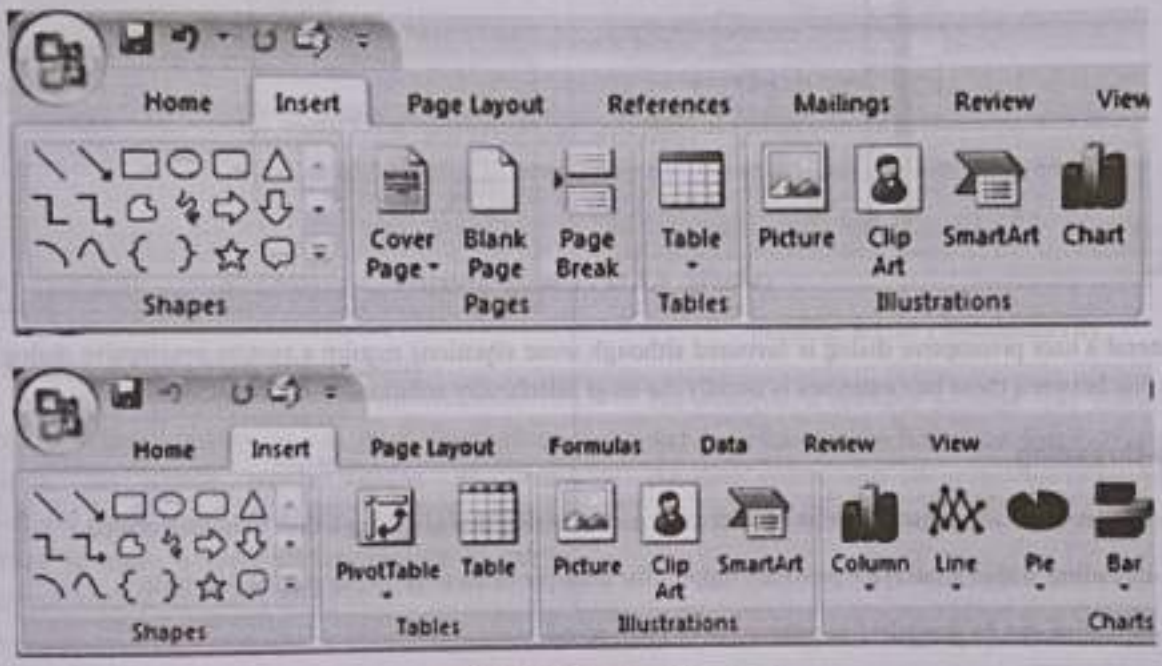
(109)Fig. 4.1.9 : Generalizability

- For example, in a graphical drawing package that draws a circle as a constrained form of ellipse, we would want the user to generalize that a square can be drawn as a constrained rectangle.
- A good example of generalizability across a variety of applications can be seen in multi-windowing systems that attempt to provide cut/paste/copy operations to all applications in the same way (with varying degrees of success). Generalizability within an application can be maximized by any expert designer.

Consistency

- Consistency relates to the likeness in behaviour arising from similar situations or similar task objectives. Consistency is probably the most widely mentioned principle in the literature on user interface design. The user relies on a consistent interface.
- However, the difficulty of dealing with consistency is that it can take many forms. Consistency is not a single property of an interactive system that is either satisfied or not satisfied. Instead, consistency must be applied relative to something. Thus we have consistency in command naming, or consistency in command/ argument invocation.
- Consistency can be expressed in terms of the form of input expressions or output responses with respect to the meaning of actions in some conceptual model of the system.

- To support generalisability, consistency is essential and is probably one of the most widely applied design principle in user interface design.
- Consistency between application is always favourable, however consistency within an application is essential.
- Standard GUI design factors should aid designers to take into account consistency at every level, and, "look and feel" issues should never be abandoned.
- The use of labels and icons should always be consistent and the same icons and labels should mean the same thing.
- The principle of "sameness" should be applied to the use of terminology, formatting and input/output behaviour arising from similar situations or task objectives.

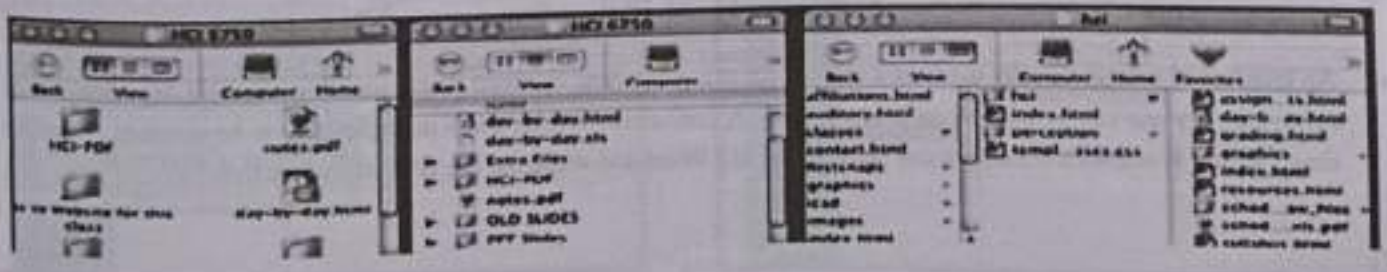


(1010)**Fig. 4.1.10 : Consistency**

- For example, before the introduction of explicit arrow keys, some word processors used the relative position of keys on the keyboard to indicate directionality for operations (for example, to move one character to the left, right, up or down). The conceptual model for display-based editing is a two-dimensional plane, so the user would think of certain classes of operations in terms of movements up, down, left or right in the plane of the display.

2. Flexibility

Flexibility in interactive design extends the way a user and the system exchange information. By applying flexibility principles to an interactive system design, designers aim to improve a system's usability.



(1011)**Fig. 4.1.11 : Flexibility**

Dialog Initiative

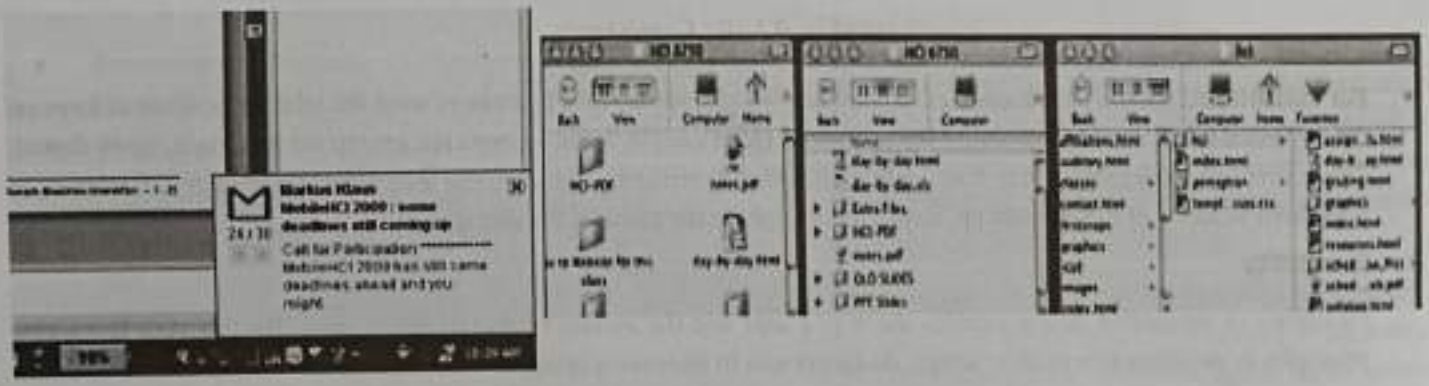
- When the system controls the dialog flow, the dialog is said to be system preemptive. Conversely,
- when the flow is controlled by the user, the dialog is said to be user preemptive.

(1D12)**Fig. 4.1.12 : Dialog initiative**

- In general a user preemptive dialog is favoured although some situations require a system preemptive dialog. In reality some line between these two extremes is usually the most satisfactory solution.

Multi-threading

- Within a user interface a thread can be considered a part of dialog that allowing a task to be performed.
- Multi-threading within a interface provides support for multiple tasks to be performed at one time.
- Multi-threading can be described as concurrent or interleaved.

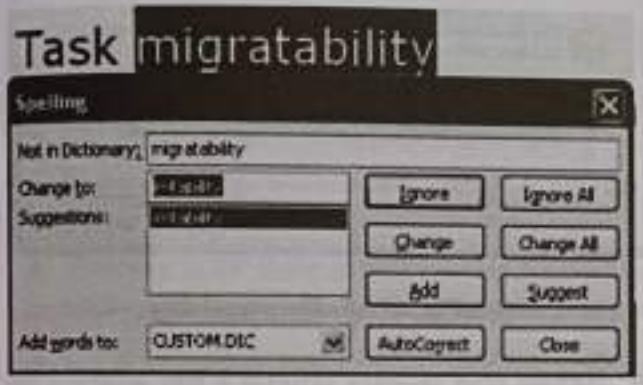


(1D13)**Fig. 4.1.13 : Multi-threading**

- An interleaved system permits work on a single task at a given time - a word processor allow multiple documents to be open, but only one can be worked on at any instant. A concurrent system allow multiple tasks to be actioned at a given time - within Windows a document can be edited in MS Word and while the file find utility is active.

Task migratability

- Task migratability means passing responsibility of execution of tasks between user and system. A computerised spell checker is a good example to this. It is a waste of time for a user to manually check a very long document and correct.

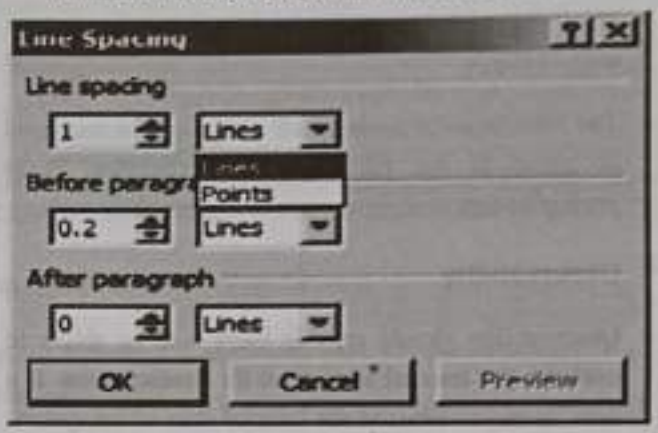


(1D14)**Fig. 4.1.14 : Task migratability**

- A spell checking facility in a word processing application can check words against its own computerised dictionary. However, it is considered 'dangerous' to allow a spellchecker program to carry out this task without the user's assistance.

Substitutivity

- Substitutivity offers a user alternative ways of specifying input or viewing output. Indeed the distinction between output and input can be blurred.

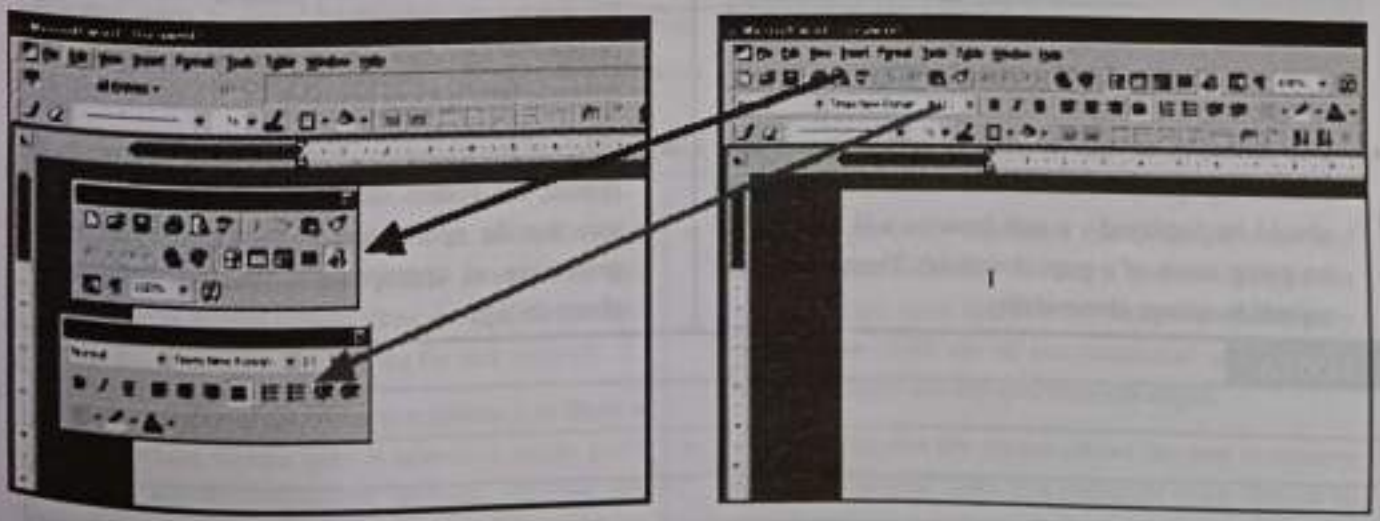


(1D15)**Fig. 4.1.15 : Substitutivity**

- For example, a drawing package may allow start and end co-ordinates of a line to be specified, conversely, the same system may allow the line to be drawn first, and the system indicates the end point co-ordinates.

Customizability

- The user interface should be able to support individual preferences. For example standard control bars in MS Word can be amended as required. The customizability principle supports a user's ability to adjust systems settings or features to a form that best suites the preferred way of usage.



(1D16)**Fig. 4.1.16 : Customizability**

Unit IV End Sem.

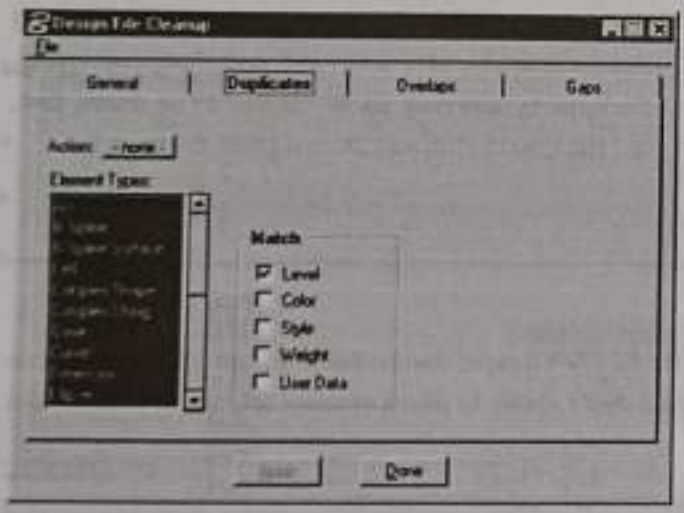
- Adaptivity can be automated but in order to be able to provide such user-centred system behaviours the system should be trained to distinguish an expert's behaviour from a novice user's behaviour. Repetitive tasks can be detected by observing a user's behaviour and macros can be automatically constructed.

3. Robustness

The robustness of an interface design can be measured in terms of the following four principles. These principles aim to support users to achieve their goals.

☞ Observability

- Observability should provide users with an ability to evaluate the internal state from its representation. If a user cannot understand the internal state of the system, there is a high likelihood that the user's confidence will be very low.

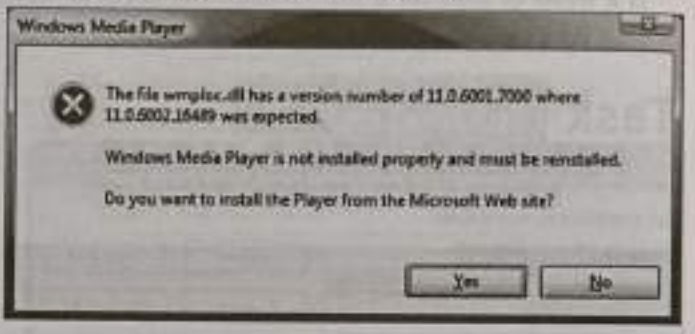


(1D17)**Fig. 4.1.17 : Observability**

- For example, if the system is performing a time consuming operation, the current status of the operation should be displayed - a web browser will indicate the on-going status of a page download. There are several aspects to system observability.

☞ Recoverability

- Users should be able to reach a desired goal after recognition of errors in previous interaction. Error recovery can be achieved in two ways, forward (negotiation) and backward (undo).



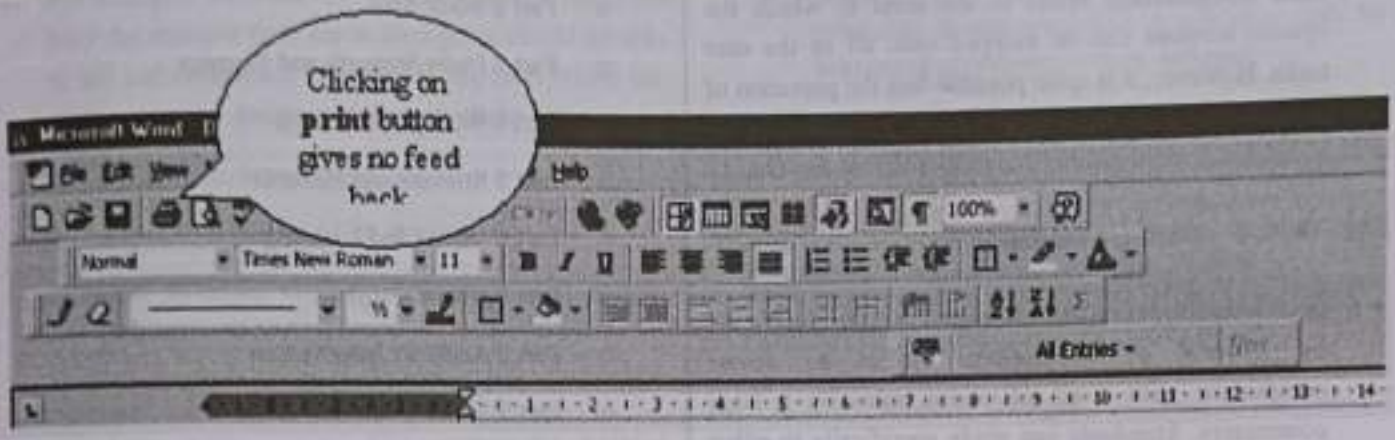
(1D18)**Fig. 4.1.18 : Recoverability**

- Forward error recovery involves a user accepting the current state of the system and negotiating from the present state towards the required state.
- A backward error recovery mechanism within a system allows a user to undo the undesired outcome of the previous interaction by returning to a previous state. In addition to providing the ability to recover forward or backward, the effort to achieve this should reflect the work being done - the commensurate effort.

☞ Responsiveness

- Responsiveness is usually measured in terms of the rate of communication between the system and a user. Response time, indicating change of states within the system, is important. Short duration or instantaneous response time is more desirable.
- When an instantaneous response cannot be given by the system, the system should be able to indicate to the user that the system has received the request and in processing an appropriate action (see definition of observability).

NOTES

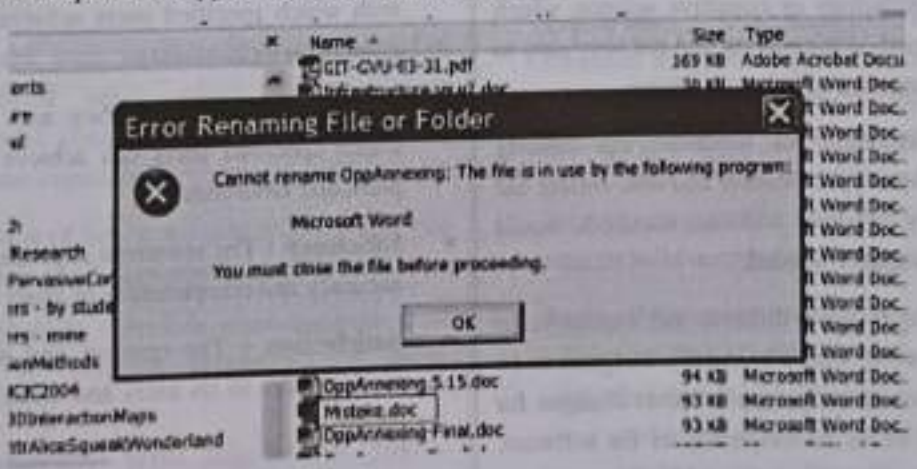


(1019)**Fig. 4.1.19 : Responsiveness**

- As illustrated in the above picture, clicking the print icon on the Microsoft Word menu does not give feedback, e.g. especially novice users may end up pressing the print icon a number of times.
- As well as response time of a system, consistency, of responsiveness is also desirable.

Task conformance

- There are two aspects of task conformance, task completeness and task adequacy. Task completeness is concerned with whether a system is capable of supporting the entire task that a user wishes to perform.



(1020)**Fig. 4.1.20 : Task conformance**

- The task adequacy is concerned with addressing the user's understanding of these tasks It is necessary that an interactive system should allow the user to perform the desired tasks as defined during the task analysis.
- Since the purpose of an interactive system is to allow a user to perform various tasks in achieving certain goals within a specific application domain, we can ask whether the system supports all of the tasks of interest and whether it supports these as the user wants.
- Task completeness addresses the coverage issue and task adequacy addresses the user's understanding of the tasks. It is not sufficient that the computer system fully implements some set of computational services that were identified at early specification stages.
- It is essential that the system allows the user to achieve any of the desired tasks in a particular work domain as identified by a task analysis that precedes system specification.

Unit IV End Sem.

- Task completeness refers to the level to which the system services can be mapped onto all of the user tasks. However, it is quite possible that the provision of a new computer based tool will suggest to a user some tasks that were not even conceivable before the tool.

4.1.4 Design Standards

- Standards for interactive system design are usually set by national or international bodies to ensure compliance with a set of design rules by a large community. Standards can apply specifically to either the hardware or the software used to build the interactive system. Smith points out the differing characteristics between hardware and software, which affect the utility of design standards applied to them:
- Underlying theory Standards for hardware are based on an understanding of physiology or ergonomics/human factors, the results of which are relatively well known, fixed and readily adaptable to design of the hardware. On the other hand, software standards are based on theories from psychology or cognitive science, which are less well formed, still evolving and not very easy to interpret in the language of software design.
- Consequently, standards for hardware can directly relate to a hardware specification and still reflect the underlying theory, whereas software standards would have to be more vaguely worded.
- Change Hardware is more difficult and expensive to change than software, which is usually designed to be very flexible. Consequently, requirements changes for hardware do not occur as frequently as for software. Since standards are also relatively stable, they are more suitable for hardware than software.
- A given standards institution, such as the British Standards Institution (BSI) or the International Organization for Standardization (ISO) or a national military agency, has had standards for hardware in place before any for software. For example, the UK Ministry of Defence has published an Interim Defence Standard 00–25 on Human Factors for Designers of Equipment, produced in 12 parts :
  - Part 1 Introduction
  - Part 2 Body Size
  - Part 3 Body Strength and Stamina
  - Part 4 Workplace Design
  - Part 5 Stresses and Hazards
  - Part 6 Vision and Lighting
  - Part 7 Visual Displays
  - Part 8 Auditory Information
  - Part 9 Voice Communication
  - Part 10 Controls
  - Part 11 Design for Maintainability
  - Part 12 Systems
- One component of the ISO standard 9241, pertaining to usability specification, applies equally to both hardware and software design. In the beginning of that document, the following definition of usability is given:
- Usability The effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments.
- **Effectiveness:** The accuracy and completeness with which specified users can achieve specified goals in particular environments.
- **Efficiency :** The resources expended in relation to the accuracy and completeness of goals achieved.
- **Satisfaction :** The comfort and acceptability of the work system to its users and other people affected by its use.

4.1.5 Design Guidelines

- Design guidelines are sets of recommendations on how to apply design principles to provide a positive user experience. Designers use such guidelines to judge how to adopt principles such as intuitiveness, learnability, efficiency and consistency so they can create compelling designs and meet and exceed user needs.
- Cognitive psychologist's conclusions from their studies have supplied the groundwork for many design principles over the years. Other design rules exist simply because they are common sense.

- For example, users can tell when a webpage looks too busy the moment they see it. Designers should be able to tell and understand why this is the case. There are various types of design guidelines, including these.
  - Style – e.g., brand logos, colors.
  - Layout – e.g., grid or list structure
  - User interface (UI) components – e.g., menus, buttons
  - Text – e.g., font, tone, labels/fields
  - Accessibility – e.g., Aria markup for disabled users
  - Design Patterns – e.g., forms
- Design guidelines are rules of thumb for you to create work which never frustrates users. Likewise, you should also cater to users who have a wide range of disabilities. How you apply design guidelines also depends on the contexts of use, your design's platform and the type of interaction users will have with it (e.g., voice-controlled).
- Industry pioneers such as Don Norman and Jakob Nielsen identified areas which designers and developers should consider to design products that offer the best user experience.
- Here's an example of how a designer might realize one of Jakob Nielsen's ten design principles.
  - **Design principle :** Provide plain-language error messages to pinpoint problems and likely solutions.
  - **Design Guideline:** Write large-lettered, jargon-free text in web-safe font. Use short sentences and draw users' attention to causes and remedies.
  - **Design rule :** Use 20-pt, black Georgia on lavender background (#e6e6fa Hex). Put instructions in bold.
  - **Take note of the distinctions.** The principles are general points of guidance. The guidelines show you how to deal with them. The rules are written in a straightforward manner. As a result, the designer starts with the design principles and then determines the design rules using design standards.
  - When creating things, designers frequently use subjective design principles. A guideline may be interpreted differently by each designer.

4.2 WHAT IS INTERACTION DESIGN?

UQ. What is design? What is the golden rule of design? Illustrate the process of interaction design.
(SPPU - Q. 5(b), May 18, Q. 5(b), Dec. 18, 8 Marks)

4.2.1 What is Design?

Definition : A simple definition is achieving goals within constraints.

- **Goals :** What is the purpose of the design we are intending to produce? Who is it for? Why do they want it? For example, if we are designing a wireless personal movie player, we may think about young affluent users wanting to watch the latest movies whilst on the move and download free copies, and perhaps wanting to share the experience with a few friends.
- **Constraints :** What materials must we use? What standards must we adopt? How much can it cost? How much time do we have to develop it? Are there health and safety issues? In the case of the personal movie player: does it have to withstand rain? Must we use existing video standards to download movies? Do we need to build in copyright protection?
- Trade-off Choosing which goals or constraints can be relaxed so that others can be met. For example, we might find that an eye-mounted video display, a bit like those used in virtual reality, would give the most stable image whilst walking along. However, this would not allow you to show friends, and might be dangerous if you were watching a gripping part of the movie as you crossed the road.

The golden rule of design

- The designs we produce may be different, but often the raw materials are the same. This leads us to the golden rule of design: understand your materials understand computers limitations, capacities, tools, platforms understand people psychological, social aspects, human error.

Unit IV End Sem.

- HCI design is considered as a problem solving process that has components like planned usage, target area, resources, cost, and viability. It decides on the requirement of product similarities to balance trade-offs.
- The following points are the four basic activities of interaction design :
  - Identifying requirements
  - Building alternative designs
  - Developing interactive versions of the designs
  - Evaluating designs
- Three principles for user-centered approach are :
  - Early focus on users and tasks
  - Empirical Measurement
  - Iterative Design
  - Design Methodologies
- Various methodologies have materialized since the inception that outline the techniques for human–computer interaction.
- Following are few design methodologies :
  - **Activity Theory :** This is an HCI method that describes the framework where the human-computer interactions take place. Activity theory provides reasoning, analytical tools and interaction designs.
  - **User-Centered Design :** It provides users the center-stage in designing where they get the opportunity to work with designers and technical practitioners.
  - **Principles of User Interface Design :** Tolerance, simplicity, visibility, affordance, consistency, structure and feedback are the seven principles used in interface designing.
  - **Value Sensitive Design :** This method is used for developing technology and includes three types of studies – conceptual, empirical and technical.
  - **Conceptual investigations** works towards understanding the values of the investors who use technology.
- Empirical investigations are a qualitative or quantitative design research study that shows the designer understands of the user's values.
- Technical investigations contain the use of technologies and designs in the conceptual and empirical investigations.

4.2.2 Golden Rules and Heuristics in Design

Shneiderman's Eight Golden Rules of Interface Design

- They are intended to be used during design but can also be applied, like Nielsen's heuristics, to the evaluation of systems. Strive for consistency in action sequences, layout, terminology, command use and so on.
- Enable frequent users to use shortcuts, such as abbreviations, special key sequences and macros, to perform regular, familiar actions more quickly.
- Offer informative feedback for every user action, at a level appropriate to the magnitude of the action. Design dialogs to yield closure so that the user knows when they have completed a task.
- 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.
- 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. Support internal locus of control so that the user is in control of the system, which responds to his actions.
- Reduce short-term memory load by keeping displays simple, consolidating multiple page displays and providing time for learning action sequences.
- Norman's Seven Principles for Transforming Difficult Tasks into Simple Ones

(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. One is to provide mental aids to help the user keep track of stages in a more complex task. Another is to use technology to provide the user with more information about the task and better feedback. A third approach is to automate the task or part of it, as long as this does not detract from the user's experience. The final approach to simplification is to change the nature of the task so that it becomes something simpler. In all of this, it is important not to take control away from the user.

(3) Make things visible: bridge the gulfs of execution and evaluation. The interface should make clear what the system can do and how this is achieved, and should enable the user to see clearly the effect of their actions on the system.

(4) Get the mappings right. User intentions should map clearly onto system controls. User actions should map clearly onto system events. So it should be clear what does what and by how much. Controls, sliders and dials should reflect the task – so a small movement has a small effect and a large movement a large effect.

(5) Exploit the power of constraints, both natural and artificial. Constraints are things in the world that make it impossible to do anything but the correct action in the correct way. A simple example is a jigsaw puzzle, where the pieces only fit together in one way. Here the physical constraints of the design guide the user to complete the task.

(6) Design for error. To err is human, so anticipate the errors the user could make and design recovery into the system.

(7) When all else fails, standardize. If there are no natural mappings then arbitrary mappings should be standardized so that users only have to learn them once. It is this standardization principle that enables drivers to get into a new car and drive it with very little difficulty key controls are standardized. Occasionally one might switch on the indicator lights instead of the windscreen wipers, but the critical controls (accelerator, brake, clutch, steering) are always the same.

4.2.3 Interaction Design

UQ. Write short note on process on Interaction design with respect to following points : Basic activities Characteristics. **(SPPU - Q. 6(a), May 19, 8 marks)**

- Interaction Design is the design of interactive products and services in which a designer's focus goes beyond the item in development to include the way users will interact with it.
- Thus, close scrutiny of user's needs, limitations and contexts, etc. empowers designers to customize output to suit precise demands.
- Interaction design focuses on creating engaging interfaces with well thought out behaviors. Understanding how users and technology communicate with each other is fundamental to this field.
- With this understanding, you can anticipate how someone might interact with the system, fix problems early, as well as invent new ways of doing things.

(1021)**Fig. 4.2.1 : Interaction design**

Unit IV End Sem.

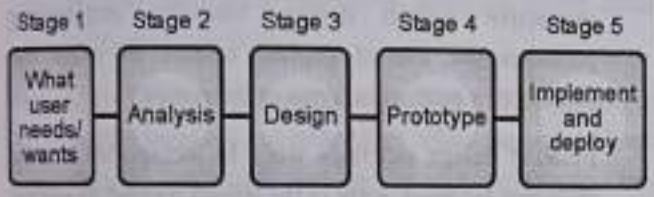
- Interaction design is about creating interventions in often complex situations using technology of many kinds including PC software, the web and physical devices.
- Interaction Design Activities involves :
  1. Achieving goals within constraints and trade-off between these
  2. Understanding the raw materials: computer and human
  3. Accepting limitations of humans and of design.
- The design process has several stages and is iterative and never complete.
- Interaction starts with getting to know the users and their context :
  - Finding out who they are and what they are like probably not like you!
  - Talking to them, watching them.
  - Scenarios are rich design stories, which can be used and reused throughout design:
  - They help us see what users will want to do.
  - They give a step-by-step walkthrough of user's interactions: including what they see, do and are thinking.
  - Users need to find their way around a system. This involves:
  - Helping users know where they are, where they have been and what they can do next
  - Creating overall structures that are easy to understand and fit the user's needs
  - Designing comprehensible screens and control panels.
  - Complexity of design means we don't get it right first time:
- So we need iteration and prototypes to try out and evaluate
- But iteration can get trapped in local maxima; designs that have no simple improvements, but are not good theory and models can help give good start points :
  - Designer's work in Interaction Design involves five dimensions words (1D), visual representations (2D), physical objects/space (3D), time (4D), and behavior (5D).
  - Interaction Design's five dimensions were first defined by a professor at London's Royal College of Art, Gillian Crampton Smith, and a senior interaction designer, Kevin Silver.
  - The dimensions represent the aspects an interaction designer considers when designing interactions:
  - Words (1D) encompass text, such as button labels, which help give users the right amount of information.
  - Visual representations (2D) are graphical elements such as images, typography and icons that aid in user interaction.
  - Physical objects/space (3D) refers to the medium through which users interact with the product or service—for instance, a laptop via a mouse, or a mobile phone via fingers.
  - Time (4D) relates to media that changes with time, such as animations, videos and sounds.
  - Behavior (5D) is concerned with how the previous four dimensions define the interactions a product affords for instance, how users can perform actions on a website, or how users can operate a car. Behavior also refers to how the product reacts to the user's inputs and provides feedback.
- Interaction designers utilize all five dimensions to consider the interactions between a user and a product or service in a holistic way. Specifically, we use them to help envision the real-world demands of a usership in relation to a design not yet introduced.
- For example, designers of an app that must process data at high speed in order to find results inside a mass-transit system (a subway/metro) will face accommodating the constraints of underground commuters – cramped spaces, fast journeys, dead zones, etc.

What is Interaction Design Process?

UQ. Illustrate the Interaction Design Process.
(SPPU - Q. 7(a), May 19, 8 Marks)

- The interaction design process is what designers use to create solutions centered on user's needs, aims and behavior when interacting with products. The interaction design process involves 5 stages: discovering what users need/want, analyzing that, designing a potential solution, prototyping it and implementing and deploying it.
- Here are the five stages that the Interactive Design Process typically involves :

General Flow of Interaction Design Process



(1022)**Fig. 4.2.2 : Interactive Design Process**

1. Find the users' needs/wants : It's easy to assume you know what users want/need and their relevant contexts. Discover their real requirements:
   (a) Observe people.
   (b) Interview people.
   (c) Examine existing solutions while remembering it's hard to envisage future needs, technologies, etc.
2. Do analysis to sort and order your findings so they make sense. This may be through a :
   (a) Narrative/story of how someone uses a system.
   (b) Task analysis, breaking down a user's steps/sub-steps.
3. Design a potential solution according to design guidelines and fundamental design principles (e.g., giving appropriate feedback for users' actions). Use the best techniques to match how users will interact with it in terms of, for example, navigation.
4. Start prototyping : Give users an idea of what the product will look like and let them test it, and/or give it to experts to evaluate its effectiveness using heuristics.
5. Implement and deploy what you have built. It's important to understand the interaction design process is a general idea of how you can start from your users' needs and progress towards a fitting solution

Characteristics of Interaction Design

1. Design must be Goal-Driven

- One of the major practices of interaction design is that it is goal-driven design. Interaction designers therefore need to know how to build the customer insights into their design, regardless of whether or not they are personally conducting user research.
- The following UX processes are good starting points to empathize with users in order to elicit the best design response :
  - **User Personas :** Based on the psychologies and behaviors of the target audience you need to create user personas and refer them at the time of making crucial design decisions.
  - **User Scenarios :** This helps to explain how a user persona will act when using the product. Designers need to explore the context where the user interacts with the product.
  - **User Experience Maps :** Also known as journey maps, user experience maps record all the conditions of a single interaction. It also includes external circumstances and emotion.
- This approach helps interaction designers to first acknowledge their constraints before devising a solution. You are considering the goals users have in mind when using your product and design it accordingly.

2. It must be Easy to Use

- This is the bare minimum for any product. If your product lacks usability, it is obvious that no one will desire it. It is the ease with which someone uses your product to achieve their desired goal. Just like UX

Unit IV End Sem.

design, interaction design needs to consider the inherent usability of the interfaces to make the underlying system to comprehend and use.

- For example, if you are designing an online movie and events ticketing app with a high level of detail where users can determine rows, seat numbers etc. The app typically needs to consolidate a multi-step and multi-program process and transform the whole experience into a single linear path.
- The usability needs to be effortless if you want people to use it extensively. In case they need to invest a lot of time just to understand the system, chances are they are going to abandon your app. The best practice would be something like:
- Open the app → Select a show/event → Select date & place → Select row & seat → Payment and checkout

3. The Interface must be Easily Learned

- You need to design intuition and familiarity into every interface as users don't really remember all functions after using a product. In order to boil down complexity, you need to create consistency and predictability. A simple example is that when a designer uses a lightbox for some images and have others opening in a new tab. This breaks both consistency and predictability and would only confuse users, if not annoying them.
- You need to maintain consistency throughout the design to create predictability, which in turn improves learnability.

4. Keep an Eye on Signifiers & Affordances

- It is important to focus on signifiers as well. Your website menu should look like a menu, otherwise it will leave your users confused. Follow the norm. Without it, users will find it difficult to perceive the affordance.
- Signifiers basically work as metaphors, telling people why they should interact with a particular item by communicating the purpose. Affordances, on the other hand, define the relation between the user and their environment. What is the underlying functionality of your product? This is what affordance explains. Designers need to utilize signifiers and affordances consistently throughout the design.

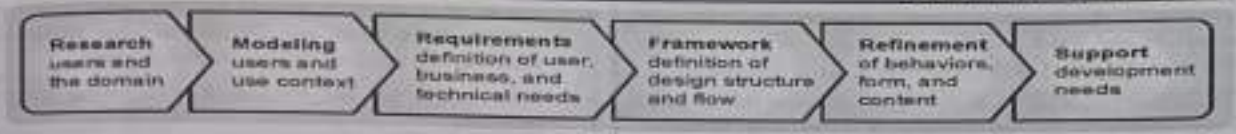
5. Feedback and Response Time

- Finally, we have feedback and response time as one of the "good" characteristics of interaction design. Feedback is key to interaction design.
- Your product must communicate with your users (provide feedback) if the desired task was accomplished and what they need to do next.

4.3 THE SOFTWARE DESIGN PROCESS

The Software Process

- Software engineering provides a means of understanding the structure of the design process, and that process can be assessed for its effectiveness in interactive system design. Usability engineering promotes the use of explicit criteria to judge the success of a product in terms of its usability.
- Iterative design practices work to incorporate crucial customer feedback early in the design process to inform critical decisions which affect usability.Design involves making many decisions among numerous alternatives. Design rationale provides an explicit means of recording those design decisions and the context in which the decisions were made
- In the development of a software product, we consider two main parties : The customer who requires the use of the product and the designer who must provide the product.
- Typically, the customer and the designer are groups of people and some people can be both customer and designer. It is often important to distinguish between the customer who is the client of the designing company and the customer who is the eventual user of the system. These two roles of customer can be played by different people. The group of people who negotiate the features of the intended system with the designer may never be actual users of the system. This is often particularly true of web applications. Here the term "customer" to refer to the group of people who interact with the design team and we will refer to those who will interact with the designed system as the user or end-user.



(10220)Fig. 4.3.1 : Software Design Process

- Software design process can be roughly divided into six phases: Research, Modeling, Requirements Definition, Framework Definition, Refinement, and Support.

These phases follow the five component activities of interaction design i.e. understanding, abstracting, structuring, representing, and detailing - with a greater emphasis on modeling user behaviors and defining system behaviors.

(1) Research

- The Research phase employs ethnographic field study techniques (observation and contextual interviews) to provide qualitative data about potential and/or actual users of the product.
- It also includes competitive product audits, reviews of market research and technology white papers and brand strategy, as well as one-on-one interviews with stakeholders, developers, subject matter experts (SMEs), and technology experts as suits the particular domain.

(2) Modeling

- During the Modeling phase, behavior and workflow patterns discovered through analysis of the field research and interviews are synthesized into domain and user models. Domain models can include information flow and workflow diagrams.
- User models, or personas, are detailed, composite user archetypes that represent distinct groupings of behaviors, attitudes, aptitudes, goals, and motivations observed and identified during the Research phase.

(3) Requirements Definition

- Design methods employed by teams during the Requirements Definition phase provide the much-needed connection between user and other models and the framework of the design.
- This phase employs scenario-based design methods with the important innovation of focusing the scenarios not on user tasks in the abstract, but first and foremost on meeting the goals and needs of specific user personas.
- Personas become the main characters of these scenarios, and the designers explore the design space via a form of role-playing.

(4) Framework Definition

- In the Framework Definition phase, designers create the overall product concept, defining the basic frameworks for the product's behavior, visual design, and - if applicable - physical form.
- Interaction design teams synthesize an interaction framework by employing two other critical methodological tools in conjunction with context scenarios.
- The first is a set of general interaction design principles that provide guidance in determining appropriate system behavior in a variety of contexts.
- The second critical methodological tool is a set of interaction design patterns that encode general solutions (with variations dependent on context) to classes of previously analyzed problems.

(5) Refinement

- The Refinement phase focus on detail and implementation. Interaction designers focus on task coherence, using key path (walkthrough) and validation scenarios focused on storyboarding paths through the interface in high detail.
- Visual designers define a system of type styles and sizes, icons, and other visual elements that provide a compelling experience with clear affordances and visual hierarchy.

Unit IV End Sem.

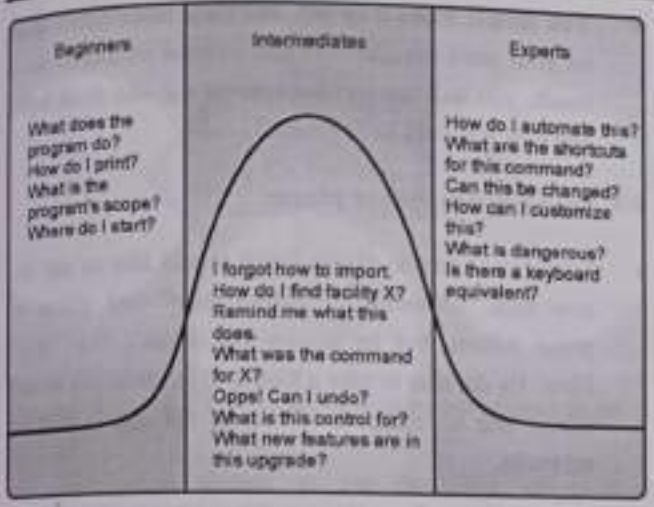
- Industrial designers, when appropriate, finalize materials and work closely with engineers on assembly schemes and other technical issues.
- The culmination of the Refinement phase is the detailed documentation of the design, a form and behavior specification, delivered in either paper or interactive media as context dictates.

(6) Development Support

- Even a very well-conceived and validated design solution can't possibly anticipate every development challenge and technical question.
- It's important to be available to answer developers' questions as they arise during the construction process.
- It is often the case that as the development team prioritizes their work and makes trade-offs to meet deadlines, the design must be adjusted, requiring scaled-down design solutions.
- If the interaction design team is not available to create these solutions, developers are forced to do this under time pressure, which has the potential to gravely compromise the integrity of the product's design.

4.4 USER FOCUS

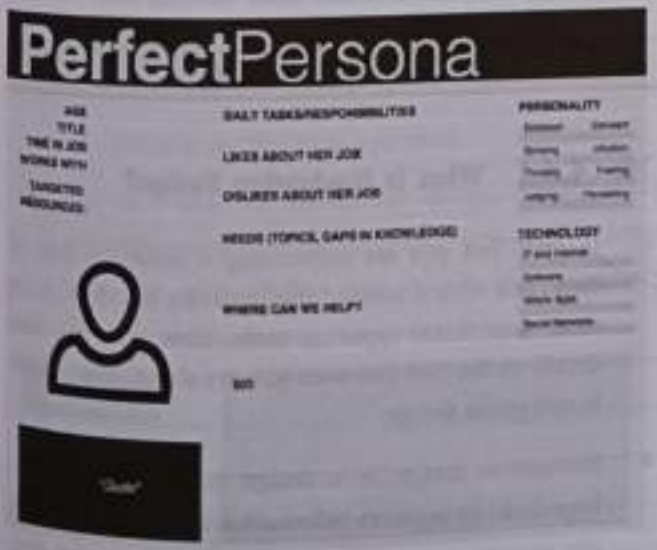
- User-Centered Design (UCD) is an iterative design process in which designers focus on the users and their needs in each phase of the design process.
- In UCD, design teams involve users throughout the design process via a variety of research and design techniques, to create highly usable and accessible products for them.
- Software design is the process that consists of a user-oriented approach where all meaningful activities are systematically framed together and utmost importance is given to the person who is going to use that system. We always know the activities in the user context-
  - Understanding user requirements.
  - Understanding purpose of the user.
  - Understanding the work environments.
- Users of a product should be the main focus of the design effort. They are the people who are personally utilizing the product to accomplish a goal.
- Information we are interested in learning from users includes :
  - The context of how the product (or analogous system, if no current product exists) fits into their people to perform activities; understanding goals allows you to understand the expectations and aspirations of your users, which can in turn help you, decide which activities are truly relevant to your design.
  - Task and activity analysis is useful at the detail level, but only after user goals have been analyzed
  - The programmer's task is to put barriers up to protect data integrity.
- The user's goal is to handle the various needs of the client.
  - **Result :** If the client hasn't decided where an order is to be shipped, the software rejects the order for lack of a valid shipping address.
  - The programmer has imposed *His* goals on the user, rather than making the software fit the user's goals.
  - The User's goals remain constant, even as the technology changes!
- When developers always thinking about the level of experience, three level of experience are Beginners, Intermediates and Experts help them to understand the level of users.
- Programmers create interactions suitable only for experts, while the marketers demand interactions suitable only for beginners, but as we have seen the largest, most stable, and most important group of users is the intermediate group.
- We prefer the term perpetual intermediates, because although beginners quickly improve to become intermediates, they seldom go on to become experts.



(1024)Fig. 4.4.1 : Experience Level of Users

Modeling Users : Persona

- Using research to create descriptive models of users is a uniquely powerful tool for interaction design. We call these user models personas.
- Personas provide us with a precise way of thinking and communicating about how users behave, how they think, what they wish to accomplish, and why.
- Personas are not real people, but they are based on the behaviors and motivations of real people we have observed and represent them throughout the design process.



(1025)Fig. 4.4.2 : Modeling User Persona

4.5 SCENARIOS

UQ. Write a scenario for Music player design?

(SPPU - Q. 6(b), May 19, 8 marks)

UQ. A scenario is an idealized but detailed description of a specific instance of human-computer interaction (HCI). Scenarios specify how users carry out their tasks in a specified context. Write scenarios for purchasing an airline ticket.

Note - Generate scenarios to cover a wide range of situations, not just the most common ones. Include problem situations that will test the system concept, not just straight forward scenarios.

(SPPU - Q. 6(b), May 18, 8 marks)

- User scenarios are stories which designers create to show how users might act to achieve a goal in a system or environment.
- Designers make scenarios to understand user's motivations, needs, barriers and more in the context of how they would use a design, and to help ideate, iterate and usability-test optimal solutions.
- Design teams systematically work to account for the most common use case and work to create a reliable narrative to use as a guide. To ideate toward accurate pictures of your users, their world and how your solution might solve their problems best, you begin by doing user research.
- Then, you create personas to represent your target users and flesh out their experiences to reflect realistic situations. So, to have the ingredients for a user scenario, you first must clearly define the following factors :
  - Background – who are your users (including their knowledge base and skillset/s)?
  - Motivations – what goals do they want to achieve?
  - Tasks – what must they do to reach those goals?
  - Context of use – how will they encounter your design?
    a. Environment – where will they try to use it?
    b. Challenges – when they try to use it, what can get in their way (e.g., signal loss)?

Unit IV End Sem.

- You can create user scenarios as highly visual narratives or storyboards with pictures of the personas you're modelling them on. Essential points to consider include these:
- Provide the context of :
  - Who – details of the persona.
  - What their goals are.
  - When they might perform tasks (including obstacles).
  - Where they might do these (including obstacles).
  - Why they want to do things, must perform subtasks, etc.
- Focus on the bigger picture but keep to the point – include the circumstances leading up to the interaction, the factors that impact the user's world and that might influence how they interact with a solution (e.g., cultural context) and anything they may need before encountering or using the solution (e.g., information).
- Make the scenario understandable for people who don't have technical backgrounds – so everyone, including stakeholders, can get on board with elements they can easily relate to and can stay open-minded about necessary processes, etc.
- Keep user scenarios tightly centered on the users themselves – to ensure any ideas about design features stay grounded in the reality of the user's context.

☞ Using Scenarios in Website Design

- It is impossible to write down every scenario that every user has for visiting your website. Instead, before you start to put the site together, write down 10 to 30 of the most common reasons that users have for visiting or tasks that users want to do.
- Scenarios can also work together with personas by serving as the stories behind why the particular persona would come to your website. What does the persona hope to accomplish by visiting the website? What characteristics of the persona might help or hinder his or her site interaction?
- You should focus on users and their tasks rather than on your site's organization and internal structure. As a result, you will know what content the site must have and how it should be organized.

☞ Scenarios in movie player

- Scenarios – movie player Brian would like to see the new film "Moments of Significance" and wants to invite Alison, but he knows she doesn't like "arty" films. He decides to take a look at it to see if she would like it and so connects to one of the movie sharing networks.
- He uses his work machine as it has a higher bandwidth connection, but feels a bit guilty.
- He knows he will be getting an illegal copy of the film, but decides it is OK as he is intending to go to the cinema to watch it. After it downloads to his machine he takes out his new personal movie player.
- He presses the 'menu' button and on the small LCD screen he scrolls using the arrow keys to 'Bluetooth connect' and presses the select button. On his computer the movie download program now has an icon showing that it has recognized a compatible device and he drags the icon of the film over the icon for the player. On the player the LCD screen says "downloading now", a percent done indicator and small whirling icon.

⏭ 4.6 NAVIGATION DESIGN

4.6.1 What Is Navigation Design?

- Imagine that you are researching a particular type of bird. You visit a website that provides information on more than 9,000 types of birds. How will you find details on the bird you want to learn about? The answer is navigation design.
- Navigation design is a design process that utilizes hyperlinks to organize information on a website so that site visitors can navigate, or find, the information they are looking for. Navigation can be text-based or image-based.

(1028)**Fig. 4.6.1 : Navigation tells us which way to go**

- Navigation design is the discipline of creating, analyzing and implementing ways for users to navigate through a website or app.
- Navigation plays an integral role in how users interact with and use your products. It is how your user can get from point A to point B and even point C in the least frustrating way possible.
- To make these delightful interactions, designers employ a combination of design patterns including links, labels and other UI elements. These patterns provide relevant information and make interacting with products easier.
- Good navigation design can :
  - Enhance a user's understanding.
  - Give them confidence using your product.
  - Provide credibility to a product.
- The best kind of navigation design is one which promotes usability. Poor navigation will result in fewer users for your product and this is why navigation design is central to user experience design.
- Navigation design is complex and there are many design patterns to choose from when optimizing the user experience. A design pattern is a general, reusable solution to a problem.
- No one pattern is necessarily better than the other. Each pattern that you use in your product will have to be carefully considered and tested before implementation.
- This ensures that the navigation pattern you have chosen is right for your product but more importantly that it is right for your users. Ideally, you want to approach navigation from a user-centered design perspective.

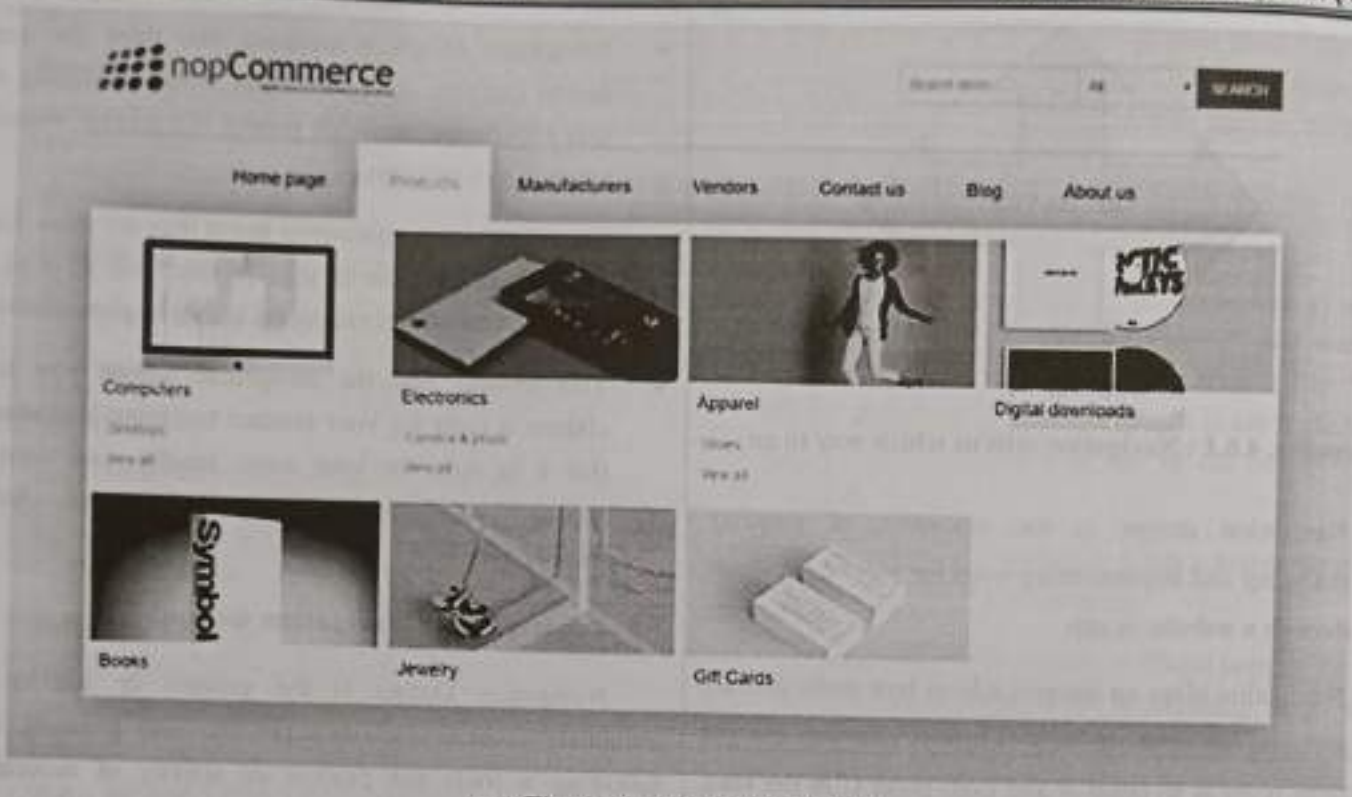
Constraints of navigation design

Navigation Design is the process or activity of accurately ascertaining one's position and planning and following a route the process or activity of accurately ascertaining one's position and planning and following a route.

- **Widgets :** The appropriate choice of widgets and wording in menusand buttons will help you know how to use them for a particular section or action.
- **Screen or Windows :** You need to find things on the screen.also understand the logical grouping of buttons.
- **Navigation within the Application :** You need to be able to understand what will happen when a button is pressed, to understand where you are in the interaction on any page.
- **Environment :** The word processor has to read documents from disk, perhaps some are on remote networks. You swap between applications, perhaps cut and paste.

Unit IV End Sem

NOTES



(1027)**Fig. 4.6.2 : Navigation Design**

4.6.2 Why Navigation Design is so Important ?

- Well-designed websites have navigation that is clear, concise, and intuitive, enabling the user to negotiate between web pages with little difficulty. When web pages have ineffective navigation, they tend to lose continuity and structure.
- Let's explore some of the basic aspects of good navigation :
  - **Link to home page :** When laying out the navigation, ensure the home page can be reached from any displayed page.
  - **Group like functions :** Gather similar functionality in one screen location where the user expects to see them. For example, think about the editing functions in a word processor. All the available editing functions are reachable from one location usually under the 'Edit' function.
  - **Limit hyperlinks :** Hyperlinks relating to the subject are always desirable, but don't overdo the number of hyperlinks. The more hyperlinks used, the more complex the page appears.
  - **Three click rule :** Web page destinations must be reached within three clicks. More than three clicks indicate a complex structure requiring reorganization.
  - **Consistency :** Maintain consistency among pages. Place navigation in the same location on each page. For example, if your home page feature vertical side bar navigation on the left side of the page, the vertical side bar navigation should appear on the left side of *every* page.

4.7 SCREEN DESIGN

4.7.1 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

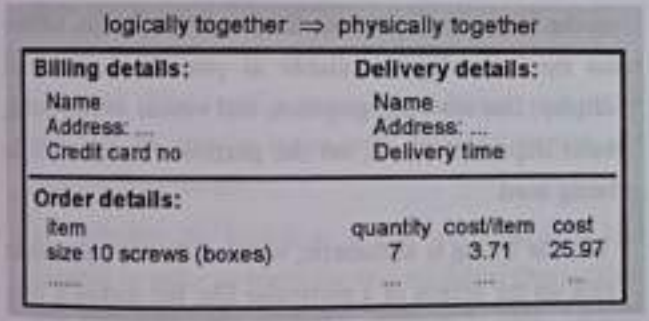
- A clear indication of when an action can make a permanent change in data

4.7.2 Screen Design Goals

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

4.7.3 Tools for Layout

- We have a number of visual tools available to help us suggest to the user appropriate ways to read and interact with a screen or device.



(1028)**Fig. 4.7.1 : Grouping related items in an order screen**

Grouping and structure

If things logically belong together, then we should normally physically group them together. This may involve multiple levels of structure. We can see a potential design for an ordering screen.

Notice how the details for billing and delivery are grouped together spatially; also note how they are separated from the list of items actually ordered by a line as well as spatially. This reflects the following logical structure :

Order

- Administrative information
  - Billing details
  - Delivery details
- Order information
  - Order line 1
  - Order line 2

Order of groups and items

In general we need to think: what is the natural order for the user?

This should normally match the order on screen. For data entry forms or dialog boxes we should also set up the order in which the tab key moves between fields. Occasionally we may also want to force a particular order; for example we may want to be sure that we do not forget the credit card details

- **Decoration**

  Decorative features like font style, and text or background colors can be used to emphasize groupings.

- **Alignment**

  Alignment of lists is also very important. For users who read text from left to right, lists of text items should normally be aligned to the left. Numbers, however, should normally be aligned to the right (for integers) or at the decimal point. This is because the shape of the column then gives an indication of magnitude – a sort of mini histogram. Items like names are particularly difficult.

- **White space**

  Spacing or whitespace, white space is any section of a document that is unused or space around an object. White spaces help separate paragraphs of text, graphics, and other portions of a document, and help a document look less crowded. Using white space effectively in a document keeps the reader reading the document and helps the reader quickly find what they are interested in reading.

- **How to create white space**

  White space is created by pressing the return key, spacebar key, or the tab key and can also be created by setting the document's margins and inserting form feeds or tables.

Unit IV End Sem.

4.7.4 User Action and Control

Entering information

- In each case the screen consists not only of information presented to the user, but also of places for the user to enter information or select options. Many of the same layout issues for data presentation also apply to fields for data entry. Alignment is still important. It is especially common to see the text entry boxes aligned in a jagged fashion because the field names are of different lengths.
- This is an occasion where right-justified text for the field labels may be best or, alternatively, in a graphical interface a smaller font can be used for field labels and the labels placed just above and to the left of the field they refer to. For both presenting and entering information a clear logical layout is important.
- The task analysis techniques can help in determining how to group screen items and also the order in which users are likely to want to read them or fill them in. Knowing also that users are likely to read from left to right and top to bottom (depending on their native language!) means that a screen can be designed so that users encounter items in an appropriate order for the task at hand.

Knowing what to do

- If everyone designs buttons to look the same and menus to look the same, then users will be able to recognize them when they see them. It is important that the labels and icons on menus are also clear.
- Standards can help for common actions such as save, delete or print. For more system-specific actions, one needs to follow broader principles. For example, a button says "bold": does this represent the current state of a system or the action that will be performed if the button is pressed?

Affordances

- These are especially difficult problems in multimedia applications where one may deliberately adopt a non-standard and avant-garde style.
- How are users supposed to know where to click? The psychological idea of affordance says that things may suggest by their shape and other attributes what you can do to them: a handle affords pulling or lifting; a button affords pushing.
- These affordances can be used when designing novel interaction elements. One can either mimic real-world objects directly, or try to emulate the critical aspects of those objects.
- What you must not do is depict a real-world object in a context where its normal affordances do not work!

4.7.5 Appropriate Appearance

Presenting information

- The way of presenting information on screen depends on the kind of information: text, numbers, maps, tables; on the technology available to present it: character display, line drawing, graphics, and virtual reality; and, most important of all, on the purpose for which it is being used.
- The file listing is alphabetic, which is fine if we want to look up the details of a particular file, but makes it very difficult to find recently updated files. Of course, if the list were ordered by date then it would be difficult to find a particular file.
- Different purposes require different representations. For more complex numerical data, we may be considering scatter graphs, histograms or 3D surfaces; for hierarchical structures, we may consider outlines or organization diagrams. But, no matter how complex the data, the principle of matching presentation to purpose remains.
- We have an advantage when presenting information in an interactive system in that it is easy to allow the user to choose among several representations, thus making it possible to achieve different goals.

name	size
chap10	12
chap5	16
chap1	17
chap14	22
chap20	27
chap8	32
...	...

(1029)Fig. 4.7.2 : File listing

Aesthetics and utility

- The beauty and utility may sometimes be at odds. For example, an industrial control panel will often be built up of the individual controls of several subsystems, some designed by different teams, some bought in. The resulting inconsistency in appearance may look a mess and suggest tidying up.
- Certainly some of this inconsistency may cause problems. The conflict between aesthetics and utility can also be seen in many well designed posters and multimedia systems.
- In particular, the backdrop behind text must have low contrast in order to leave the text readable; this is often not the case and graphic designers may include excessively complex and strong backgrounds because they look good. The results are impressive, perhaps even award winning, but completely unusable!
- In consumer devices these aesthetic considerations may often be the key differentiator between products, for example, the sleek curves of a car. This is not missed by designers of electronic goods: devices are designed to be good to touch and feel as well as look.

Making a mess of it: colour and 3D

- The increasing use of 3D effects in interfaces has posed a whole new set of problems for text and numerical information.
- Whilst excellent for presenting physical information and certain sorts of graphs, text presented in perspective can be very difficult to read and the all too common 3D pie chart is all but useless.

Localization / internationalization

- If you are working in a different country, you might see a document being word processed where the text of the document and the file names are in the local language, but all the menus and instructions are still in English. The process of making software suitable for different languages and cultures is called localization or internationalization. It is clear that words have to change and many interface construction toolkits make this easy by using resources.
- When the program uses names of menu items, error messages and other text, it does not use the text directly, but instead uses a resource identifier, usually simply a number. A simple database is constructed separately that binds these identifiers to particular words and phrases.
- A different resource database is constructed for each language, and so the program can be customized to use in a particular country by simply choosing the appropriate resource database.

4.8 PROTOTYPING TECHNIQUES

UQ. What is a Prototype? Explain different types of rapid prototyping techniques.
(SPPU - Q. 5(b), May 19, Q. 6(a), May 18, Q. 6(a), Dec. 18, 8 Marks)

UQ. Explain Hill Climbing Approach with Prototyping?
(SPPU - Q. 8(a), May 19, 8 Marks)

Unit IV End Sem.

- Prototypes are experimental and incomplete designs which are cheaply and fast developed. Prototyping, which is the process of developing prototypes, is an integral part of iterative user-centered design because it enables designers to try out their ideas with users and to gather feedback.
- Prototyping is a development approach used to improve planning and execution of software projects by developing executable software systems (prototypes) for experimental purposes. It is very suitable for gaining experience in new application areas and for supporting incremental or evolutionary software development.

- Prototyping has many objectives including evaluation of the design, functionality and user interface. This report will be focusing on the user interface aspect with some linkage to the functionality. A function defined in the specification may seem useful and well defined but when the users finally try to use the application they find that their view was incorrect.
- Prototypes thus let user validate and evaluate their requirements and thus users can discover requirements omissions early in the process. Rapid Application Development Methodology uses development of the prototypes by the software team, working closely with the users in the requirements identification phase. Then these prototypes are either discarded (throw away prototypes) or enhanced (evolutionary prototypes) into production systems.
- Prototyping is an example of what is known as a hill-climbing approach. Imagine you are standing somewhere in the open countryside. You walk uphill and keep going uphill as steeply as possible. Eventually you will find yourself at a hill top.
- This is exactly how iterative prototyping works: you start somewhere, evaluate it to see how to make it better, change it to make it better and then keep on doing this until it can't get any better.

Fig. 4.8.1 : Moving little by little ..... but to where?

However, hill climbing doesn't always work. Imagine you start somewhere near Cambridge, UK. If you keep moving uphill (and it is very difficult to work out which direction that is because it is very flat!), then eventually you would end up at the top of the Gog Magog hills, the nearest thing around . . . all of 300 feet. However, if you started somewhere else you might end up at the top of the Matterhorn.

Hill climbing methods always have the potential to leave you somewhere that is the best in the immediate area, but very poor compared with more distant places. Figure shows this schematically:

Fig. 4.8.2 : Hill Climbing Approach with Prototyping

- If you start at A you get trapped at the local maximum at B, but if you start at C you move up through D to the global maximum at E.
- This problem of getting trapped at local maxima is also possible with interfaces. If you start with a bad design concept you may end at something that is simply a tidied up version of that bad idea.
- From this we can see that there are two things you need in order for prototyping methods to work:

  1. To understand what is wrong and how to improve.
  2. A good start point.

- The first is obvious; you cannot iterate the design unless you know what must be done to improve it. The second, however, is needed to avoid local maxima. If you wanted to climb as high as you could, you would probably book a plane to the Himalayas, not Cambridgeshire.
- A really good designer might guess a good initial design based on experience and judgment. However, the complexity of interaction design problems means that this insight is hard. Another approach, very common in graphical design, is to have several initial design ideas and drop them one by one as they are developed further.

- This is a bit like parachuting 10 people at random points of the earth. One of them is perhaps likely to end up near a high mountain.

4.8.1 Types of Prototypes

There are three types of prototypes –

1. Throwaway Prototypes
2. Evolutionary Prototypes
3. Incremental Development

1. Throwaway Prototypes

- Throwaway prototypes essentially use prototypes to clarify functional / user interface requirements and also help the managers identify any potential risks. Once the prototype is evaluated and the requirements updated the prototype is discarded and the development starts afresh.
- The main problem faced in this approach is when the management is tempted to take the prototype and enhance it into the real application. I actually have seen this in my professional life and this can lead to problems like – uncontrolled changes during the prototyping stage, design compromises while adding new features and missing features like security, performance & reliability at base levels.
- Since the prototype is not the final product the cost of iterations should be kept well in control because of the tendency of the development team to go into too many details of the application and then tempting to reuse some of that application in the final product.

2. Evolutionary Prototypes

- Evolutionary prototyping is based on developing an initial application and then iterating through multiple user evaluations and improvements. The important aspect of this is that the application has to be written in an environment that allows such development.
- A good language for such application development would be Visual Basic or PowerBuilder. The important aspects are that the design should be updated thoroughly at every iteration to avoid any possibility of compromises being introduced in the application.

3. Incremental Development

- Incremental development is the methodology where each delivered production software is taken as a prototype and after the evaluation by the users the changes and suggested and incorporated in the next release.
- This is a useful in the environment where the development cycles are small and the user group is anxious to get some functionality for being able to work. The main drawback in this methodology is that the system architecture needs to be determined initially and any changes to the architecture could lead to compromises in the application robustness/reliability.
- The main purpose of prototyping is to involve the users in testing design ideas and get their feedback in the early stage of development, thus to reduce the time and cost. It provides an efficient and effective way to refine and optimize interfaces through discussion, exploration, testing and iterative revision.
- Early evaluation can be based on faster and cheaper prototypes before the start of a full-scale implementation. The prototypes can be changed many times until a better understanding of the user interface design has been achieved with the joint efforts of both the designers and the users.
- Prototyping can be divided into low-fidelity prototyping, medium-fidelity prototyping and high-fidelity prototyping. Where low-fidelity prototyping is mainly about paper-based mock-up, and high-fidelity is mainly about computer-based simulation.
- The determining factor in prototype fidelity is the degree to which the prototype accurately represents the appearance and interaction of the product, not the degree to which the code and other attributes invisible to the user are accurate.

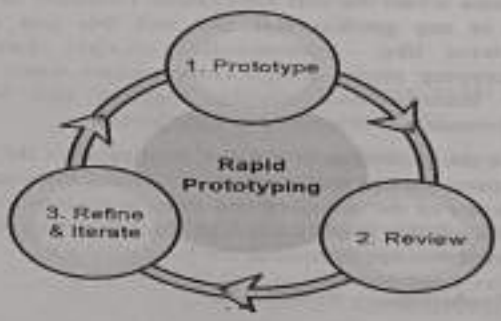
Unit IV End Sem.

4.8.2 Goals for Rapid Prototyping

- Conducting iterative testing and revision early in the product development cycle. This approach allows user testing to be an integral part of product design. Using measurable objectives (benchmarks) for user performance and attitude.

- Empirical measures prevent the endless debate that can sometimes follow user testing. If subject behavior falls below the benchmark there is little doubt among team members that something must be done.
- Using simple, proven testing techniques.
- A methodology that can be used for development of prototypes is :
  - **Identify Objectives :** First, a definition of the problem to be solved must be expressed, along with the criteria by which its success is to be judged.
  - **Identify Risk :** No realistic software effort can expect a clear and complete statement as the result of a step labeled 'define the problem', as the above item might suggest. In the gray and uncertain areas of the problem definition lurk great risks. These gray areas of risk must be identified and made explicit.
  - **Formulate Prototyping Hypotheses :** Once the risk areas have been expressed, the developer is in a position to design experiments to specifically address those risks. Each experiment addresses a hypothesis concerning the nature and potential remedy of the risk to a product's success. The hypothesis should include explicit linkage to the parts of the project that may be affected.
  - **Construct Prototype System :** A system that implements the hypothesized solution must be developed. Traditional programming practices are too expensive to make this practical in many situations. Thus there is a need to use specialized tools that facilitate inexpensive prototyping.
  - **Instrument Prototype :** Since the primary purpose of constructing a prototype is to answer one or more questions concerning its functional or other characteristics, it will usually be necessary to gather dynamic information regarding the operation of the software. In a large prototype, the complexity and volume of information is likely to exceed the limit of a single person's ability to comprehend it. It would be desirable to use automated tools for this purpose.
  - **Experiment :** The prototype system must be exercised to determine its behavior and gather the output from the system instrumentation.
  - **Evaluate Results :** The results of the experiments must be evaluated to assess the efficacy of the hypothesized solution.
  - **Repeat :** This entire process is repeated until one of three outcomes is reached sufficient information has been derived from the prototyping activities to begin the development of a product no solution to the problem can be found or, equivalently, that the solution would be too complex or costly to be of benefit the prototype system itself is of sufficient quality to serve as the production-quality system or, that it can be modified to serve as the production-quality system at less cost than a full-scale production-quality development effort.

4.8.3 Effective Techniques for Rapid Prototyping



(1030) **Fig. 4.8.3 : Rapid Prototyping**

- **Step 1 – Prototype :** In prototyping it is essential to understand the user requirements and expectations from the client in 2 to 3 meetings. After gathering all the requirements, convert them into mock ups. Mock ups could be in the form of sketches or low-fidelity wireframes depending on the project demand. While mocking up, it is important to take into account the user experience standards and the best practices.

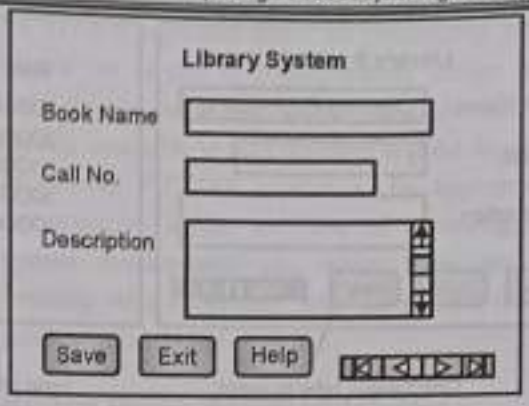
- **Step 2 – Review :** In this step, share the wireframes or prototypes with the client and other stakeholders to check whether these are up to their expectations and that everything that they require is incorporated in the prototype.
- **Step 3 – Refine & Iterate :** Depending on the feedback received from the client in step 2, pitch on the key areas that need refinement and clarity. Finally iterate the whole prototyping process until you get what the client expects. This whole process could take multiple iterations until the prototype is finalized.

4.8.4 Rapid Prototyping Techniques

Low-fidelity prototyping

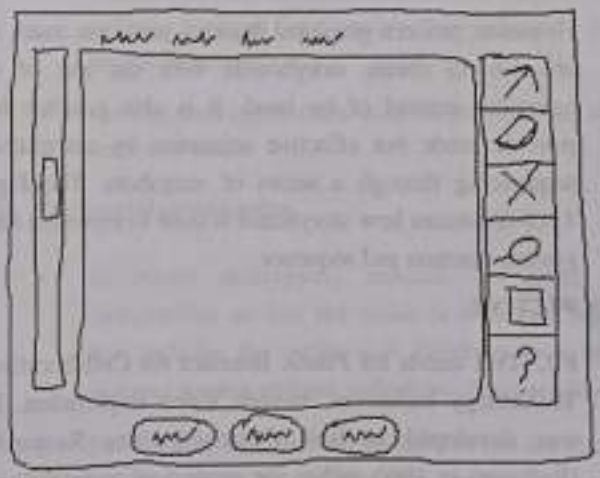
(1) Sketches

- Sketching techniques, a kind of visual brainstorming, can be useful for exploring all kinds of design ideas. After producing initial sketches the best ideas can be further developed by constructing cardboard representations of the design, which can be evaluated with users. This can then be followed by developing scenarios, software or video prototypes.
- Freehand sketches are essential for crystallizing ideas in the early stages of design. Through the act of putting ideas down on paper and inspecting them, designers see new relations and features that suggest ways to refine and revise their ideas.
- Sketches make apparent to designers not only perceptual features but also inherently non-visual functional relations, allowing them to extract functions from perception in sketches.
- The type of mock-up depends on how advanced the idea is. It may be quicker and cheaper to use paper-and-pencil forms at early stages, whereas computer-based prototypes may be important in later stages for exploring and demonstrating interaction and design consistency.
- As one can imagine, the sketch technique is as simple as drawing the outward appearance of intended system on paper. However, creativeness is needed. The Fig. 4.8.4 is a sketch of a screen design.



(1031)**Fig. 4.8.4 : A Sketched Screen Design**

- Besides paper-and-pencil work, sketches can also be made with the aid of computer software, such as the Paint package in Windows and SILK. SILK allows designers to quickly sketch an interface electronically and interactively. The Fig. 4.8.5 is a sketch created by SILK.

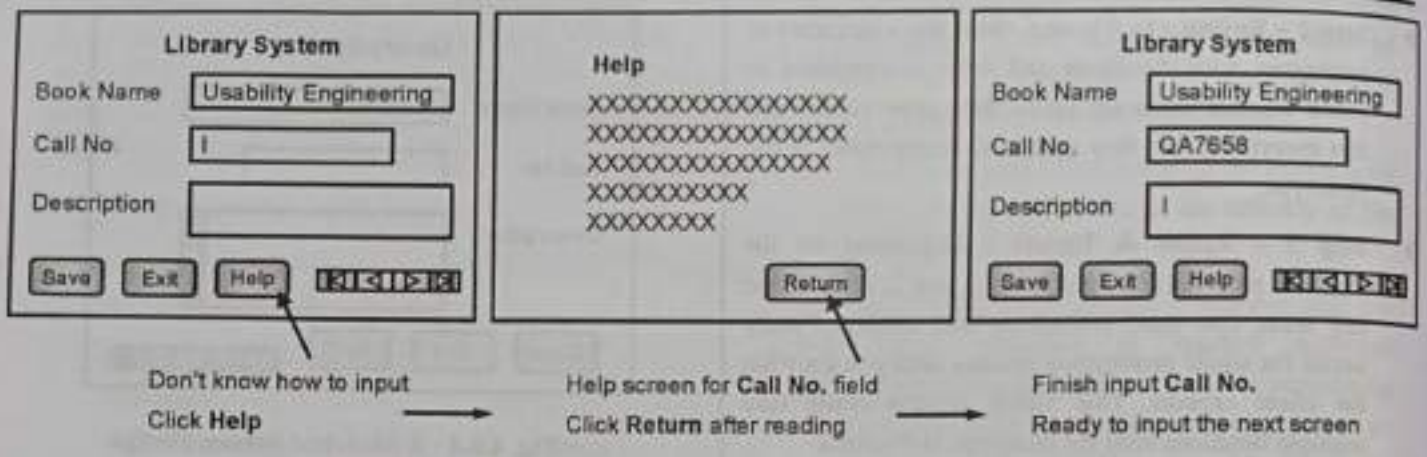


(1032)**Fig. 4.8.5 : Sketched Application Interface Created with SILK**

(2) Storyboard

- Storyboard origins from the film industry, where a series of panels roughly depicts snapshots from an intended film sequence in order to get the idea about the eventual scene.
- Storyboard is a graphical depiction of the outward appearance of the intended system without accompanying system functionality. Storyboard provides snapshots of the interface at particular points in the interaction so that the users can determine quickly if the design is heading in the right direction.

Unit IV End Sem.



(1033)**Fig. 4.8.6 : An Example of Storyboard**

- Storyboards do not require much in terms of computing power to construct, in fact, they can be mocked up without the aid of computers. The materials needed are office stationery, such as pens or pencils of different colors, stickers, and so on.
- However, modern graphical drawing packages make it possible to create storyboards with the aid of a computer instead of by hand. It is also possible to provide crude but effective animation by automated sequencing through a series of snapshots. The Fig. 4.8.6 illustrates how storyboard is used to represent the system function and sequence .

(3) PICTIVE

- PICTIVE stands for Plastic Interface for Collaborative Technology Initiatives through Video Exploration. It was developed at Bell Communications Research (Bellcore) in 1990 within the context of participatory design. The initial experiments of PICTIVE were conducted by Muller and his group in their projects. It is an experimental participatory design technique that is intended to enhance user participation in the design process.
- The PICTIVE technique provides a fine-grained, dynamic paper and pencil concretization mock-up of what the system will eventually look like and how it will behave. The components are literally made of colored plastic. Their relative durability and inexpensiveness encourage an atmosphere of exploration and invention.
- The PICTIVE mock-up is intended to be extensively modified in real-time by the users. PICTIVE is less as a means for the evaluation of an already-designed interface, but rather for the creation of the design of an interface.
- PICTIVE was begun in reaction to software rapid prototyping environments, in which developers have a disproportionate design impact, but non-computer users are relatively disempowered by the complexity of current software prototyping environments.
- The PICTIVE techniques were designed to be used by people who were not necessarily programming professionals, so all participants have equal opportunity to contribute their ideas.
- The apparatus for PICTIVE includes video camera and a collection of design objects. The design objects fall into two categories. The first category is simple office materials, including pens, high-lighters, papers, stickers labels and paper clips -- all in a range of bright colors. The second category is materials prepared by the developer, such as menu bars, dialogue boxes, special icons for the project and so on.

☞ **The procedures of PICTIVE are as follows**

- First, both the users and the developers are asked to prepare a "homework". Typically, the users are asked to think about task scenarios, for instance, "what you would like the system to do for you and what steps are required to finish the job".

- The developers need to construct a preliminary set of system components for the users to manipulate based on prior discussions with the users. Fig. 4.8.7 shows some of such components made of plastic .

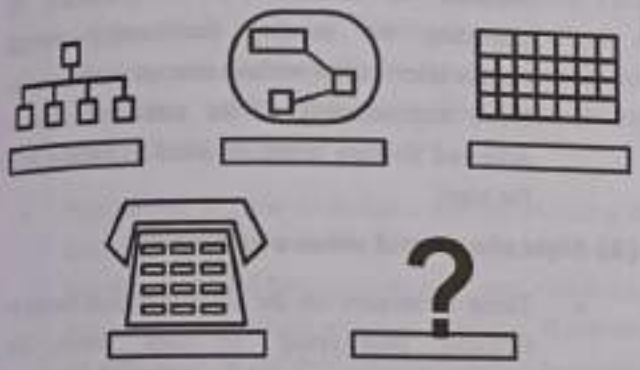


Fig. 4.8.7 : PICTIVE Plastic "Icons"

- Second, during the PICTIVE session, both the users and developers manipulate the design objects on a design surface, where the designs are put together as multiple layers of sticky notes and overlays that can be changed by simple colored pens. Each participant brings her or his expertise.
- The resulting design is not driven by any single participant, but represents a synthesis of the different participants' different views. The users' work scenarios are explored in detail, leading to a discussion of how the developers' technology can be applied to meet the users' human needs. A coordinator may be needed to keep the group on track.
- A video camera is focused on the design objects and to record the voices of the design team as they manipulate those objects. The video record of the design session will serve as a dynamic presentation of the design. The Fig. 4.8.8 illustrates a scene in the PICTIVE session .

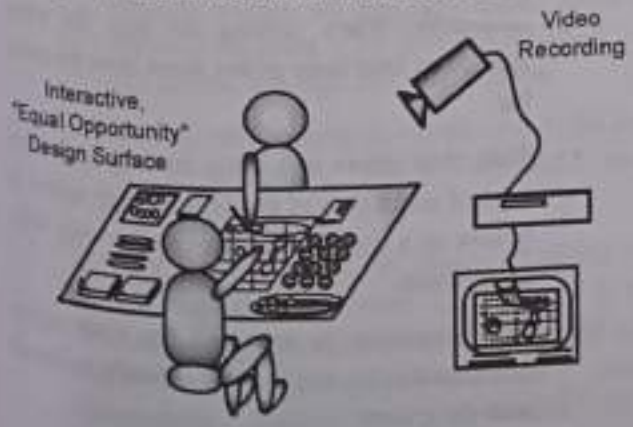


Fig. 4.8.8 : PICTIVE Setting

- PICTIVE is especially suited for prototyping activities carried out as part of a participatory design process since the low-tech nature of the materials make them equally accessible to both the users and the developers. In general, PICTIVE appears to be appropriate in circumstances similar to those of knowledge-based system development, i.e., when there are users available who understand what they need from the product or the project.

Medium (High) -fidelity prototyping

(1) Computer-based simulation

Medium-fidelity prototypes simulate or animate some but not all features of the intended system. There are three approaches to limit prototype functionality.

(a) Vertical prototyping

- Vertical prototyping cuts down on the number of features, so that the result is a narrow system that includes in-depth functionality, buy only for a few selected features.
- Vertical prototypes allow users to perform and test some real tasks.

(b) Horizontal prototyping

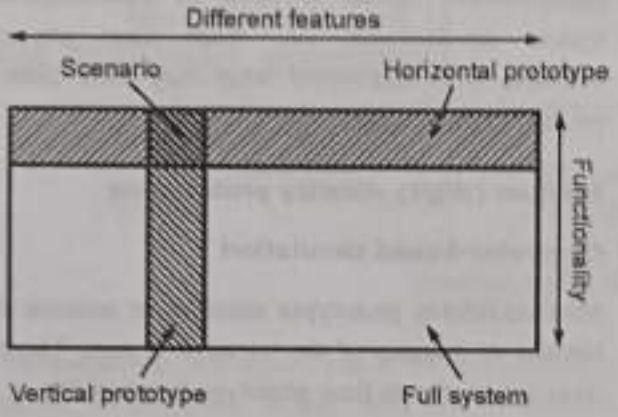
- Horizontal prototyping reduces the level of functionality so that the result is a surface layer that includes the entire user interface to a full-featured system without underlying functionality.
- Horizontal prototypes allow users to feel the entire interface, even though they can not perform any real tasks.
- The main advantages of horizontal prototypes are that they can be implemented fast with the use of prototyping and screen design tools, and they can be used to assess the interface as a whole.

(c) Scenario

- Scenario reduces both the number of features and the level of functionality. It can simulate the user interface as long as the user follows a previously planned path, i.e., a user can use a specific set of computer facilities to achieve a specific outcome under specified circumstances.

Unit IV End Sem.

- Scenarios can be easy and cheap to build, and to be used during early evaluation of a user interface design to get user feedback without the expense of constructing a running prototype. It can also be used for user testing if they are developed with slightly more detail than a pure narrative.



(1036)**Fig. 4.8.9 : Two Dimensions of Prototyping**

- The concepts of vertical, horizontal and scenario prototyping are illustrated in Fig. 4.8.9.
- Computer-based prototypes become easier to implement than before because there are more and more software prototyping tools, such as RAPID , HyperCard , CHIRP . Besides, there are many easy-to-learn and easy-to-use 4th generation languages, such as Smalltalk and Microsoft Visual Basic.

(2) Wizard of Oz

- This is a method of testing a system that does not exist. It allows designers to test ideas without implementation a system.
- The Wizard of Oz technique works as follows: the user interacts with a screen, but instead of a piece of software responding to the user's requests, a developer (the wizard) is sitting at another screen (generally in another room) simulating the system's intelligence and interacting with the user . The wizard may simulate all or part of the system function.
- When setting up a Wizard of Oz simulation, experience with previously implemented systems is helpful in order to place realistic bounds on the wizard's "abilities".
- Wizard of Oz is ideal for the testing of preliminary prototypes and to gather users' expectations on the system. With this technique, the developers can develop a limited functionality prototype and enhance its functionality in evaluation by providing the missing functionality through human intervention without cost on programming. Extra understanding of the users can also be achieved through being involved so closely with the users.

(3) Slide shows and video prototyping

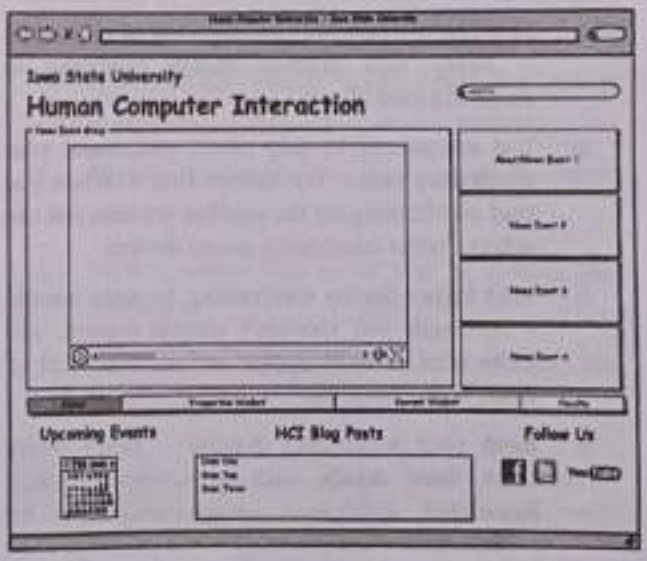
- These techniques use the communication media to facilitate prototyping. In slide shows, the storyboard prototype is encoded on a computer with software tools.
- The scene transition is activated by simple user input. The slides form a simple horizontal or vertical prototype.
- Video prototyping eliminates software limitations. It requires no post-production editing or any special expertise in video production. Here is an example of how to make video prototype for a pull down menu. First, we draw a menu bar on paper and a mouse arrow on clear acetate.
- Second, turn on the camera and move the acetate so that it looks as if the mouse is moving over the menu bar.
- When the mouse is over the menu title, we make a clicking sound and pause the camera. In the third step, we draw the menu on a small piece of paper, put it under the menu title, and restart the camcorder. When viewing the tape, the menu appears to have been pulled down from the menu bar.
- Both slide shows and video prototyping provide kind of simulation of the system. They appear to behave as a real system although they only show some scenes.
- The simulation is restricted by some tightly predefined tasks, and the user is hardly to interact with the system.

4.9 WIREFRAMING

- Wireframing is a process where designers draw overviews of interactive products to establish the structure and flow of possible design solutions. These outlines reflect user and business needs. Paper or software-rendered wireframes help teams and stakeholders ideate toward optimal, user-focused prototypes and products.
- Wireframing is a way to design a website service at the structural level. A wireframe is commonly used to layout content and functionality on a page which takes into account user needs and user journeys. Wireframes are used early in the development process to establish the basic structure of a page before visual design and content is added.

4.9.1 Types of Wireframes (Low to Medium Fidelity)

- Wireframes can vary both in their production, from paper sketches to computer-drawn images and in the amount of detail that they convey. Low and high-fidelity are terms used to identify the level of wireframe production or functionality. (Low to Medium Fidelity)
- A wireframe is a low fidelity representation of the product. It represents the layout of a webpage that demonstrates what interface elements will exist on key pages. It generally covers the 3 basic questions in the design process – What, Where & How.
  - What are the main key elements of the layout? e.g. carousel, widget, left nav, hero space. Where are the elements going to be placed in the layout. e.g. top, bottom, left, right or middle section.
  - How are they going to be placed.? e.g. in the form of a navigation, blocks of text, visual representation.
- Sketching (Low Fidelity): The fastest and easiest way to start creating a wireframe is to sketch it out. Sketching is faster and is often done in the very early stages of a design process, particularly during brainstorming sessions to understand client requirements and get early feedback from them.
- Creating Wireframes Using Online Tools (Medium Fidelity): There are many tools available on the web to create medium fidelity wireframes.. As we start using these tools, the fidelity of a design increases in most areas. Medium fidelity designs give a more formal and polished look.
- A medium fidelity design helps users to understand the behavior of an application giving less importance to the aesthetics and look & feel. Interactivity is also incorporated in such designs, but limited to some extent. One common interactive feature at this stage is linking pages/screens with one another to understand the navigation of an application.



(2017)Fig. 4.9.1 : Wireframing

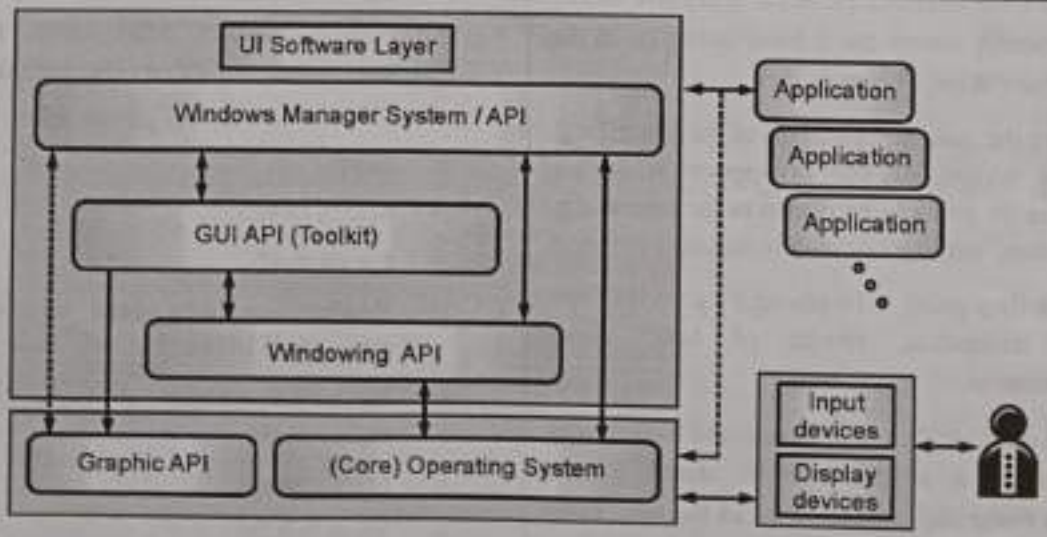
- Although wireframes differ from site to site, the following elements often are included as standard elements on wireframes :
  - Logo
  - Search field
  - Breadcrumb
  - Headers, including page title as the H1 and subheads H2-Hx
  - Navigation systems, including global navigation and local navigation
  - Body content
  - Share buttons
  - Contact information
  - Footer

Unit IV End Sem

- As a Designer, you should show what elements your users would expect to find and how these work in flow. To begin, you should:
  - Focus on functionality, accessibility, layout and navigation to make a design easier to use, produce and sell – Leave nice-to-have features out.
  - Structure a hierarchy with a list of prioritized elements for each page – Determine the information architecture early so you can categorize information clearly.
  - Divide the screen into large blocks for content. Fine-tune these blocks with details – links, placeholders for images, etc.
  - Maintain a clean grid-oriented view of all content – Apply best practice design principles to maximize ease of use.
  - Use annotations to help others understand your wireframes faster. Put mobile first – When you start wireframing for the smallest screens, you can achieve better consistency across devices.
  - With higher-fidelity wireframing, be more specific – Although you shouldn't overdo content, still show what needs to appear and accurate sizes of fonts, icons, links, etc.
  - Keep your wireframes concise – Don't worry about finer details such as aesthetic appeal. Remember, wireframes are primarily tools for collaboration toward making better prototypes and products faster.

4.10 UNDERSTANDING THE UI LAYER AND ITS EXECUTION FRAMEWORK

- Interactive applications are implemented and executed using the user interface (UI) software layers (collectively the UI layer). The UI layer refers to a set of software that operates above the core operating system (and underneath the application). It encapsulates and exposes system functions for input and output (I/O).
- Facilitation of development of I/O functionalities (in the form of an application programming interface/library [API] or toolkit). Run-time management of graphical applications and UI elements often manifested as windows or graphical user interface (GUI) elements (in the form of separate application often called the window manager).
- Since most interfaces are graphical, the UI layer uses a two - or three-dimensional (2-D or 3-D) graphical system based on which GUI elements are implemented.
- Thus, to summarize, the UI layer is largely composed of
  (a) An API for creating and managing the user interface elements (e.g., windows, buttons, menus) and
  (b) A window manager to allow users to operate and man-age the applications through its own user interfaces.



(1098)**Fig. 4.10.1 : User Interface software layer for windows based multitasking UI**

- The Fig. 4.10.1 illustrates the UI layer as part of the system software in many computing platforms. The user interacts with the window/GUI-based applications using various input and output devices.
- At the same time, aside from the general applications, the user inter-acts with the computer and manages multiple application windows/tasks (e.g., resizing, focusing, cutting and pasting, etc.) using the window manager.

Window Manager

- **The window manager is regarded as both an application and API. User** applications are developed using the APIs that represent abstracted I/O-related functionalities of the UI layer, such as those for window managing (resizing, dragging, copy and paste, etc.), GUI elements and widgets (windows, menus, buttons, etc.) and basic windowing (creating/destroying window, deactivating window, etc.).
- These APIs are abstracted from the even lower-level APIs for 2-D/3-D graphics and the operating system. Note that the architecture described here can be equally applied to non-window-based systems, such as those for layer-based systems(e.g. Mobile phones). Through such an architecture and **abstraction, it becomes much easier to develop and implement interactive applications and their user interfaces.**

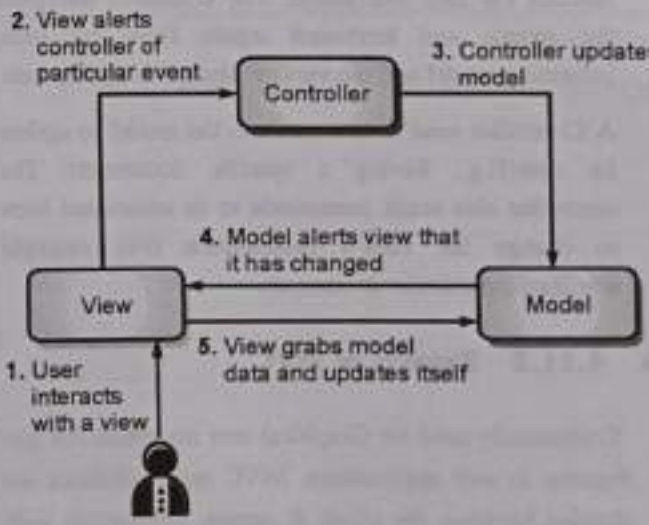
4.11 MODEL-VIEW-CONTROLLER (MVC) FRAMEWORK

UQ. What is the need of MVC pattern? Draw figure and explain. **(SPPU - Q. 8(b), May 19, 8 Marks)**

UQ. Write short note on : (i) Wire-framing
(ii) Mobile-View-controller (MVC) Framework
(SPPU - Q. 8(b), Dec 19, 8 Marks)

4.11.1 MVC Architecture

- The architecture components of the MVC pattern are designed to handle different aspects of an application in development. The MVC design pattern serves to separate the presentation layer from the business logic.
- MVC patterns separate the input, processing, and output of an application. This model divided into three interconnected parts called the model, the view, and the controller. All of the three above given components are built to handle some specific development aspects of any web or .net application development.
- In the MVC application development, the controller receives all requests for the application and then instructs the model to prepare any information required by the view. The view uses that data prepared by the controller to bring the final output.
- Why do developers care about MVC? MVC is popular in app and web development, and it's one of the most widely used software design patterns for app and web development. The Model View Controller design pattern separates concerns into one of 3 buckets :
  1. **Model :** It includes all the data and its related logic
  2. **View :** Present data to the user or handles user interaction
  3. **Controller :** An interface between Model and View components



(1D39)**Fig. 4.11.1 : MVC Architecture**

Model

- The model component stores data and its related logic. It represents data that is being transferred between controller components or any other related business logic.

Unit IV End Sem.

- For example, a Controller object will retrieve the customer info from the database. It manipulates data and sends back to the database or uses it to render the same data.
- It responds to the request from the views and also responds to instructions from the controller to update itself. It is also the lowest level of the pattern which is responsible for maintaining data.

☞ View

- A View is that part of the application that represents the presentation of data.
- Views are created by the data collected from the model data. A view requests the model to give information so that it resents the output presentation to the user.
- The view also represents the data from charts, diagrams, and tables. For example, any customer view will include all the UI components like text boxes, drop downs, etc.

☞ Controller

- The Controller is that part of the application that handles the user interaction. The controller interprets the mouse and keyboard inputs from the user, informing model and the view to change as appropriate.
- A Controller send's commands to the model to update its state(E.g., Saving a specific document). The controller also sends commands to its associated view to change the view's presentation (For example scrolling a particular document).

4.11.2 Benefits of MVC

- Traditionally used for Graphical user interfaces (GUIs). Popular in web applications. MVC responsibilities are divided between the client & server, compatible with web application architecture
- MVC is helpful design pattern when planning development. Separation of Concerns: that code is divided based on function to either the model, view, or controller bucket. Works well with Ruby on Rails.

☞ Loosely Coupled

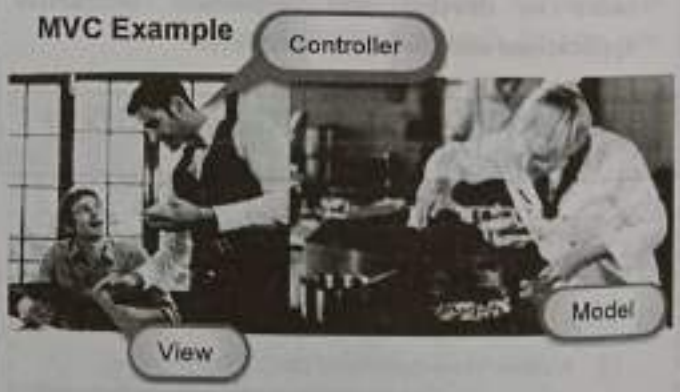
- Removes unnecessary dependencies
- Reuseable without modification
- MVC makes model classes reusable without modification
- Code reuse
- Extendable code
- High Cohesion
- Easier to maintain or modify
- Supports Multiple views
- Each part can be tested independently (Model, view, controller)

☞ MVC Examples

Let's see Model View Controller example from daily life:

Example 1

- Let's assume you go to a restaurant. You will not go to the kitchen and prepare food which you can surely do at your home. Instead, you go there and wait for the waiter to come on.



(1040) **Fig. 4.11.2 : MVC Example 1**

- Now the waiter comes to you, and you order the food. The waiter doesn't know who you are and what you want he just written down the detail of your food order.

- Then, the waiter moves to the kitchen. In the kitchen, waiter does not prepare your food. The cook prepares your food. The waiter is given your order to him along with your table number.
- Cook then prepared food for you. He uses ingredients to cooks the food. Let's assume that your order a vegetable sandwich. Then he needs bread, tomato, potato, capsicum, onion, bit, cheese, etc. which he sources from the refrigerator
- Cook final hand over the food to the waiter. Now it is the job of the waiter to moves this food outside the kitchen. Now waiter knows which food you have ordered and how they are served.
- In this MVC architecture example,

  View= You

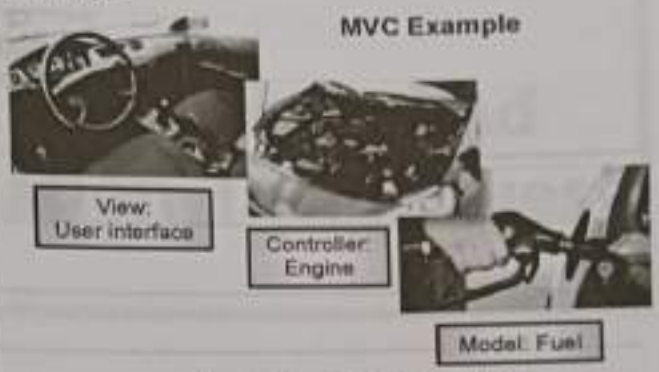
  Waiter= Controller

  Cook= Model

  Refrigerator= Data

  Let see one more MVC model example,

Example 2



(1041)**Fig. 4.11.3 : MVC Example 2**

- Car driving mechanism is another example of the MVC model.
- Every car consist of three main parts.
- View= User interface : (Gear lever, panels, steering wheel, brake, etc.)
- Controller- Mechanism (Engine)
- Model- Storage (Petrol or Diesel tank)
- Car runs from engine take fuel from storage, but it runs only using mentioned user interface devices.

Chapter Ends...

❑❑❑

UNIT V

CHAPTER 5 HCI Guidelines and Evaluation Techniques

Syllabus

Using toolkits, User interface management system (UIMS), Goals of evaluation, Categorization of Evaluation techniques, Choosing an Evaluation Method. DECIDE, Heuristic Evaluation, cognitive walk through, Usability testing

5.1 USING TOOLKITS

- Toolkits as a way to encapsulate interface design concepts for programmers, including widget sets, interface builders, and development environments. Such toolkits are used by designers and developers to create **interactive** applications. The software tools have a variety of names and provide a variety of functionality.
- The simplest are "toolkits". Libraries of program subroutines that create and run the many on-screen objects of a graphical user interface, such as windows, buttons, and menus. These can save some programming time and maintain consistency with existing applications, but they still leave most of the work to the programmer. A giant step up from toolkits are user interface management systems (UIMS). A UIMS gives you a toolkit and a programming environment for using it.

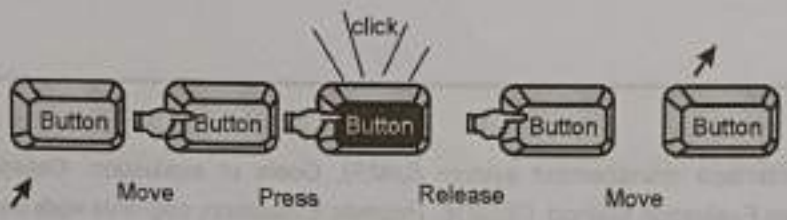


Fig. 5.1.1 : Toolkit

- The Fig. 5.1.1 show an example of how input and output are combined for interaction with a button object. As the user moves the mouse cursor over the button, it changes to a finger to suggest that the user can push it.
- Pressing the mouse button down causes the button to be highlighted and might even make an audible click like the keys on some keyboards, providing immediate feedback that the button has been pushed.
- Releasing the mouse button unhighlights the button and moving the mouse off the button changes the cursor to its initial shape, indicating that the user is no longer over the active area of the button.
- From the programmer's perspective, even at the level of a windowing system, input and output are still quite separate for everything except the mouse, and it takes quite a bit of effort in the application program to create the illusion of the interaction object such as the button we have just described.
- To aid the programmer in fusing input and output behaviors, another level of abstraction is placed on top of the window system – the toolkit.
- A toolkit provides the programmer with a set of ready-made interaction objects – alternatively called interaction techniques, gadgets or widgets – which user can use to create an application programs.

Toolkits provide this level of abstraction

- programming with interaction objects (or – techniques, widgets, gadgets)
- promote consistency and generalizability
- through similar look and feel
- amenable to object-oriented programming

The interaction objects have a predefined behavior, such as that described for the button, that comes for free without any further programming effort. Toolkits exist for all windowing environments (for example, OSF/Motif and XView for the X Window system, the Macintosh Toolbox and the Software Development Toolkit for Microsoft Windows).

To provide flexibility, the interaction objects can be tailored to the specific situation in which they are invoked by the programmer.

For example, the label on the button could be a parameter which the programmer can set when a particular button is created. More complex interaction objects can be built up from smaller, simpler ones.

Ultimately, the entire application can be viewed as a collection of interaction objects whose combined behavior describes the semantics of the whole application.

5.2 USER INTERFACE MANAGEMENT SYSTEM (UIMS)

- UIMS is a tool, or a set of tools, which helps to specify, implement, test and maintain a user interface.
- The role of a UIMS is to mediate the interaction between a user and an application; satisfying user requests for application actions, and application requests for data from the user. It accepts as input a dialogue specification, describing the detailed structure of the interaction.
- The following advantages of constructing user interfaces with a UIMS:
  - A UIMS improves the efficiency with which user interface designers can use their skills;
  - A UIMS speeds the incremental process of creating a user interface;
  - A UIMS makes it possible to create prototypes that can be discussed with the end user;
  - A UIMS can adapt to different user profiles;
  - A UIMS makes the integration of new application functionalities easier;
  - A UIMS allows the application to be portable, while the user interface can be tailored to the particular environment;
  - A UIMS can ease the debugging of interactive applications.
- The set of programming and design techniques which are supposed to add another level of services for interactive system design beyond the toolkit level are user interface management systems, or UIMS for short. The term UIMS is used quite widely in both industrial and academic circles and has come to represent a variety of topics.

The main concerns of a UIMS, for our purposes, are:

- A conceptual architecture for the structure of an interactive system which concentrates
- On a separation between application semantics and presentation;
- Techniques for implementing a separated application and presentation whilst
- Preserving the intended connection between them;
- Support techniques for managing, implementing and evaluating a run-time
- interactive environment.

5.2.1 UIMS as a Conceptual Architecture

A major issue in this area of research is one of separation between the semantics ofthe application and the interface provided for the user to make use of that semantics.

There are many good arguments to support this separation of concerns:

- **Portability :** To allow the same application to be used on different systems it is best to consider its development separate from its device-dependent interface.
- **Reusability :** Separation increases the likelihood that components can be reused in order to cut development costs.
- **Multiple interfaces :** To enhance the interactive flexibility of an application, several different interfaces can be developed to access the same functionality.
- **Customization :** The user interface can be customized by both the designer and the user to increase its effectiveness without having to alter the underlying application.

Once we allow for a separation between application and presentation, we mustconsider how those two partners communicate. This role of communicationis referred to as *dialog control*. Conceptually, this provides us with the three majorcomponents of an interactive system: the application, the presentation and the dialogcontrol. In terms of the actual implementation, this separation may not be so clear.

MostUIMS fall into this class of external dialog control systems, since they promote, to agreater extent, the separation between presentation and application. They do not, however, all use the technique of callbacks as was demonstrated for theuse of toolkits.

Unit V End Sem.

The logical components of a UIMS were identified as

- **Presentation :** The component responsible for the appearance of the interface,including what output and input is available to the user.
- **Dialog control :** The component which regulates the communication between thepresentation and the application.
- **Application interface :** The view of the application semantics that is provided as theinterface.

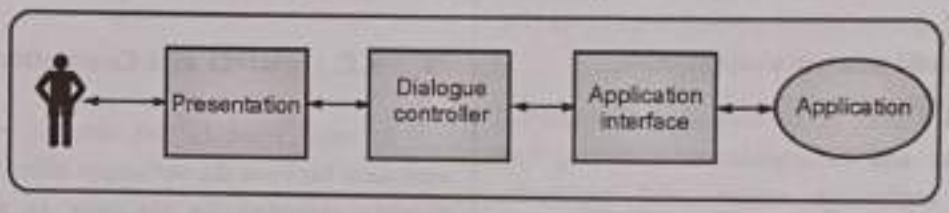


Fig. 5.2.1 : The Seeheim model of the logical components of a UIMS

- Fig. 5.2.1 presents a graphical interpretation of the Seeheim model. This included both application and user in Fig. 5.2.1 to place the UIMS model more in the context of the interactive system. The application and the user are not explicit in the Seeheim model.
- The UIMS approach does more than save programming time. It also helps you build a better interface, by maintaining consistency within the application and across applications under a common operating system, and by making it easier to rapidly iterate through the implement-and-test cycle.
- An additional advantage is that a UIMS can provide answers to the difficult question of what you can legally copy. The fundamental purpose of a UIMS is to help programmers develop interfaces rapidly, by copying controls that users already know.
- That goal is shared by programmers, users, and the vendor of the underlying operating system. The UIMS is sold by a company that has legal rights to the interface techniques you want to use, and purchasing the UIMS gives you clearly defined rights to sell the systems you develop.

5.3 GOALS OF EVALUATION

5.3.1 Evaluation

- Evaluation is the process of determining the worth of a program. A plan must be made to accurate evaluate any type of program, whether it is a training course or concerns the effectiveness of and customer satisfaction with a new product.
- Aim of evaluation is to test the functionality and usability of the design and to identify and rectify any problems. A design can be evaluated before any implementation work has started, to minimize the cost of early design errors.
- The role of evaluation in a design process to support the design of usable interactive systems. However, even if such a process is used, we still need to assess our designs and test our systems to ensure that they actually behave as we expect and meet user requirements.

5.3.2 Goals of Evaluation

This evaluation has three main goals:

- To assess the extent and accessibility of the system's functionality.
- To assess users' experience of the interaction.
- To identify anyspecific problems with the system.

(1) The system's functionality is important in that it must accord with the user'srequirements. In other words, the design of the system should enable users to performtheir intended tasks more easily. This includes not only making the appropriatefunctionality available within the system, but making it clearly reachable bythe user in terms of the actions that the user needs to

take to perform the task. It also involves matching the use of the system to the user's expectations of the task.

For example, if a filing clerk is used to retrieving a customer's file by the postal address,the same capability (at least) should be provided in the computerized file system.Evaluation at this level may also include measuring the user's performance with thesystem, to assess the effectiveness of the system in supporting the task.

(2) In addition to evaluating the system design in terms of its functional capabilities,it is important to assess the user's experience of the interaction and its impact uponhim. This includes considering aspects such as how easy the system is to learn, itsusability and the user's satisfaction with it. It may also include his enjoyment andemotional response, particularly in the case of systems that are aimed at leisure orentertainment. It is important to identify areas of the design that overload the userin some way, perhaps by requiring an excessive amount of information to be remembered,for example.

(3) The final goal of evaluation is to identify specific problems with the design. Thesemay be aspects of the design which, when used in their intended context, causeunexpected results, or confusion amongst users. This is, of course, related to both thefunctionality and usability of the design.

5.4 CATEGORIZATION OF EVALUATION TECHNIQUES

5.4.1 Types of Evaluation Methods

(1) Formative Evaluation (also known as 'evaluability assessment')

- Formative evaluation is used before program design or implementation. It generates data on the need for the program and develops the baseline for subsequent monitoring.
- It also identifies areas of improvement and can give insights on what the program's priorities should be. This helps project managers determine their areas of concern and focus, and increases awareness of your program among the target population prior to launch.

When:

- New program development
- Program expansion

What:

- The need for your project among the potential beneficiaries
- The current baseline of relevant indicators, which can help show impact later

Why:

- Helps make early improvements to the program
- Allows project managers to refine or improve the program

How:

Conduct sample surveys and focus group discussions among the target population focused on whether they are likely to need, understand, and accept program elements.

Questions to ask:

- Is there a need for the program?
- What can do to improve it?

(2) Process Evaluation(also known as 'program monitoring')

- Process evaluation occurs once program implementation has begun, and it measures how effective your program's procedures are.
- The data it generates is useful in identifying inefficiencies and streamlining processes, and portrays the program's status to external parties.

When:

- When program implementation begins
- During operation of an existing program

What:

- Whether program goals and strategies are working as they should
- Whether the program is reaching its target population, and what they think about it

Why:

- Provides an opportunity to avoid problems by spotting them early

Unit V End Sem.

- Allows program administrators to determine how well the program is working

How:

Conduct a review of internal reports and a survey of program managers and a sample of the target population. The aim should be to measure the number of participants, how long they have to wait to receive benefits, and what their experience has been.

Questions to ask:

- Who is being reached by the program?
- How the program is being implemented and what are the gaps? Is it meeting targets?

(3) Outcome Evaluation (also known as 'objective - based evaluation')

- Outcome evaluation is conventionally used during program implementation. It generates data on the program's outcomes and to what degree those outcomes are attributable to the program itself.
- It is useful in measuring how effective your program has been and helps make it more effective in terms of delivering the intended benefits.

When:

- After the program has run for some time period
- At an appropriate time to measure outcomes against set targets – usually benchmarked time periods

What:

- How much the program has affected the target population
- Clearly establish the degree of benefit provided by the program

Why:

- Helps program administrators tell whether a program is meeting its objectives
- Insights from outcome-focused feedback can help increase effectiveness

How : A randomized controlled trial, comparing the status of beneficiaries before and during the program or comparing beneficiaries to similar people outside of the program. This can be done through a survey or a focus group discussion.

Questions to ask:

- Did participants report the desired change after the implementation of the program?
- What are the short or long-term results reported by participants?

(4) Economic Evaluation(also known as 'cost analysis', 'cost-effectiveness evaluation', 'cost-benefit analysis', and 'cost-utility analysis')

- Economic evaluation is used during the program's implementation and looks to measure the benefits of the programs against the costs.
- Doing so generates useful quantitative data that measures the efficiency of the program. This data is like an audit, and provides useful information to sponsors and backers who often want to see what benefits their money would bring to beneficiaries.

When:

- At the beginning of a program, to remove potential leakages
- During the operation of a program, to find and remove inefficiencies.

What:

- What resources are being spent and where
- How these costs are translating into outcomes

Why:

- Program managers and funders can justify or streamline costs
- The program can be modified to deliver more results at lower costs

How : systematic analysis of the program by collecting data on program costs, including capital and man-hours of work. It will also require a survey of program officers and the target population to determine potential areas of waste.

Questions to ask:

- Where is the program spending its resources?
- What are the resulting outcomes?

(5) Impact Evaluation

- Impact evaluation studies the entire program from beginning to end (or at whatever stage the program is at), and looks to quantify whether or not it has been successful.
- Focused on the long-term impact, impact evaluation is useful for measuring sustained changes brought about by the program or making policy changes or modifications to the program.

When:

- At the end of the program
- At pre-selected intervals in the program

What:

- Assesses the change in the target population's well-being
- Accounts for what would have happened if there had been no program

Why:

- To show proof of impact by comparing beneficiaries with control groups
- Provides insights to help in making policy and funding decisions

How : A macroscopic review of the program, coupled with an extensive survey of program participants, to determine the effort involved and the impact achieved. Insights from program officers and suggestions from program participants are also useful, and a control group of non-participants for comparison is helpful.

Questions to ask:

- What changes in program participants' lives are attributable to your program?
- What would those not participating in the program have missed out on?

(6) Summative Evaluation

- Summative evaluation is conducted after the program's completion or at the end of a program cycle. It generates data about how well the project delivered benefits to the target population.
- It is useful for program administrators to justify the project, show what they have achieved, and lobby for project continuation or expansion.

When:

- At the end of a program
- At the end of a program cycle

What:

- How effectively the program made the desired change happen
- How the program changed the lives of program participants

Why:

- Provides data to justify continuing the program
- Generates insights into the effectiveness and efficiency of the program

How : Conduct a review of internal reports and a survey for program managers and target populations. The aim should be to measure the change that the project has brought about and compare the change to the costs.

Questions to ask

- Should the program continue to be funded?
- Should the program be expanded? If so, where? What factors worked in its favor and what worked against it?

(7) Goals-Based Evaluation (also known as 'objectively set evaluation)

- Goals-based evaluation is usually done towards the end of the program or at previously agreed-upon intervals.
- Development programs often set 'SMART' targets – Specific, Measurable, Attainable, Relevant, and Timely – and goals-based evaluation measures progress towards these targets.
- The evaluation is useful in presenting reports to program administrators and backers, as it provides them the information that was agreed upon at the start of the program.

When:

- At the end of the program
- At pre-decided milestones

What:

- How the program has performed on initial metrics
- Whether the program has achieved its goals

Why:

- To show that the program is meeting its initial benchmarks
- To review the program and its progress

How: This depends entirely on the goals that were agreed upon. Usually, goals-based evaluation would involve some survey of the participants to measure impact, as well as a review of input costs and efficiency.

Questions to ask:

- Has the program met its goals?
- Were the goals and objectives achieved due to the program or externalities?

Development programs with effective monitoring and evaluation frameworks use different types of evaluation at different points of time. Some programs might even run two different types of evaluation at the same time for entirely different purposes.

5.4.2 Types of Evaluation Techniques

- **Inspection methods (no users needed) : Evaluation based on expert analysis**
  - Heuristic evaluations
  - Walkthroughs
  - Other Inspections
- **User Tests (users needed) :Evaluation based on user participation**
  - Observations/Ethnography
  - Usability tests/ Controlled Experiments

There are 4 types of evaluations methods conducted by experts or system designers:

1. Cognitive walkthrough
2. Heuristic evaluation
3. Model based
4. Review based.

(1) Cognitive walkthrough

- "walkthrough" means a sequence of steps. E.g., "code walkthrough" is when a programmer shows her source code to others and explains, line by line, how the code works
- Cognitive walkthrough is when a user explains how she is using an interface, talking as sheperforms each action and explaining as she goes.
- Then, experts analyze the user's process.
- For each step:
  - is the effect of the user's action the same as the user's goal?
  - will the user see that the necessary action is available (in the interface)?
  - will the user know which widget in the interface enables the desired action?
  - after the action is performed (e.g., clicking on a button), will the user receive understandable feedback?
  - description of users' task(s)
  - list of user actions to accomplish the task(s)
  - profile(s) of target user(s) (i.e., expected background, experience, etc.)
- In order to conduct a cognitive walkthrough, experts need: – system specifications or system prototype

(2) Heuristic evaluation

- Typically performed on a design specification,but can be performed on a prototype or full system
- There are 10 heuristics that are frequently assessed
- Each is assessed on a scale of 0 to 4, as follows:

  0 = not a problem

  1 = cosmetic problem; fix if time

  2 = minor usability problem; low priority fix

  3 = major usability problem; important to fix

  4 = catastrophic usability problem; must be fixed before system/interface is released

The 10 heuristics are

1. visibility of system status (i.e., inform users with feedback)
2. match between system and real world (i.e., "speak" user's language; is system understandable by the user?)
3. user control and freedom (i.e., can user "undo" and "redo" actions?)
4. consistency and standards (does system follow standard conventions for symbols, menu items, etc?)
5. error prevention (does system make it hard for the user to make mistakes?)
6. recognition rather than recall (does system make it easy for the user to remember what to do? are there no or few commands to memorize?)
7. flexibility and efficiency of use (does the system let users tailor the interface to accommodate frequent actions or sequences of actions, e.g., macros?)
8. aesthetic and minimalist design (e.g., dialogues shouldn't contain extra information; interface shouldn't be cluttered)
9. help users recognize, diagnose and recover from errors (are error messages in clear language?)
10. help and documentation (does the system have any documentation and/or a help facility?)

(3) Model-based

- high-level models (such as GOMS) are used to predict user's performance
- low-level models (such as Fitts law keystroke, dialogue timing) are used to assess limits of user's performance.

(4) Review based

- Empirical models evolved from previous studies give indicators of generic interface elements
- For example, menu types, command name recall, color schemes, button labels, icondesigns, etc.

5.5 CHOOSING AN EVALUATION METHOD

- 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 ifapplied appropriately.
- Author Alen Dix explained in his book "Human Computer Interaction", about choosing an evaluation method, However, there are a number of factors that should be takeninto account when selecting evaluation techniques. These also provide a way of categorizing the different methods so that we can compare and choose between them.

5.5.1 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.

Design vs. implementation

- The first factor to affect our choice of evaluation method is the stage in the design process at which evaluation is required. The main distinction between evaluation of a design and evaluation of an implementation is that in the latter case a physical artifact exists.

Unit V End Sem.

- This may be anything from a paper mock-up to a full implementation, but it is something concrete that can be tested.Evaluation of a design, on the other hand, precedes this stage and seeks instead to provide information to feed the development of the physical artifact.
- Roughly speaking, evaluation at the design stage needs to be quick and cheap so might involve design experts only and be analytic, whereas evaluation of the implementation needs to be more comprehensive, so brings in users as participants.

☞ Laboratory vs. field studies

- Laboratory studies allow controlled experimentation and observation while losingsomething of the naturalness of the user's environment. Field studies retain thelatter but do not allow control over user activity.
- Ideally the design process shouldinclude both styles of evaluation, probably with laboratory studies dominating theearly stages and field studies conducted with the new implementation.

☞ Subjective vs. objective

- Evaluation techniques also vary according to their objectivity – some techniquesrely heavily on the interpretation of the evaluator, others would provide similarinformation for anyone correctly carrying out the procedure.
- The more subjectivetechniques, such as cognitive walkthrough or think aloud, rely to a large extent onthe knowledge and expertise of the evaluator, who must recognize problems andunderstand what the user is doing.
- They can be powerful if used correctly and willprovide information that may not be available from more objective methods.However, the problem of evaluator bias should be recognized and avoided.
- Oneway to decrease the possibility of bias is to use more than one evaluator. Objectivetechniques, on the other hand, should produce repeatable results, which are not dependent on the particular evaluator.

☞ Qualitative vs. quantitative measures

- The type of measurement provided by the evaluation technique is also an importantconsideration. There are twomain types: *quantitative measurement* and *qualitative measurement*.
- The former is usually numeric and can be easily analyzed using statistical techniques. The latter is non-numeric and is therefore more difficult to analyze, but can provide important detail that cannot be determined from numbers.
- The typeof measure is related to the subjectivity or objectivity of the technique, with subjectivetechniques tending to provide qualitative measures and objective techniques,quantitative measures.

☞ Information provided

- The level of information required from an evaluation may also vary. The informationrequired by an evaluator at any stage of the design process may range fromlow-level information to enable a design decision to be made (for example, whichfont is most readable) to higher-level information, such as 'Is the system usable?'.
- Some evaluation techniques, such as controlled experiments, are excellent at providinglow-level information – an experiment can be designed to measure a particularaspect of the interface.
- Higher-level information can be gathered using questionnaireand interview techniques, which provide a more general impression of the user'sview of the system.

☞ Immediacy of response

- Another factor distinguishing evaluation techniques is the immediacy of theresponse they provide. Some methods, such as think aloud, record the user'sbehavior at the time of the interaction itself.
- Others, such as post-task walkthrough, rely on the user's recollection of events. Such recollection is liable to suffer frombias in recall and reconstruction, with users interpreting events according to their preconceptions.

- Recall may also be incomplete. However, immediate techniques canalso be problematic, since the process of measurement can actually alter the way theuser works.

☞ Intrusiveness

- Related to the immediacy of the response is the intrusiveness of the technique itself.Certain techniques, particularly those that produce immediate measurements, areobvious to the user during the interaction and therefore run the risk of influencingthe way the user behaves.
- Sensitive activity on the part of the evaluator can help toreduce this but cannot remove it altogether. Most immediate evaluation techniquesare intrusive, with the exception of automatic system logging.

☞ Resources

- The final consideration when selecting an evaluation technique is the availabilityof resources. Resources to consider include equipment, time, money, participants, expertise of evaluator and context.
- Some decisions are forced by resource limitations:it is not possible to produce a video protocol without access to a video camera.

⏭ 5.6 DECIDE

The DECIDE framework can be used to structure your evaluation

1. Determine the overall goals that the evaluation addresses
2. Explore the specific questions to be answered
3. Choose the evaluation paradigm and techniques to answer the questions
4. Identify practical issues that must be addressed, such as selecting participants
5. Decide how to deal with ethical issues
6. Evaluate, interpret and present the data

☞ D - Determine the goals

- **Introduction :** Usability Testing for the improvement of Reservation System Application
- **GOAL :** To examine the usability of the current application in the perspective of students who used and never used the system.
- To evaluate effectiveness of content presentation in the application.
- Focus on : First glance, Content presentation, Implementation

☞ E - Explore the questions

Question

- We have to ask the basic information of the prospective users about using the reservation system including
- Have the users ever used the Reservation system before ?
- What type of room that the users reserved ?
- Which channel did the users choose to reserve ?
- Do the users know that System has Online Reservation system ?
- Then, we will ask the questions that related to using of our application
- Can the prospective users reserve required room ?
- Can the prospective users Cancel the room that have reserved ?
- Can the prospective users reserve room from the History Reservation Page ?

☞ C - Choose the evaluation approach and methods

- In the part of the evaluation approach, we had interview prospective users in Pune University including Pune Learning Center around True Coffee and location which convenient to participant e.g., In area around the faculty of participants.
- Furthermore, we also recorded videos and voices of participants while we were interviewing our participants for apply in usability testing.Before, we recorded videos we had given the video consent form to our participants for assure that participant will allow us to record the videos of them.

Unit V End Sem.

- After they approved the video consent form, we will explain briefly about our project proposal and the application we created. Then, we will describe what features we consists in the prototype and how to use it.
- Then, we will give our prototype of application to participants for trying to use it and we ask them try to using Reservation System Application prototype.

☞ I - Identify the practical issues

Participants

- Undergraduate students in Pune University.
- One or Two participants from each faculty who have been using or not using the reservation system before.
- Recruit from university students via Mobile phone list.
- Provide an Android smartphone with the application prototype installed.
- Make an appointment with the participant.

☞ D - Decide to the ethical issues

- Before we recorded video, We have to give the video consent form to the participants to assure that they will allow us for recording the videos. In addition, we will tell to participants for their permission and our ethical responsibility including
- Explained the purpose and process of the test.
- Ask the participant to sign the consent form if she or he accepted.
- Ensure the privacy of participant's personal information.
- Ask for the participant's permission for video record.
- Allow the participant to leave anytime.

☞ E - Evaluation, Analyze, Interpret and Present the data

For the Evaluation part, we created the evaluation form to measure the performance, satisfaction rate from users, and level of the complexity of system by using usability specification table an record the statistic for each participants.

⏭ 5.7 HEURISTIC EVALUATION

- Heuristic evaluation is a process where experts use rules of thumb to measure the usability of user interfaces in independent walkthroughs and report issues. Evaluators use established heuristics (e.g., Nielsen-Molich's) and reveal insights that can help design teams enhance product usability from early in development.
- Heuristic evaluation is an assessment of a product's user interface, and its purpose is to detect usability issues that may occur when users interact with a product and identify ways to resolve them. Heuristic evaluation, developed by Jakob Nielsen and Rolf Molich, is a method for structuringthe 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 forevaluating 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.
- Each evaluator assesses the system and notes violations of any of these heuristics that would indicate a potential usability problem. The evaluator also assesses the severity of each usability problem, based on four factors: how common is the problem, how easy is it for the user to overcome, will it be a one-off problem or a persistent one, and how seriously will the problem be perceived?

These can be combined into an overall severity rating on a scale of 0–4:

0 = I don't agree that this is a usability problem at all

1 = Cosmetic problem only: need not be fixed unless extra time is available on project

2 = Minor usability problem: fixing this should be given low priority

3 = Major usability problem: important to fix, so should be given high priority

4 = Usability catastrophe: imperative to fix this before product can be released (Nielsen)

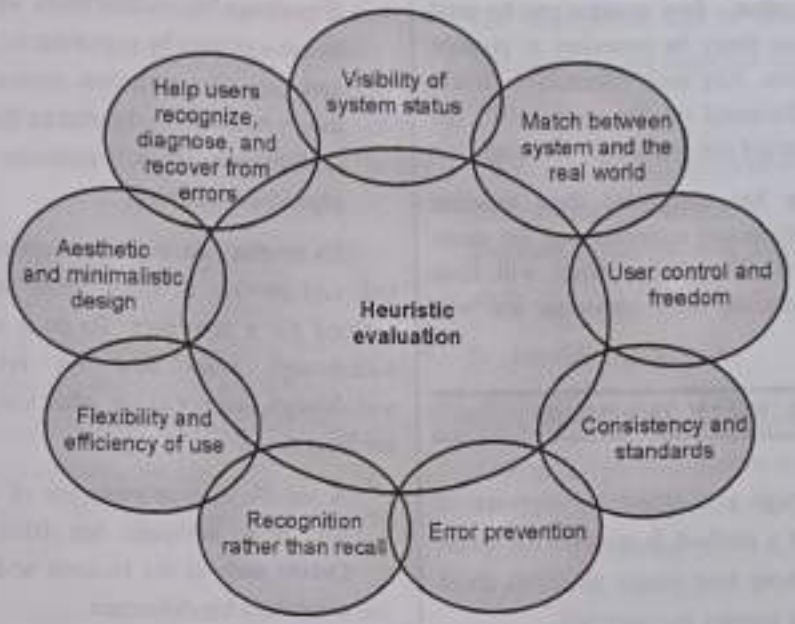


Fig. 5.7.1 : Heuristic Evaluation

5.7.1 Nielsen's Ten Heuristics are

UQ. Explain Nielsen's ten heuristics.
(SPPU - Q. 7(b), May 18, 8 Marks)

1. **Visibility of system status :** Always keep users informed about what is going on, through appropriate feedback within reasonable time. For example, if a systemoperation will take some time, give an indication of how long and how much iscomplete.
2. **Match between system and the real world :** The system should speak the user'slanguage, with words, phrases and concepts familiar to the user, rather thansystem-oriented terms. Follow real-world conventions, making informationappear in natural and logical order.
3. **User control and freedom :** Users often choose system functions by mistakeand need a clearly marked 'emergency exit' to leave the unwanted state withouthaving to go through an extended dialog. Support undo and redo.
4. **Consistency and standards** Users should not have to wonder whether words, situations or actions mean the same thing in different contexts. Follow platformconventions and accepted standards.
5. **Error prevention** Make it difficult to make errors. Even better than good errormessages is a careful design that prevents a problem from occurring in the firstplace.
6. **Recognition rather than recall** Make objects, actions and options visible. Theuser should not have to remember information from one part of the dialog toanother. Instructions for use of the system should be visible or easily retrievablewhenever appropriate.
7. **Flexibility and efficiency of use** Allow users to tailor frequent actions.Accelerators – unseen by the novice user – may often speed up the interactionfor the expert user to such an extent that the system can cater to both inexperiencedand experienced users.
8. **Aesthetic and minimalist design :** Dialogs should not contain information thatis irrelevant or rarely needed. Every extra unit of information in a dialog competeswith the relevant units of information and diminishes their relative visibility.
9. **Help users recognize, diagnose and recover from errors :** Error messages shouldbe expressed in plain language (no codes), precisely indicate the problem, andconstructively suggest a solution.

Unit V End Sem.

10. **Help and documentation :** Few systems can be used with no instructions so itmay be necessary to provide help and documentation. Any such information should be easy to search, focussed on the user's task, list concrete steps to becarried out, and not be too large.

Once each evaluator has completed their separate assessment, all of the problemsare collected and the mean severity ratings calculated. The design team will then determine the ones that are the most important and will receive attention first.

5.8 COGNITIVE WALK THROUGH

- A cognitive walkthrough is a structured approach to evaluating usability of a product. It involves the tester, who is not a user, asking four simple questions about the way a specific user journey is conducted.
- They will record the outcomes of these questions, in their opinion, and use these observations to improve the product further. Cognitive walk throughs are used to evaluate a product's usability.
- It focuses on the new user's perspective by narrowing the scope to tasks needed to complete specific user goals. It was created in the early 90's by Cathleen Wharton, John Rieman, Clayton Lewis, Peter Polson.

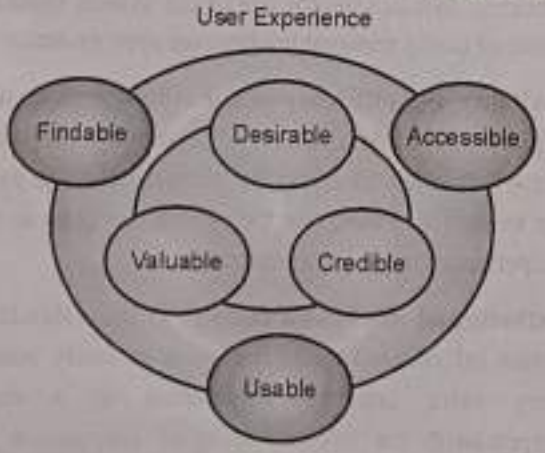


Fig. 5.8.1 : Cognitive Walkthrough

- 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 orexamination of a user's manual. So the checks that are made during the walkthrough hask questions that address this exploratory learning.

To do this, the evaluators gothrough each step in the task and provide a 'story' about why that step is or is notgood for a new user. To do a walkthrough (the term walkthrough from now on refersto the cognitive walkthrough, and not to any other kind of walkthrough), you needfour things:

1. A specification or prototype of the system. It doesn't have to be complete, but itshould be fairly detailed. Details such as the location and wording for a menu canmake a big difference.
2. A description of the task the user is to perform on the system. This should be arepresentative task that most users will want to do.
3. A complete, written list of the actions needed to complete the task with the proposedsystem.
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 to critique the system and tell a believable story about itsusability. To do this, for each action, the evaluators try to answer the following four questions for each step in the action sequence.

1. **Is the effect of the action the same as the user's goal at that point?**

   Each useraction will have a specific effect within the system. Is this effect the same as whatthe user is trying to achieve at this point? For example, if the effect of the actionis to save a document, is 'saving a document' what the user wants to do?

2. **Will users see that the action is available?**

   Will users see the button or menu item, for example, that is used to produce the action? This is *not* asking whether they will recognize that the button is the one they want.

This is merely asking whetherit is visible to them at the time when they will need to use it. Instances wherethe answer to this question might be 'no' are, for example, where a VCR remotecontrol has a covered panel of buttons or where a menu item is hidden away in asubmenu.

3. **Once users have found the correct action, will they know it is the one they need?**

This complements the previous question. It is one thing for a button or menuitem to be visible, but will the user recognize that it is the one he is looking for tocomplete his task? Where the previous question was about the visibility of theaction, this one is about whether its meaning and effect is clear.

4. **After the action is taken, will users understand the feedback they get?**

If you nowassume that the user did manage to achieve the correct action, will he know thathe has done so? Will the feedback given be sufficient confirmation of what hasactually happened? This is the completion of the execution–evaluation interactioncycle. In order to determine if they have accomplished theirgoal, users need appropriate feedback.

5.8.1 How to Conduct a Cognitive Walkthrough

At its core, a cognitive walkthrough has three parts:

1. Identify the user goal you want to examine
2. Identify the tasks you must complete to accomplish that goal
3. Document the experience while completing the tasks

1) Identifying the user goal

- A user goal is a big, overarching objective and doesn't include specific, step-by-step tasks. From a user's point-of-view, it doesn't necessarily matter how the goal is accomplished, as long as it gets completed.
- For example, I host a dinner party every month. Beforehand, I ask everyone invited to send me 10 songs they love.
- Then, I use Spotify to create a playlist of those songs to play during the party. As a user, my goal here is to create a playlist with others to play at my dinner party.

2) Identifying the tasks

I'll note here I'll only be walking through one possible path of accomplishing these goals. Spotify has a number of ways to accomplish these goals. Ideally, you'd identify the optimal path and tasks for each interface. However, in this article, I'll only be walking through one path.

Goal : Create a Playlist

- Open Spotify web player
- Enter user name in user name field
- Enter password in password field
- Click the login button
- Click the your library section
- Click the new playlist button
- Type a name into the playlist name field
- Click the create button

Goal : Add a track to the playlist

- Click search icon
- Enter track name into the field
- Click tracks tab
- Find track in results
- Hover over track
- Click "..."
- Click "add to playlist"
- Select playlist

Unit V End Sem.

3) Documenting the experience

Since experience is subjective, it's important to structure how an evaluator documents it so that all walkthroughs use the same criteria. Traditionally, the evaluator asks/answers four questions during each task.

- Will users understand how to start the task?
- Are the controls conspicuous?
- Will users know the control is the correct one?
- Was there feedback to indicate you completed (or did not complete) the task?

UQ. Complete the cognitive walk-through example for the video remote control design.

(SPPU - Q. 9(a), Dec. 19, 9 Marks)

☑ Ans. :

We will assume that the user is familiar with VCRs but not with this particular design.

- The next step in the walkthrough is to identify the action sequence for this task.
- We specify thisin terms of the user's action (UA) and the system's display or response (SD). The initial display isas the left-hand picture in Figure

UA 1 : Press the 'timed record' button

SD 1 : Display moves to timer mode. Flashing cursor appears after 'start:'

UA 2 : Press digits 1 8 0 0

SD 2 : Each digit is displayed as typed and flashing cursor moves to next position

UA 3 : Press the 'timed record' button

SD 3 : Flashing cursor moves to 'end:'

UA 4 : Press digits 1 9 1 5

SD 4 : Each digit is displayed as typed and flashing cursor moves to next position

UA 5 : Press the 'timed record' button

SD 5 : Flashing cursor moves to 'channel:'

UA 6 : Press digit 4

SD 6 : Digit is displayed as typed and flashing cursor moves to next position

UA 7 : Press the 'timed record' button

SD 7 : Flashing cursor moves to 'date:'

UA 8 : Press digits 2 4 0 2 0 5

SD 8 : Each digit is displayed as typed and flashing cursor moves to next position

UA 9 : Press the 'timed record' button

SD 9 : Stream number in top right-hand corner of display flashes

UA 10: Press the 'transmit' button

SD 10: Details are transmitted to video player and display returns to normal mode

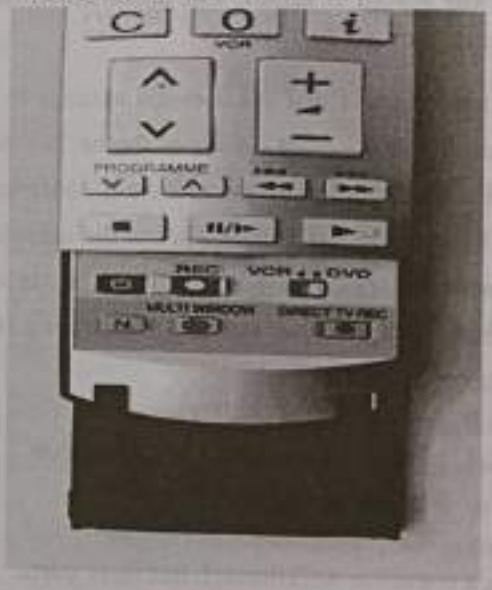


Fig. 5.8.2 : VCR Remote Control

- Having determined our action list we are in a position to proceed with the walkthrough. For eachaction (1–10) we must answer the four questions and tell a story about the usability of the system.

☞ Beginning with UA 1

UA 1 : Press the 'timed record' button

Question 1: Is the effect of the action the same as the user's goal at that point?

The timed record button initiates timer programming. It is reasonable to assume that a userfamiliar with VCRs would be trying to do this as his first goal.

Question 2 : Will users see that the action is available?

The 'timed record' button is visible on the remote control.

Question 3: Once users have found the correct action, will they know it is the one they need?

It is not clear which button is the 'timed record' button. The icon of a clock (fourth buttondown on the right) is a possible candidate but this could be interpreted as a button to changethe time. Other possible candidates might be the fourth button down on the left or the filledcircle (associated with record). In fact, the icon of the clock is the correct choice but it is quitepossible that the user would fail at this point. This identifies a potential usability problem.

Question 4 : After the action is taken, will users understand the feedback they get?

Once the action is taken the display changes to the timed record mode and shows familiar headings(start, end, channel, date). It is reasonable to assume that the user would recognize theseas indicating successful completion of the first action. So we find we have a potential usability problem relating to the icon used on the 'timed record'button. We would now have to establish whether our target user group could correctly distingaishthis icon from others on the remote.

The analysis proceeds in this fashion, with a walkthrough form completed for each action. We willleave the rest of the walkthrough for you to complete as an exercise. What other usability problemscan you identify with this design?

5.9 USABILITY TESTING

5.9.1 What is Usability Testing ?

- Usability testing is a method of evaluating a website's or app's readiness for release by testing it with real users who are part of the target audience.
- Usability tests evaluate the overall user experience by measuring the relative ease with which end users can accomplish a set of tasks that a typical user of the app or website would need to accomplish.
- Usability testing is the practice of testing how easy a design is to use with a group of representative users. It usually involves observing users as they attempt to complete tasks and can be done for different types of designs. It is often conducted repeatedly, from early development until a product's release.

5.9.2 Why is Usability Testing Important?

- The goal of usability testing is to understand how real users interact with your website and make changes based on the results.

Fig. 5.9.1 : Why usability test?

- It is important to be sure that your app or website is easy to navigate and that tasks can be completed with ease; otherwise, people will leave and go to a competitor's site.
- The primary purpose of a usability test is to gather the data needed to identify usability issues and improve a website's or app's design.
- Even the best web design and development teams can benefit from usability testing as the tests indicate trouble spots for users and the areas where they are getting stuck or confused.

5.9.3 Components of a Usability Test

- In a usability test, there are two groups: **end users and observers**. Ideally, the two groups do not know one another, so the observers can gather more objective data.
- If you are setting up a usability test, you construct a scenario for users to accomplish a set of tasks – ones

Unit V End Sem.

that a new visitor to your website would need to accomplish – like signing up or inviting a friend or making a purchase.

- From there, user researchers will recruit participants to create a focus group compromising users that are ideally in the target market for their product. The users try to accomplish this set of tasks under controlled conditions.
- While the users try to complete tasks, the observers watch and/or measure their overall success in accomplishing those goals and look to identify usability problems. They can either take notes while observing or record the session with audio or video for convenient recall.
- The observers take note of where the users succeed and where they have trouble, so they can revisit their designs at a later date and make improvements.

5.9.4 Methods of Usability Testing

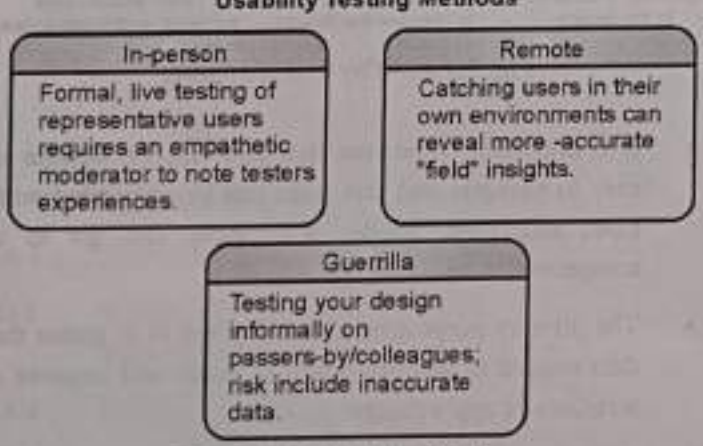


Fig. 5.9.2 : Usability Testing

The following are common methods of usability testing:

Card sorting

- Often used for testing a taxonomy or navigation structure, users organize sets of items into groups and give names or labels to them.
- This type of testing is informative for determining what to call various user-interfaces, screens, pages or functions and how to group them.

In-person testing

- This type of test is run by one or more observers in a fixed environment such as a conference room, either with small groups or individuals.
- Users are asked to accomplish a set of tasks and the observer can interact with them at any point to ask questions or to probe further.
- This type of usability evaluation is often done as part of a larger usability study by the user research team; however, any of these methods can be used in a larger usability study.

Remote testing

- In remote testing, moderators will create a test plan and ask users to conduct a series of tasks in their own environment – and their attempts to accomplish tasks are often recorded via a browser webcam.
- This type of user testing can be done either with a moderator (using webinar or conference call technology) or as a self-guided test.

Guerilla testing

- Guerilla testing works best in the early stages of the product development process. When you have a tangible design (wireframes or lo-fi prototypes) and what to know whether you're moving in the right direction or not.
- Guerilla testing is also good for collecting personal opinions and emotional impressions about ideas and concepts.

A/B Testing

- A/B testing is a type of testing methodology that doesn't involve simulated experiences or observation; it puts two live variations of a website or app to the test and sends half of the traffic to one and half to the other, tallying the data for which variation had a higher conversion rate.

5.9.5 Usability Testing is an Iterative Process

To make usability testing work best, you should:

(1) Plan

(a) Define what you want to test. Ask yourself questions about your design/product. What aspect/s of it do you want to test? You can make a hypothesis from each answer. With a clear hypothesis, you'll have the exact aspect you want to test.

(b) Decide how to conduct your test – e.g., remotely. Define the scope of what to test (e.g., navigation) and stick to it throughout the test. When you test aspects individually, you'll eventually build a broader view of how well your design works overall.

(2) Recruit Participants:

Whom you recruit, and how, depends on your testing goals (for example, how much information you want and therefore how long your sessions need to be) and your budgetary constraints. The most popular ways to find participants for your study:

(a) **Hire an agency :** if you're looking for a very specific subsection of the population (like web-savvy oncologists, or single mothers under 35), the most efficient way to find them is to hire a specialized recruitment agency. These companies have vast resources for finding desirable candidates and can do so very efficiently.

(b) **Use your website :** if you already have an established user base, recruit people there. Use a pop-up poll (Hotjar can help with this) to find users who are willing to participate.

(c) **Use social media :** if you have a social media following, use your social channels to reach out to potential participants.

(d) **Recruit your clients :** reach out to your clients/customers directly and ask if they would be willing to help (provided they've given you consent to be contacted for these initiatives. You don't want to spam them unnecessarily!).

(3) Set user tasks

(a) Prioritize the most important tasks to meet objectives (e.g., complete checkout), no more than 5 per participant. Allow a 60-minute timeframe.

(b) Clearly define tasks with realistic goals.

(c) Create scenarios where users can try to use the design naturally. That means you let them get to grips with it on their own rather than direct them with instructions.

(4) Conduct the test

- Know who your users are as a target group. Use screening questionnaires (e.g., Google Forms) to find suitable candidates.
- You can advertise and offer incentives. You can also find contacts through community groups, etc. If you test with only 5 users, you can still reveal 85% of core issues.

(5) Facilitate/Moderate testing

- Set up testing in a suitable environment. Observe and interview users. Notice issues. See if users fail to see things, go in the wrong direction or misinterpret rules.
- When you record usability sessions, you can more easily count the number of times users become confused. Ask users to think aloud and tell you how they feel as they go through the test.

Unit V End Sem.

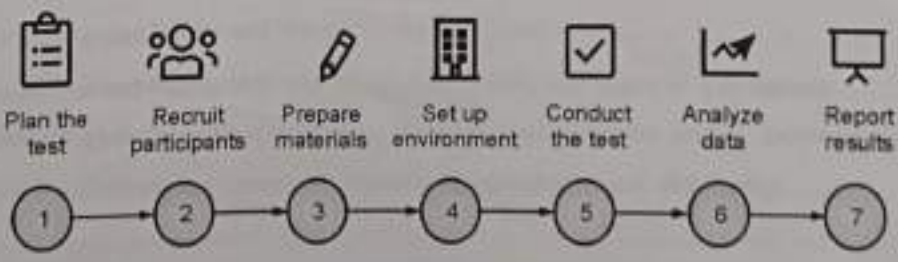


Fig. 5.9.3 : Steps of Usability Testing

- From this, you can check whether your designer's mental model is accurate: Does what you think users can do with your design match what these test users show?

5.9.6 Advantages and Disadvantages

Advantages

(1) There are many advantages of usability testing including:

- Feedback direct from the target audience to focus the project team
- Internal debates can be resolved by testing the issue to see how users react to the different options being discussed
- Issues and potential problems are highlighted before the product is launched

(2) The business advantages of usability testing can be seen at the end of the project:

- It increases the likelihood of usage and repeat usage
- It minimises the risk of the product failing
- Users are better able to reach their goals, which results in the business meeting its targets

Disadvantages

(1) Usability testing provides many benefits, but there are a few disadvantages in using this methodology, which should be noted.

(2) Firstly, testing is not 100% representative of the real-life scenario, e.g. a mother will not have her two young children running around like she might have at home.

Also, usability testing is mainly qualitative, so does not provide the large samples of feedback that a questionnaire might, but the feedback can be far more accurate and insightful.

Chapter Ends...

□□□

UNIT VI

CHAPTER 6

Future Trends

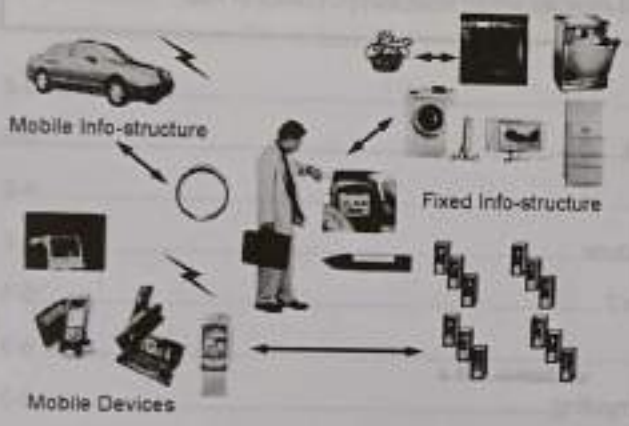
Syllabus

Ubiquitous Computing, Design thinking, Finding things on web, Augmented Reality, Virtual Reality, Challenges in designing interfaces for smart homes, smart devices, handheld devices, smart wrist watch, Future of HCI

6.1 UBIQUITOUS COMPUTING

GQ. Explain Ubiquitous Computing in detail. **(8 Marks)**

- Ubiquitous computing is a paradigm in which the processing of information is linked with each activity or object as encountered. It involves connecting electronic devices, including embedding microprocessors to communicate information. Devices that use ubiquitous computing have constant availability and are completely connected.
- Ubiquitous computing focuses on learning by removing the complexity of computing and increases efficiency while using computing for different daily activities. Ubiquitous computing is also known as pervasive computing.



(1F1)**Fig. 6.1.1 : Ubiquitous Computing**

- Pervasive computing or ubiquitous computing, is the growing trend of embedding computational capability (generally in the form of microprocessors) into everyday objects to make them effectively communicate and perform useful tasks in a way that minimizes the end user's need to interact with computers as computers. Pervasive computing devices are network-connected and constantly available.
- Unlike desktop computing, pervasive computing can occur with any device, at any time, in any place and in any data format across any network and can hand tasks from one computer to another.
- The main focus of ubiquitous computing is the creation of smart products that are connected, making communication and the exchange of data easier with less interfacing.

6.1.1 Key features of Ubiquitous Computing include

- Focus on many-to-many relationships, instead of one-to-one, many-to-one or one-to-many in the environment, along with the idea of technology, which is constantly present.
- Consideration of the human factor and placing of the paradigm in a human, rather than computing, environment
- Use of inexpensive processors, thereby reducing memory and storage requirements
- Capturing of real-time attributes
- Relies on converging Internet, wireless technology and advanced electronics
- Totally connected and constantly available computing devices
- Includes local/global, social/personal, public/private and invisible/visible features and considers knowledge creation, as well as information spread.
- Increased surveillance and possible restriction and interference in user privacies, as the digital devices are wearable and constantly connected

6.1.2 Pervasive Computing Devices have Evolved to include

1. Laptops
2. Tablets
3. Notebooks
4. Smartphones
5. Wearable devices
6. Sensors (for example, Smart home appliances).

6.1.3 How Ubiquitous Computing is used in Applications ?

- Pervasive or ubiquitous computing applications have been designed for consumer use and to help people to do their jobs.

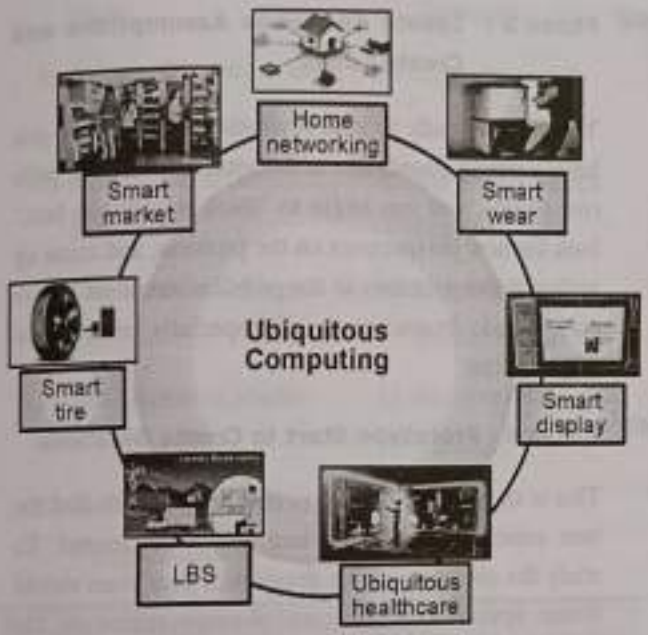


Fig. 6.1.2 : Ubiquitous Computing uses in Smart systems

- An example of pervasive computing is an Apple Watch that alerts the user to a phone call and allows the call to be completed through the watch. Another example is when a registered user for Audible, Amazon's audio book server, starts his or her book using the Audible app on a smartphone on the train and continues listening to the book through Amazon Echo at home.
- An environment in which devices, present everywhere, are capable of some form of computing can be considered a ubiquitous computing environment. Industries spending money on research and development (R&D) for ubiquitous computing include the following :
  - Healthcare
  - Military
  - Entertainment
  - logistics
  - Logistics
  - Energy
  - healthcare
  - military

6.1.4 Importance of Ubiquitous Computing

- Because pervasive computing or ubiquitous computing systems are capable of collecting, processing and communicating data, they can adapt to the data's context and activity.
- That means, in a network that can understand its surroundings and improve the human experience and quality of life.

6.1.5 Advantages and Disadvantages of Ubiquitous Computing

Advantages

1. Manage information quickly, efficiently and effortlessly.
2. Remove complexity of new technology.
3. Smart Environments will be embedded with computing technology.
4. Invisible.
5. Socialization.
6. Decision Making.

Disadvantages

1. Ubiquitos Computing is not entirely secure
2. Frequent line connection that are broken
3. Slow connections.
4. Very expensive operating cost.

6.2 DESIGN THINKING

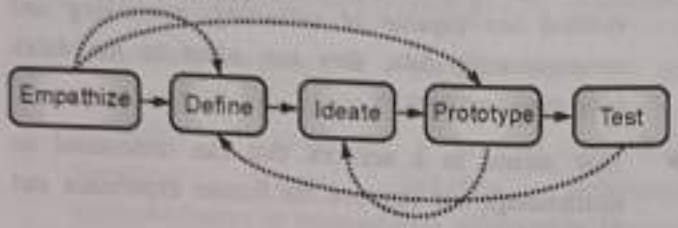
GQ. What is Design Thinking? **(5 Marks)**

GQ. Explain five stages of design thinking process. **(10 Marks)**

- Design thinking is a non-linear, iterative approach that allows teams to better understand their customers, challenge assumptions, redefine challenges, prototype and test novel solutions.

UNIT VI End Sem.

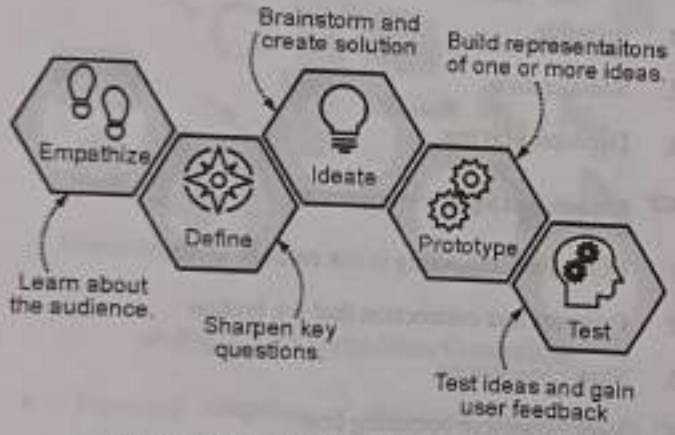
- It is especially useful for tackling unknown challenges, as it involves five phases: Empathize, Define, Ideate, Prototype and Test.



(IF3)Fig. 6.2.1 : Phases of Design Thinking

- Design thinking gives teams the freedom to come up with novel ideas. Your team can use it to get behind hard-to-find information and apply a variety of hands-on ways to help uncover unique solutions.
- These phases are not usually in order, and teams frequently run them in parallel, out of order, and iteratively repeat them.

6.2.1 The Phases (Stages) of Design Thinking Process



(IF4)Fig. 6.2.2 : Design Thinking Process

Phase 1 : Empathize Research Users' Needs

Typically, user research will help you get an empathic grasp of the problem you're trying to solve. Empathy is essential in a human-centered design approach like design thinking because it allows you to put your own ideas about the world aside and obtain a true understanding of users and their needs.

Phase 2 : Define State Users' Needs & Problems

It's time to compile all of the data obtained during the Empathize stage. You then synthesis and analyse your observations to define the fundamental challenges you and your team have uncovered. Problem statements are the names for these definitions. Before moving on to brainstorming, you can use personas to maintain your efforts human-centered.

Phase 3 : Ideate Challenge Assumptions and Create Ideas

You're now ready to start brainstorming. Because you have a strong foundation of information from the prior two phases, you can begin to "think outside the box," look for new perspectives on the problem, and come up with creative solutions to the problem statement you've constructed. Brainstorming is especially effective in this situation.

Phase 4 : Prototype Start to Create Solutions

This is the start of a test period. The goal is to find the best solution to each problem that is discovered. To study the concepts you've developed, your team should create several low-cost, scaled-down copies of the product (or certain aspects inside the product). Paper prototyping could be used in this case.

Phase 5 : Test Try Your Solutions Out

- Prototypes are thoroughly tested by evaluators. Despite the fact that this is the final phase, design thinking is iterative: teams frequently use the outcomes to redefine one or more problems.
- As a result, you can go back to earlier phases to make more iterations, changes, and improvements and to find or rule out alternate ideas.

6.3 FINDING THINGS ON WEB

- Searching the internet can be a frustrating business. You enter a word or a phrase into a search engine and up comes a stack of irrelevant information. Because the Web is so vast, you need to identify ways to find things on the web or Internet.
- A search engine is your best tool to find what you need on the Internet. What you need is the ability to refine your search to get exactly what you want.

- These following steps that you can take to pinpoint specific information online. All examples below refer to the Google search engine.

1. Lost in hyperspace
2. Designing structure
3. Making navigation easier
4. History, Bookmarks, etc.
5. Indices, Directories and Search
6. Complex search
7. Finding research literature
8. Vary Your Search Engine
9. Use Specific Keyword
10. Simplify Your Search Terms
11. Use Quotation Marks
12. Remove Unhelpful Words
13. Refine Your Search Using Operators

6.4 AUGMENTED REALITY

- Augmented reality (AR) is an experience where designers enhance parts of users' physical world with computer-generated input.
- Designers create inputs ranging from sound to video, to graphics to GPS overlays and more in digital content which responds in real time to changes in the user's environment, typically movement.
- Augmented reality is enhancing the view of reality by supplementing virtual objects using technology. Using AR technology, the environment around a person can become much more interactive and digital. Apart from sense of sight AR applies to all senses, such as hearing, smell, and touch.
- The most common example seen nowadays is the cricket score that we see on our television screen. It may also include removal of real objects from real environment. Supplementing of real objects in virtual environment is called Augmented Virtuality.
- Augmented reality takes a real world scene with the help of camera on device and superimposes images, videos or sounds on the real world scene.
- AR works in two ways, first based on positioning of markers which is identified by the software on the device and then the content hidden in the marker is displayed and second way is to identify the location of device through GPS and displays the content according to the field of view of the device.
- Augmented Reality, involves a live direct or indirect experience of an environment, overlaid with computer-generated sensory input usually in the form of graphics, video and/or sound.

(TF5) **Fig. 6.4.1: Augmented Reality**

6.5 VIRTUAL REALITY

GQ. What is Augmented Reality and Virtual Reality? **(6 Marks)**

GQ. Explain Augmented Reality (AR) and Virtual Reality (VR) with example. **(10 Marks)**

- Virtual Reality involves providing sensory input to a user that replicates being present in a real or imagined environment.
- Most commonly the sensory input is limited to sight and sound, but it can also include other senses such as touch.

UNIT VI End Sem

Fig. 6.5.1 : Virtual Reality

- VR is a three-dimensional computer-generated environment which a person can explore and interact with. Put simply, VR allows you to experience situations through computers that don't really exist, such as living in an alternate world, like Mars, or reenacting a historical battle. To be considered VR, the following criteria must be met:
  - **Believable** : Feels like living in the fantasy (virtual) world.
  - **Interactive** : As you move around, the VR environment needs to move with you.
  - **Explorable** : A VR world needs to be large and detailed enough to explore.
  - **Immersive** : To be both believable and interactive, VR needs to engage both body and mind.
- **Applications of VR and AR include** : education – enhanced learning experiences; medical and healthcare – treatments for PTSD, phantom pain, anxieties and phobias, autism in children; support for complex tasks such as surgery, equipment assembly, or maintenance and repair by adding relevant information to the field of view of the user; training for medical personnel, law enforcement, military, and emergency responders; architectural design – experiencing a virtual building before it's built; engineering and design; telepresence – for meetings and remote workers; market research – experiencing a virtual product that doesn't yet exist; entertainment – cinema, music, and sports; tourism; product advertising and promotion; computer games.

6.6 CHALLENGES IN DESIGNING INTERFACES FOR SMART HOMES, SMART DEVICES, HANDHELD DEVICES, SMART WRISTWATCH.

GQ. What are the Challenges in designing interfaces for smart devices? **(10 Marks)**

Fig. 6.6.1 : Smart Home devices

(TF6)Fig. 6.6.2 : Handheld devices

1. Meeting Requirements with UI/UX Design

- User Interface (UI) and User Experience (UX) are required factors for any app. Without having proper UI/UX, we cannot create a proper web application. Ultimately, a user-friendly UI/UX will help in attracting more users to the application.

- Hence, while establishing the app's documentation, it is vital to list the primary purpose of the app, the devices it will support, what platform will be used to launch, and so on.
- Once all the primary requirements are listed in detail with features and functionality, the designer should initiate their work. UI/UX's essential elements need to develop that offer accessibility and a smooth surfing experience correctly.
- Sometimes because of time constraints, unclear requirements, or some other reason, UI/UX is overlooked. However, it should not be the case while developing a mobile app for a smart device. This challenge can be overcome with precise documentation and clarity on the functions of an application.

2) Development Technology

- Choosing the best suitable development technology and building a native, hybrid or Cross-platform mobile application is the most common development challenge that most companies encounter. This final decision should be taken as per the business requirements and users' preferences.
- However, asking for expert advice from an IoT app development services provider would be a plus. Build the app on a flexible platform to be adaptable and best suit the client's target functionality.

3) Assuring Compatibility between Sensors and Networks

- Sensors and networks via which they communicate are an indispensable part of IoT development. One challenging aspect for many IoT specialists in determining the terminal compatibility between different sensors and various network types.
- These days, many sensor manufacturers and vendors do not perpetually work under similar protocols. Consequently, you may end up in a circumstance where two devices are not compatible and exchange valuable data.
- Sometimes, these challenging issues are encountered later or at the end of the app development process. Make sure to verify these details with the app development company you have selected.

4) Validating Hardware Compatibility

- It is challenging and complicated to anticipate the system and performance requirements while the app development is in the early phase.
- Often, the development teams do not get specific hardware information. This issue needs to be resolved ASAP to prevent issues with the app functioning later on.
- Thus, some reputed IoT app development companies emphasize choosing the right hardware from the early stage of development to modify their design and/or find ways to define hardware requirements.

5) Managing Devices Connection to the Network

- The provocation of connectivity and networking remains one of the most significant matters when there is a requirement to connect different devices to work within a single app.
- The devices can be united using one of the classic internet networks (such as LAN or WAN) or other networks.
- As the IoT system and its architecture evolve, networks' demand increases along with the connected devices. Sometimes, the devices exceed the performance and coverage of specific networks, producing data processing and exchange issues.

6) Consistency and Data Flow Management

- While developing a application for smart devices, consistency and data flow management is crucial. For instance, certain devices in a smart home collect the data and then represent it on the user's devices (phone, tablet, smartwatch, *etc.*).
- A stable and uniform database connection assures that up-to-date data is displayed across all devices, all the time.
- Most of the data that smart devices operate with these days is unorganized, so it might create difficulty to store it in a relevant SQL format.
- The data must also ensure an ideal data flow through multiple layers of the smart system's architecture.

UNIT VI End Sem.

7) **Optimal Performance**

- Besides developing a successful app and providing a superior customer experience, most app developers face a common challenge for ensuring a world-class app performance.
- The challenge involves the functioning of an application without crashes or bugs and at the same time utilizes as little space in the device without altering battery life.
- Firstly, you should confirm the app should work in the right way. If an app offers optimal performance, then only users will use it.

8) **App Security**

- Security concerns can be annoying for mobile developers. For instance, malware problems may arise, and software/hardware fragmentation only adds to the distress' list.
- There is a lot of effort required to address such app security problems, which consumes a lot of time and money. Unless proper security standards are complied with, security lapse can commence to information misuse and manipulation, poor user experience, and scarce app adoption.
- In short, challenges should not hinder the performance of your app. Understand each challenge and take appropriate actions to overcome the issue and develop a smooth operating app.
- IoT app development services providers closely work on overcoming challenges and bypass the issues entirely with their expertise and experience.
- When you need to develop an app for smart devices, look for the app development service provider who has skilled resources and worked on similar projects.

6.7 FUTURE OF HCI

- Although the computer has been an indispensable part of our lives for decades, the mouse/keyboard method of communicating with our devices is the one most of us still use.
- Only recently have we been able to move on to more sophisticated methods of interacting with our devices:
  - Smartphones have introduced new **touchscreen techniques,** now available now on large screen devices, as well.
  - Many of us use "Android Voice" or "Siri," artificial intelligence **voice recognition assistants** that answer questions, give us direction and much more.
  - This technology has advanced to a level where your smartphone is now able to turn your voice messages into texts one with satisfying accuracy. Voice commands also work with a wide range of other applications, including many in our vehicles.
- Unfortunately, there's still much room for improvement with this technology, since our devices too often misunderstand what we are saying and then either ignore us or give us incorrect information.

(IF7) **Fig. 6.7.1: Future of HCI**

Emerging and just on the horizon

- Another promising new interface technology is Gestural Interface Technology (**g-speak**). This system uses cameras and Radiofrequency Identification Tags **(RFID)** to track your movements with special reflective gloves.
- This gives you the ability to control images on a screen without touching or coming near the device at all. RFID is the foundation for the exciting, widely praised new versions of virtual reality gaming.

- More practical uses are on the horizon, e.g., helping you to prepare a meal when your Internet of Things interface (IoT) recognizes that your refrigerator contains the ingredients of five of your favorite meals.
- When you get home, you'll find that your oven has been preheated and a printout of several new recipes that are consistent with your preferences as new options. On the downside, RFID tags much like your smartphones may be used to track people moving through any environment.
- The most interesting long-term computer interface is **direct brain-device communication**. This will require accurately mapping the signals of our brain neurons in order for us to communicate with and control external devices simply by thinking a command. In reality, this will be a formidably complicated thing to accomplish. First, because we don't yet know how to detect brain activity without the invasive insertion of electrodes. Second, because the human brain is so incredibly complex and easily distracted.
- Such an interface would need to be able to distinguish between clear mental commands and all the background 'noise' that the brain makes. Nonetheless, we are witnessing amazing progress in this technology on a daily basis. So, it shouldn't come as a surprise when in the future we send the keyboard and mouse combination to a more-than-deserved retirement.
- As a UI designer, the key here will be staying up to date with **new voice-guided apps and technologies** as they enter the market.

Chapter Ends...

❑❑❑