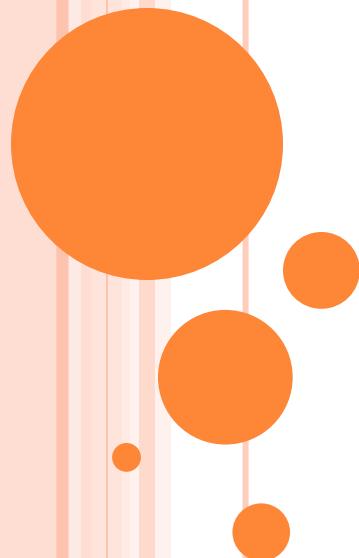


# HUMAN COMPUTER INTERACTION - AN OVERVIEW



**Professor Ram Mohana Reddy Guddeti**  
**Information Technology Department**  
**NITK Surathkal, Mangalore, India**

# OVERVIEW OF HCI

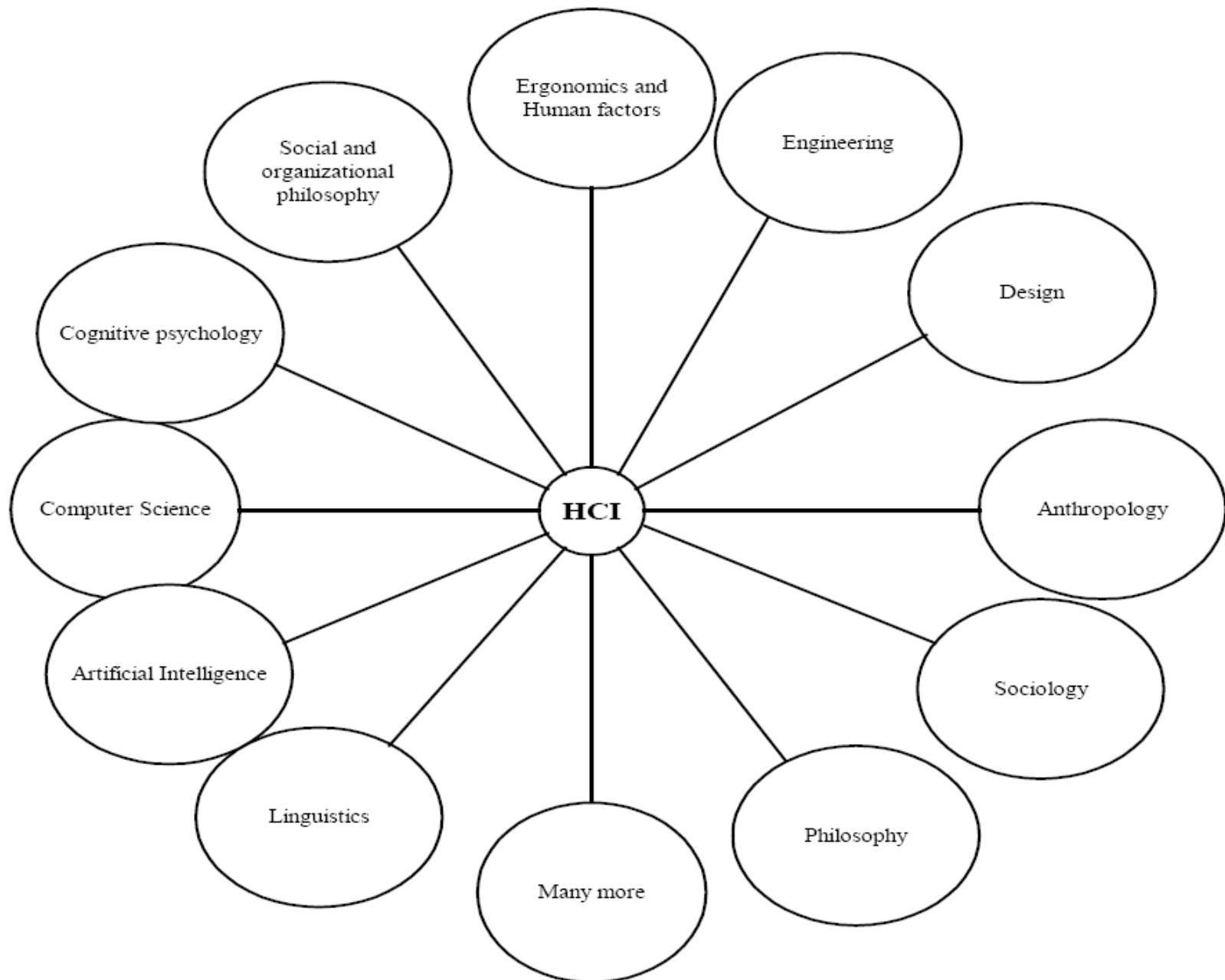
- HCI stands for **Human-Computer Interaction**, earlier it was known as the **man-machine studies**
- Sometimes Computer-Human Interaction (CHI) is used
- HCI Definition as per ACM SIGCHI: Human Computer Interaction (HCI) is a discipline deals with the Design, Implementation & Evaluation of Interactive Computing Systems for Effective Use by Humans and with the study of major phenomenon surrounding them



# INTERDISCIPLINARY NATURE OF HCI

- HCI is an interdisciplinary and multidisciplinary area that deals with several disciplines, each with different emphasis
  - ❖ Computer Science & Engineering and Information Technology (Application Design and Engineering of Human Interfaces)
  - ❖ Psychology (Application of Theories of Cognitive Processes and the Empirical Analysis of User Behaviour)
  - ❖ Sociology and Anthropology (Interaction between Technology, Work and Organization)
  - ❖ Industrial Design (Interactive products such as Airport's Self-Service Check-In Kiosk, Automated Teller Machine, Bank Passbook Printing Kiosk, Cell Phone (Smartphone), Microwave Oven, Smart TV, Vending Machine, Washing Machine, etc.)





# COMPUTER (OXFORD ENGLISH DICTIONARY)

- Computer – It is an electronic device (system) which is capable of receiving information (data) in a particular form and of performing a sequence of operations in accordance with a predetermined but variable set of procedural instructions (program) to produce a result in the form of information or signals
- Computing – The use or operation of computers



# EXAMPLES OF COMPUTING SYSTEMS

- Desktop, Laptop and Tablet PCs
- Smartphones
- Wearable Digital Pedometer
- Microwave Oven
- Touch Operated Smart TV (Internet TV)
- Automatic Ticket Vending Machine
- Automatic Coffee/Tea/Beverages Vending Machine
- Automatic Snacks Vending Machine
- Airport's Self-Service Check-In Kiosk
- Automated Teller Machine
- Bank Passbook Printing Kiosk
- Washing Machine etc.



# USER-CENTRIC DESIGN OF COMPUTING SYSTEMS

- It should be noted that the computing systems mentioned in the previous slide are not some specialized equipment to be used by the experts in a specialized environment; but these are meant for a very large group of people, majority of these people do not (and need not) have technical domain knowledge of these electronic gadgets
- The questions are: “How to Build the Product (Electronic Computing Systems) for a Very Large Group of People”
- “How Can We Design these Products such that the Users of the Product find those Easy to Use?”



# USER-CENTRIC DESIGN: DEFINITION

- User-Centric Design (UCD) - It is defined as the “Process to Design an Electronic Product, which satisfies the definition of a Computing System, in which the Users’ Needs and Expectations are taken care of, while considering the Users’ Characteristics and their Background”
- Three Key Elements of User-Centric Design are:
  - (i) Design of Interactive Electronic Products that Satisfies the Definition of a Computing System
  - (ii) The Products are to be used by the Users (need not be Technology Experts i.e. Non-experts)
  - (iii) The Design Process takes care of the users’ needs and expectations, by considering the users’ characteristics and their background, so as to make the product “Easy to Use”



# USER-CENTRIC COMPUTING

- User-Centric Computing refers to the Computation of User Interface (UI) Layout and Transitions, by taking the Human Behaviour and Cognition into account
- Similarly the emphasis should be on the design rather than the computation; however computation gives us much more power in terms of saving the design time and effort as well as creating an “Adaptive Interface”
- User-Centric Product: Four Key Aspects of the Design
  - Design Elements that are Acceptable to the Users,
  - Design Layouts that meet the Users’ Expectations,
  - Helps the User to Perceive the “System State”,
  - Design Interaction that fulfils the Users’ needs, while considering their desired ‘System States’ into account



# **HCI: Interactive System Design**

## **(Usability Engineering Perspective)**

Professor Ram Mohana Reddy Gudde  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Usability Engineering

UE

## Overview of Usability Engineering (UE)

- The need for usability
- What do usability and UE mean?
- What happens without UE?
- What is the UE Lifecycle
- User-Centered Design (UCD)  
Methodology



# Usability Engineering

- Jacob Nielsen: Usability Engineering (1993) Well known book.
- Kristine Faulkner (2000): defines it as follows

“UE is an approach to the development of software and systems which involves user participation from the outset and guarantees the usefulness of the product through the effective use of a usability specification and metrics”

- UE refers to the USABILITY FUNCTION aspects of the entire **process** of conceptualizing, executing & testing products (**both hardware and software**), from requirements gathering stage to installation / marketing & testing of their use.

# Definition of Usability

- **Usability** is the effectiveness, efficiency and satisfaction with which users achieve specific goals / objectives in particular environments; where
  - **Effectiveness** is the accuracy and completeness with which specified users can achieve specified goals in particular environments;
  - **Efficiency** is the resources expended in relation to the accuracy and completeness of goals achieved; and
  - **Satisfaction** is the comfort (experience) and acceptability of the work system to its users and other people affected by its use.

## User's Definition of Usability

USABILITY : The ability of a User to Use the product/ system / environment as desired  
Usability Engineering: The ‘affordance’ offered by a product that makes it useable.

**Usability does not happen by it self. It has to be “engineered” into the product.**

## **Usability is related to Human Performance**

Capabilities  
Constraints / Limits  
Consequences

### Intuitiveness

Maximum success for first-time users, with minimum training, explanation or thought

### Efficiency

Maximum success for long-term users, with minimum time, mental load, physical effort

---

Usability is conceptualized into the product by **DESIGN**

Usability has three major components in Design

### Appearance

Visual Quality

### Technology

Build Quality

**DESIGN**

Interaction  
Use Quality

# Some Standard Definitions .....

- ‘**Usability**’ is the measure of the *quality* of a User’s experience when interacting with a product or system
- ‘**Usability Engineering**’ is the processes of deriving, specifying, measuring, constructing and evaluating usability features into products and systems.
- **Usability Study** is the systematic analysis based on heuristics and/or experimental evaluation of the interaction between people and the products including the environment of use.  
**Psychology / Cognitive Science / Behavioral Science**
- **Usability Testing** is the scientific verification of the specified usability parameters with respect to the Users needs, capabilities, expectations, safety & satisfaction.

Usability as applied to **Product Design**

Usability as applied to **Human Computer Interaction**

Usability as applied to **Human Environment Interaction**

Usability as applied to **Systems (including Engineering Systems)**

# The UE Lifecycle UCD Methods (ISO 13407)

SYSTEM LIFE CYCLE						
FEASIBILITY		REQUIREMENTS		DESIGN	IMPLEMENT	RELEASE
USER REQs	CONTEXT OF USE	FUNCTIONAL	TECHNICAL	PROTOTYPE	USEABILITY TESTING	FEEDBACK

## Design Stages

Task	Information Produced
Knowing the User	User Characteristics, User Background
Knowing the Task	User's Current Task, Task Analysis
User Requirements	User Requirements Specification
Setting Usability Goals	Usability Specification
Design Process	Design Specification
HCI Guidelines & Heuristic Analysis	Feedback for Design Iteration
Prototyping	Prototype for User Testing
Evaluation with Users	Feedback for Freezing Design
Redesign and Evaluate with Users	Finished Product
Evaluate with Users and Report	Feedback on Product for Future Systems

# The Goals of Usability Engineering

## 5 Es

- Effective to use - Functional
- Efficient to use - Efficient
- Error free in use - Safe
- Easy to use - Friendly
- Enjoyable in use - Pleasurable Experience

Achieves 5 times Enhancement in Engineering value.

# Home Work

## Usability Evaluation

Conduct a quick Usability evaluation of your mobile phone & Compare it with the evaluation of your friends phone.

Rating out of 10

Effective to use - Functional

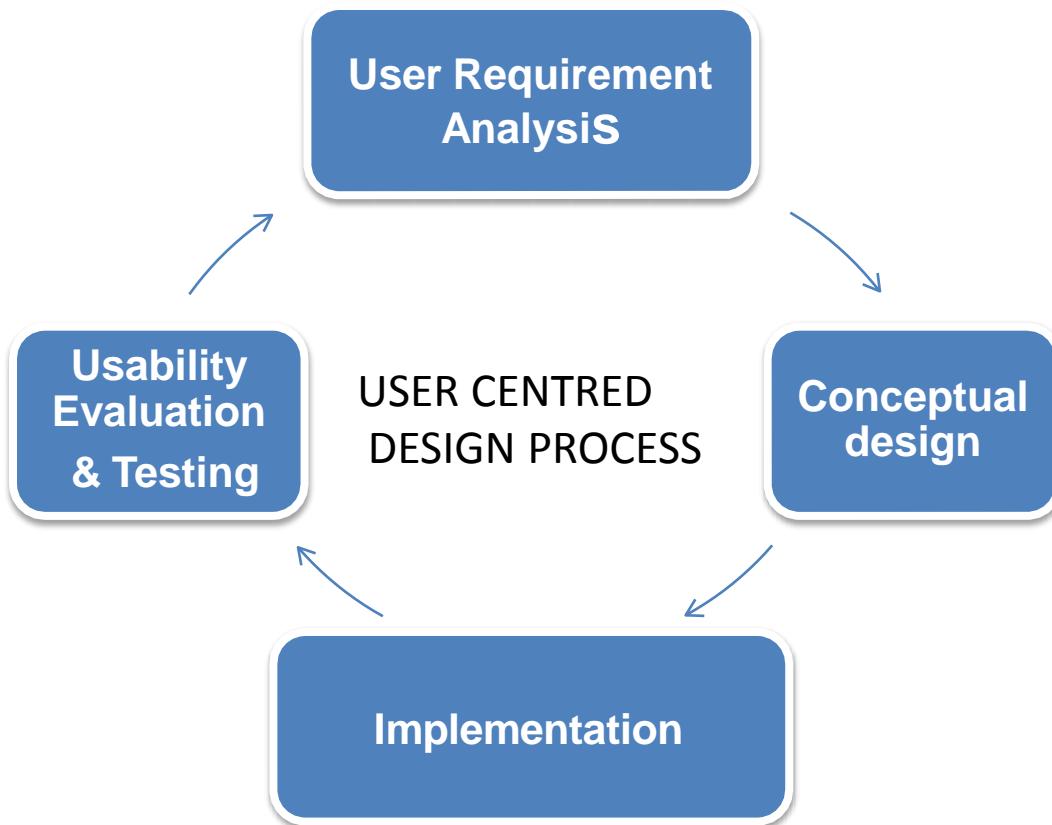

Efficient to use - Efficient


Error free in use - Safe


Easy to use - Friendly


Enjoyable in use - Pleasurable

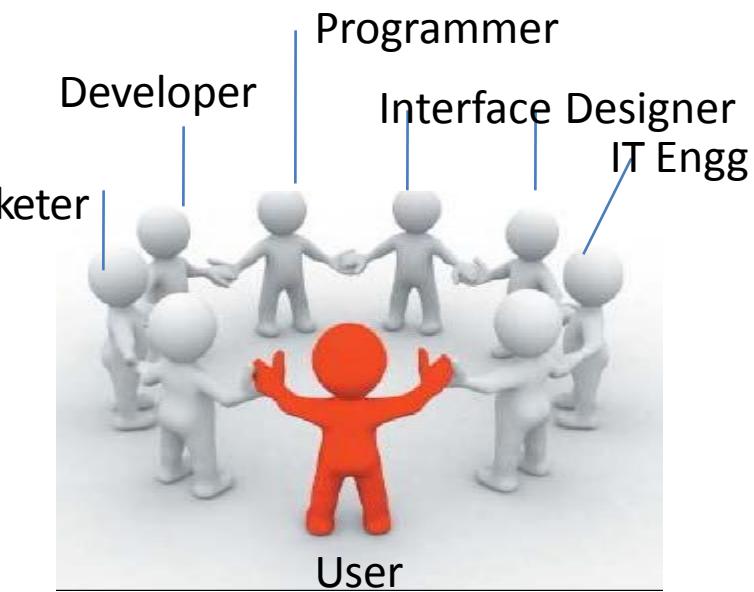

Total :



- UE is based on a **User-Centred Design (UCD)** approach to analysis and design. It concentrates on those aspects of products & services that have a bearing on their effective, efficient & pleasurable USE by humans.

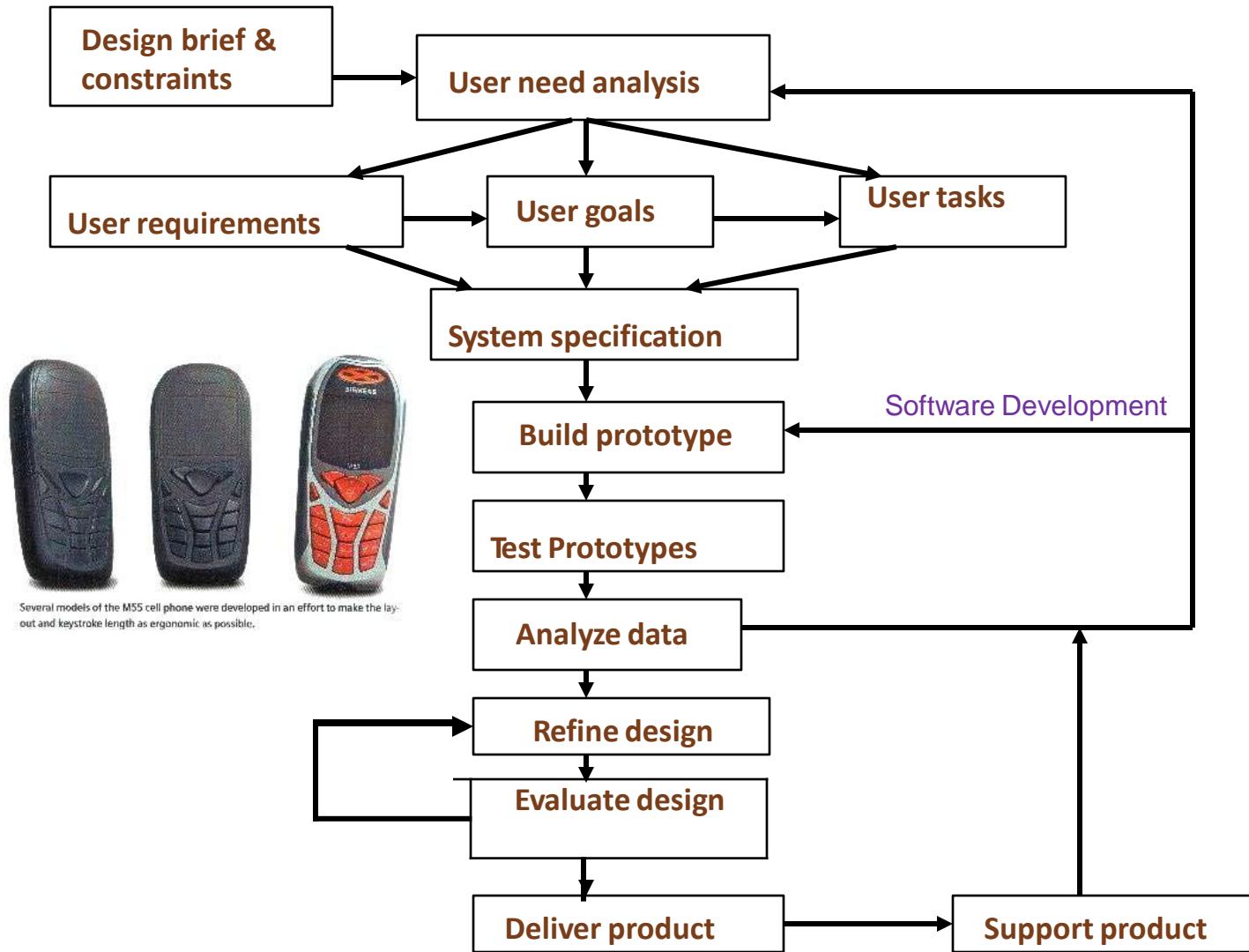
**ISO 13407, 1999**

*"Human-centered design is an approach to interactive system development that focuses specifically on making systems usable.  
It is a multi-disciplinary activity."*



# The UCD Methodology.

## User Centered Design Processes: UCD



# Definition of UE & Other Related Fields

**HCI:** Human Computer Interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them. *ACM - Association for Computing Machinery.*

---

**Human Factors & Ergonomics:** Stress on human physical issues (physiology) and on optimizing work processes

---

**User Interface Design:** Focuses on interface layer assuming all deeper functions are fixed.

---

**HCD- Human Centered Design:** Approaches to software engineering with user focus at all stages of software design

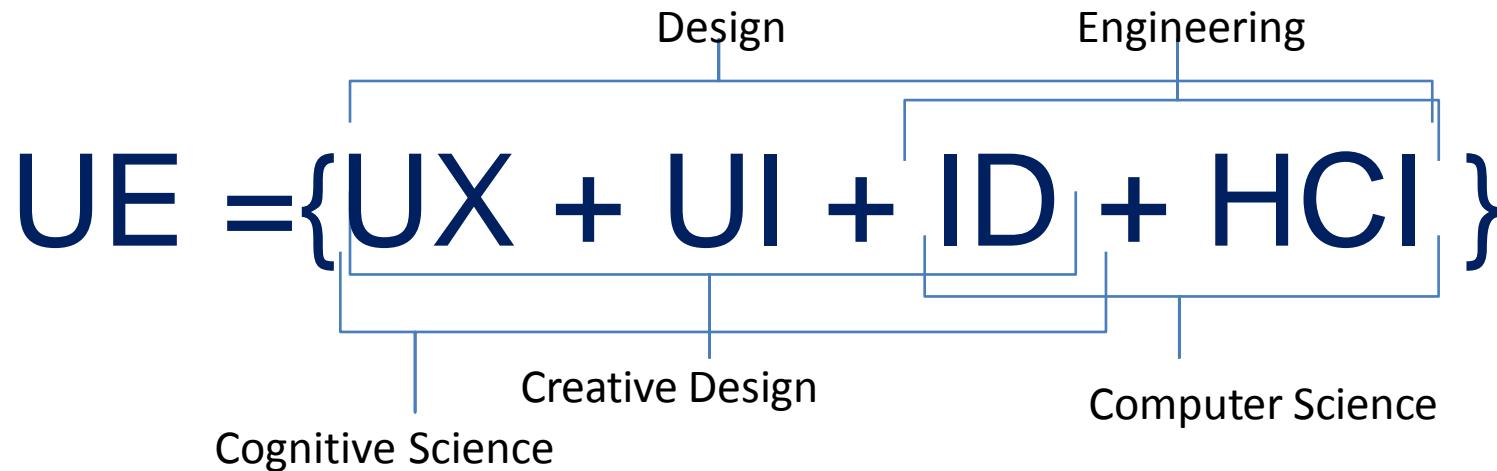
---

**ID – Interaction Design:** Wider scope in terms of devices beyond computers. More emphasis on cognitive & experiential factors.

---

**UE- Usability Engineering:** Focuses on design & implementation processes. It is essentially research & design based activity .  
**There are overlaps in the above fields. Each is independent. UE has all of them.**

# Relationship between UE & Human Computer Interaction; Interaction Design; Experience Design; GUI Design



UX = User Experience

UI = User Interface

ID = Interaction design

HCI = Human Computer Interaction

UE = Usability Engineering

Please note : UE is written as 'Usability' as well as 'Use-ability'.

# UE vs Software Engineering

- Key difference (Karat and Dayton, 1995):
  - “In most cases of the design and development of commercial software, usability is not dealt with at the same level as other aspects of SE, (e.g.
    - Clear **usability objectives** are not set; and
    - Resources for appropriate activities are not given priority by project management).
- To produce *usable* interactive products requires (Mayhew, 1999):
  - **UI design principles** and guidelines.
  - **Structured methods** for achieving usability.

# Usability Testing & UE – the Difference

- Usability engineering
  - Methodical approach to producing user interface + Experience + function + aesthetics
  - A way to deliver a product that works
- Usability Testing
  - Part of process of UE
  - Real users performing real tasks

# Usability Testing

- **Analytical Evaluation:**

- By simulating *how* the user's activity will be performed.
- Heuristic evaluation measures design against a list of usability factors.

- **Empirical Evaluation:**

- By building and testing a *prototype*.
- Formal usability testing tests a component of the design under controlled conditions - actual users; thus needs a usability laboratory.



# Cost-justifying Usability

**\$1 spent on usability = \$10 saved (Nielsen, 1993).**

Rs. 50 spent saves Rs 500 worth of trouble shooting due to poor design

## Ignoring UE .....

### Frustrated users Low productivity

*Poor user interface design is the cause*

High costs Support/Help desk costs

*Entering data incorrectly*

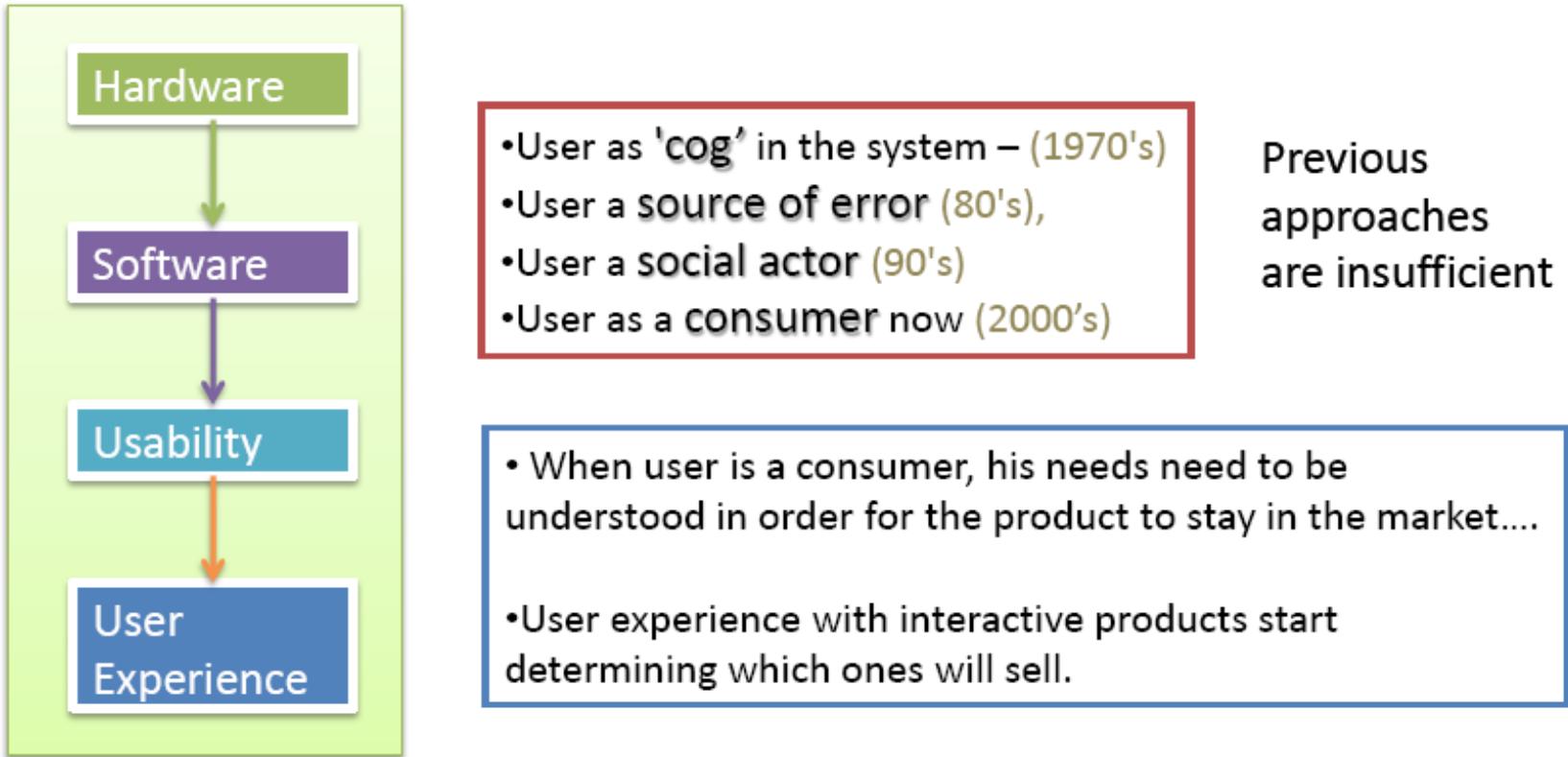
*Deleting data*

*Loss of market share, good will  
Competitors rush in.*

Mobile / Tablet / Device companies now are heavily investing in UE as the value adder as well as product differentiator.

They do not consider ‘cost’ as a constraining factor w.r.t. UE.

# Evolution of HCI and understanding of Users



The UE processes is based on **four fundamental axioms of Design**

- **User is the only constant entity of an artificially created system.**
- **User is the starting point of all design**
- **User is the final datum of reference for all design decisions**
- **User is the measure of all things.**

Nielsen (1993) identified 5 attributes that contribute to usability:

- **Learnability.** The user should be able to promptly start performing their tasks with the system.
- **Efficiency.** Once the user has learned the system, a high level of productivity should be possible.
- **Memorability.** The casual user should be able to return to the system after not having used it for some time, without having to relearn everything.
- **Errors.** Users should not make many errors using the system, and if they do, they should be able to easily recover from them. Catastrophic errors should not occur.
- **Satisfaction.** Users should like using the system and should be subjectively satisfied when using it. The system should be pleasant to use.

# Digging Deeper into Usability    What makes a product **usable** ?

Is it all subjective ?..... Can we measure Usability?

Stanton & Barber 1996 proposed measuring the following :

**Learnability   Effectiveness   Attitude   Flexibility   Compatibility**

---

**Learnability:** A product (system) should allow users to reach the acceptable levels of competency and the performance within a specified time.

**Learnability  
Consistency  
Familiarity  
Standards**

- Help the users to master the system
- Let the users have to learn only once
- Build on users' prior knowledge
- Respect established cultural and application specific conventions .

**Self-descriptiveness   Help**

- Make objects and controls intuitive
- Provide easy access to 'help' resource

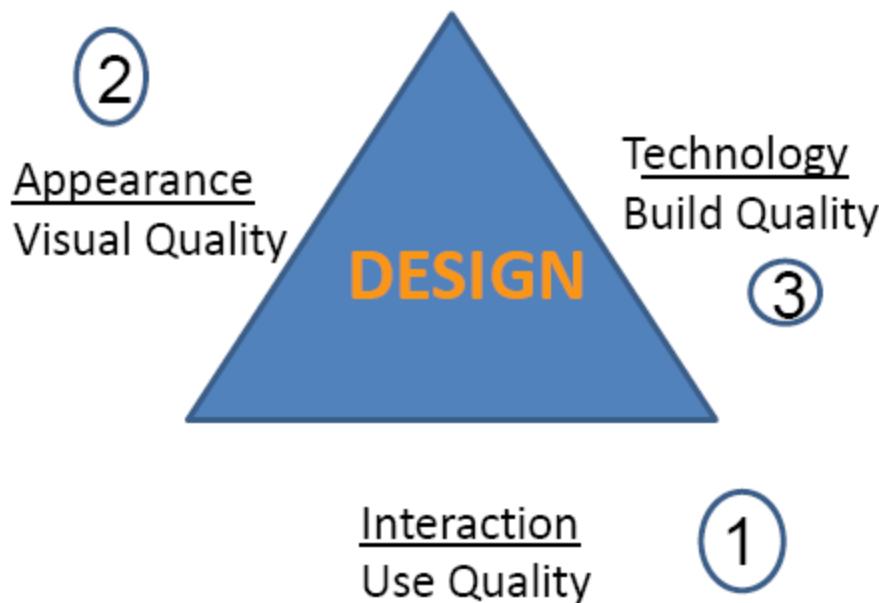


**'Intuitive' User Interfaces do not require investing resources in 'Learning'. Such interfaces follow the User's Mental Model of Interaction**

# Designing User Interface for Mobiles / Tablets

UI

1 , 2, 3.



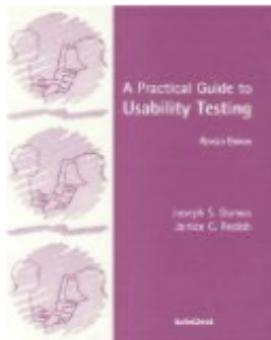
**Technological feasibility is different from Usability.**

**Engineering / Software  
should not dictate usability**

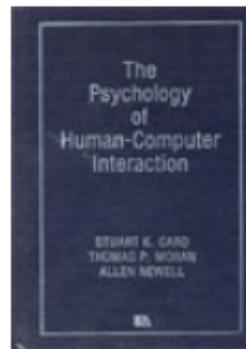
What is involved in GUI design ?

- Designing for ease of use
  - Usability : Semantics , Dialogue, Communication
  - Mental Models**
- Designing for attractiveness
  - Aesthetics
  - User Experiences ....
- Designing for contextual awareness
  - Culture , Behavior

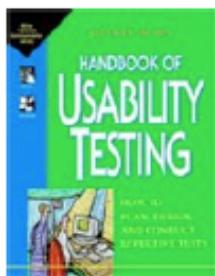
# Some Usability Books



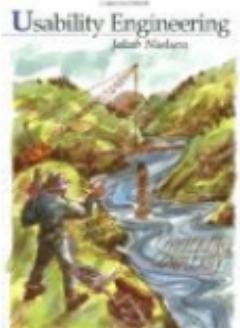
. A Practical Guide to Usability Testing by Joe Dumas & Ginny Redish (1993)



The Psychology of Human Computer Interaction  
Stuart Card, Thomas Moran & Allen Newell (1983)



2. Handbook of Usability Testing by Jeffrey Rubin (1994)



3. Usability Engineering by Jakob Nielsen (1993) Morgan Kaufman , Academic Press London.

# **Interactive System Design**

## **(HCI and Software Engineering)**

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Learning Objective

- In the previous lecture, we learned basics of “usability”
  - While designing an interactive computing system, what and why we should do take care of the usability?
- In this lecture, we shall learn about the answer to the above question

# Learning Objective

- In particular, we shall learn / discuss the following
  - The key differences between a software design and an interactive computing system design
  - User-Centered and Participatory Design
  - The interactive system design life cycle

# The Central Idea

- Suppose we are asked to design a DBMS then what are our design objectives?
  - Efficient storage of large databases (storage)
  - Efficient way to retrieve the results of a query from the database (retrieval)
  - Allow the user to access the database (interaction)

# The Central Idea

- Note that this is a scenario where the user interacts with the system (database)
- However, the user is a “computer expert”, who has “technical knowledge”
  - Through some query language, the user can access, manipulate and update the database

# The Central Idea

- Let us consider a tourist information system
- In the background, it is nothing but a database of various tourist-related information
- However, its users may or may not be “computer experts”
  - They do not care about what goes inside the database
  - They just want to “get” the information “easily”

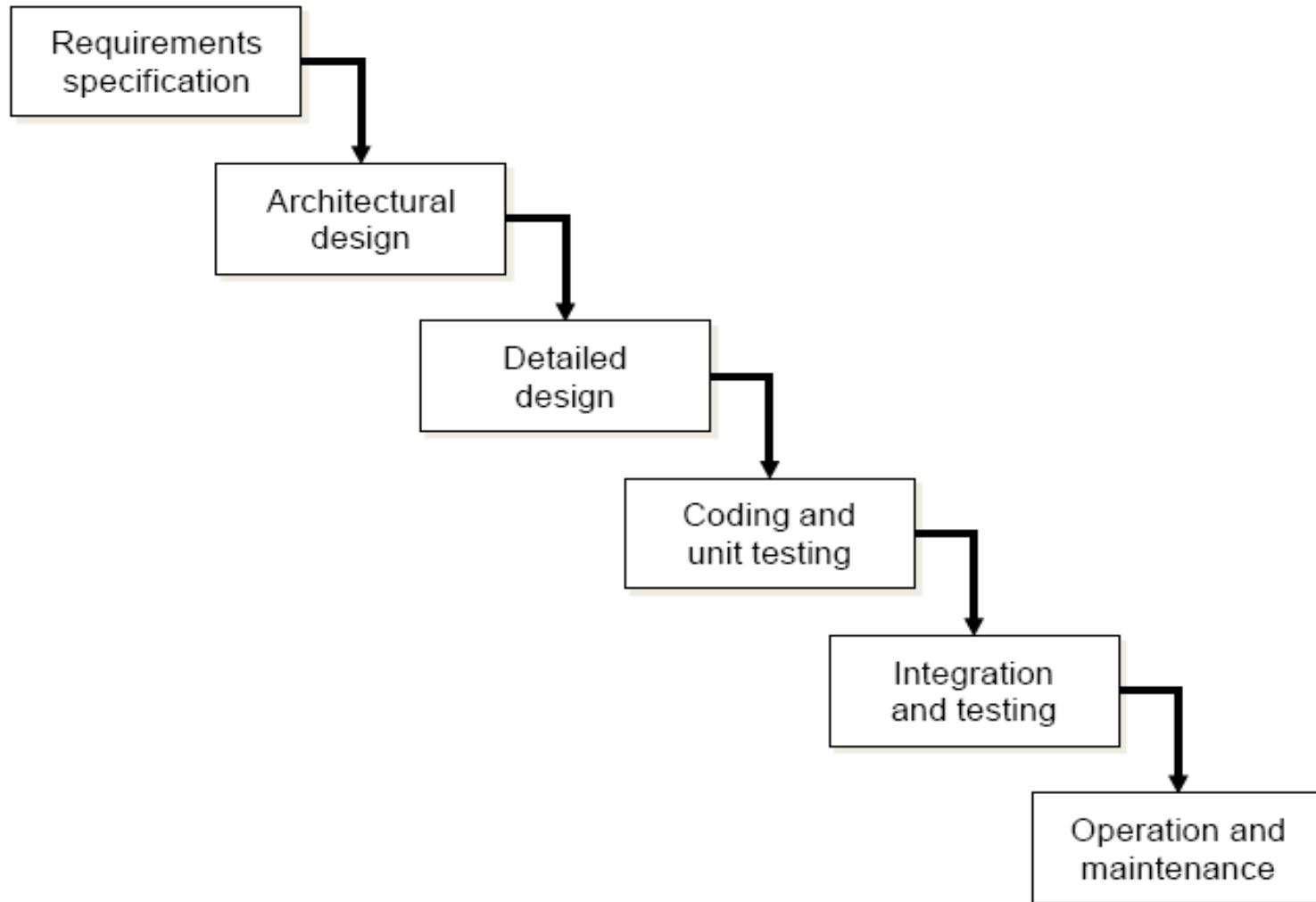
# The Central Idea

- The term “easily” is very significant
  - It means, we need to have an effective/efficient interface and interaction mechanism that do not require any specialized knowledge
- That is, we need a “usable” system
- **Design goal of an interactive system:  
increase usability**

# What Happens in Software Engineering

- The waterfall model: the simplest and typical way to visualize the software design
- Design process composed of a series of sub-stages
  - Each sub-stage follows the previous stage and precedes the next stage (looks like a waterfall)

# The Waterfall Model



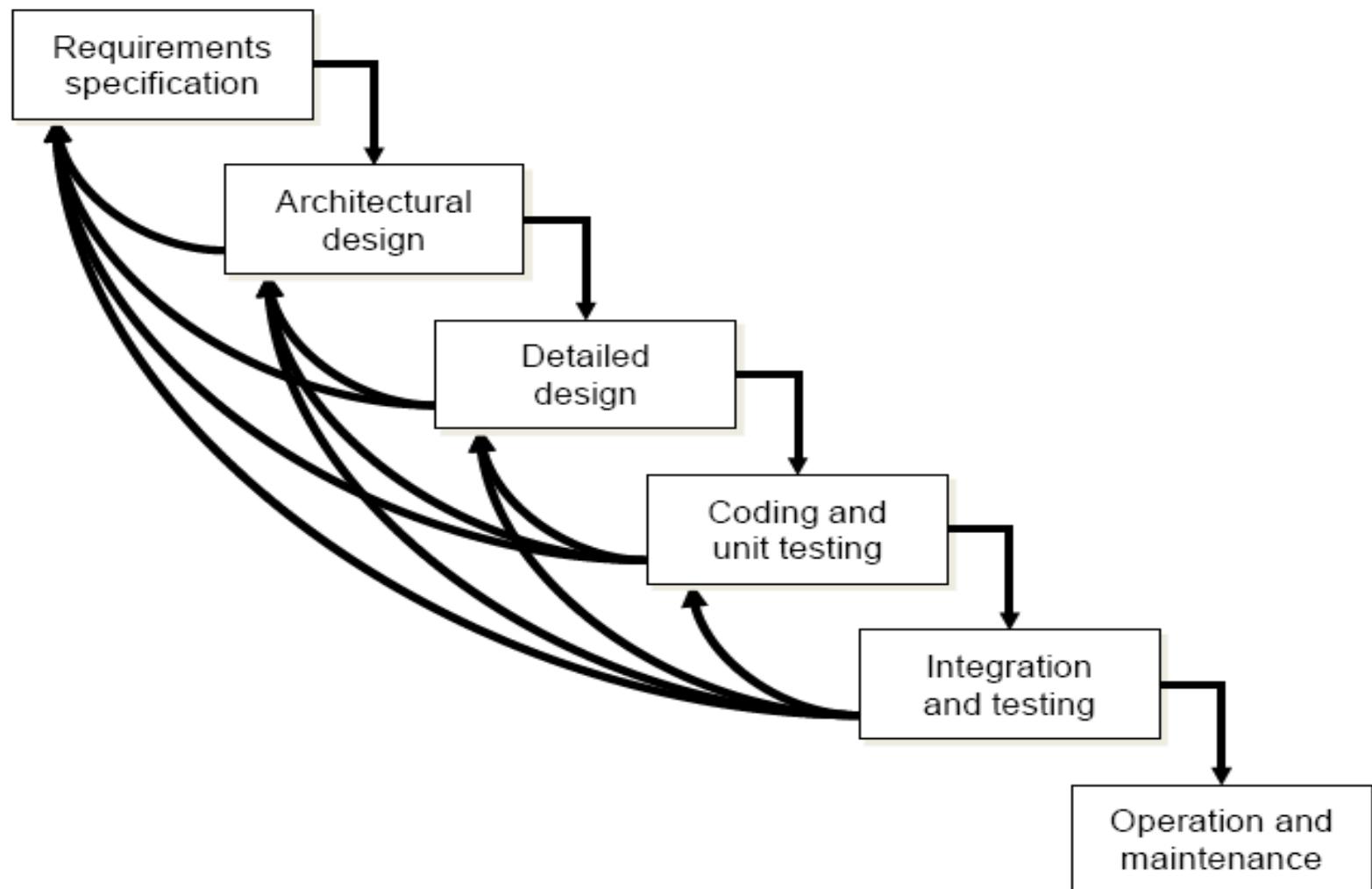
# What Happens in Software Engineering

- Please note that the Uni-directional flow (that's how the real waterfalls work anyway!!)
- In other words,
  - Each stage depends on the previous stages but not vice-versa

# Interactive System Design

- The uni-directional flow is not appropriate for an interactive system design
- In other words,
  - Each stage depends on the previous stages. It may also depend on the next stages (feedback).
- It is no longer the (linear) waterfall model

# Interactive System Design



# Why This Difference

- We will try to design “for” the user (User-Centred Design)
  - Not for a programmer’s convenience or expert’s use
- What should we consider
  - Abilities and needs of the users
  - Their usage context
  - Their work setting
- In other words, we need to “Know the User”

# Need: To Know The User

- A never ending process because there is so much to know and because the users keep changing
- An interactive system designer should consider the human factors that characterize the users

# The Human Factors (Users' Characteristics)

- Perception: Our ability to perceive our surroundings
  - Can be visual, auditory or haptic (touch)
- Cognition: The way we process the perceived information in our “mind” and accordingly take the decisions
- Motor Action: This is the mechanism through which we interact with the surroundings
  - Examples: Hand movement, Eyeball movement, Speech etc.

# Need: To Know The User

- The following factors (user's characteristics) vary with
  - Age, gender, physical and cognitive abilities, personality
  - Education, cultural or ethnic background
  - Training, motivation, goals
- An interactive system designer should recognize its diversity

# Need: To Recognize Diversity

- Systems used by several communities of users
  - No single design can satisfy all users and situations
- Designers face the real challenge to cater to the needs of each community
  - Designers must characterize users and situations as precisely and completely as possible

# A Generic User Characterization

- Novice or First Time Users
  - They Know nothing about the task or interface concepts
  - They Often anxious about the computer and its functionality
- Knowledgeable or Intermediate Users
  - They Have stable task concepts and broad interface concepts
- Expert Users
  - They are Thoroughly familiar with the task and interface concepts
  - They Want to get their job done quickly

# So, Why The Difference?

- Designers must know the user
  - This knowledge can not be captured at once
- Design involves in acquiring the new knowledge and using it to refine design in continuous cycle (till some “acceptable” design is found)
  - This is the reason for so many “feedbacks” in the waterfall model

# User-Centered Design (UCD)

- The design process, where the designer collects the feedback about the design from users and use this towards refining the design, is known as “user-centered design” or UCD
- UCD is based on understanding the domain of work or play in which people are engaged and in which they interact with the computers

# User Centered Design (UCD)

- Assumptions
  - Result of a good design is a *satisfied user*
  - Process of design is a *collaboration between the designers and the users.*
  - Design *evolves and adapts* to the users' changing concerns, and the process produces a specification as an important byproduct
  - The user and designer are in *constant communication* during the entire design process

# UCD Drawbacks

- In UCD, the user involvement is “passive”
  - The designer elicits the feedback from the user (through interviews, informal discussions etc.)
  - Prepares specification on the basis of user response
  - Take feedback on the design and makes refinements

# UCD Drawbacks

- Problems with “passive” involvement of user
  - User intuition about a new interface may not be correct (the feedback is not reliable)
  - The interview process itself may not be formulated properly (the designer asks wrong questions)
  - It is not possible for the designer to identify all possible issues to take the feedback from users, as the designer’s knowledge about the user may not be complete

# Participatory Design

- What is the Solution: Let us make (representative) users as a part of the design team
- Such a design process, where the end users are part of the design team, is known as “participatory design”

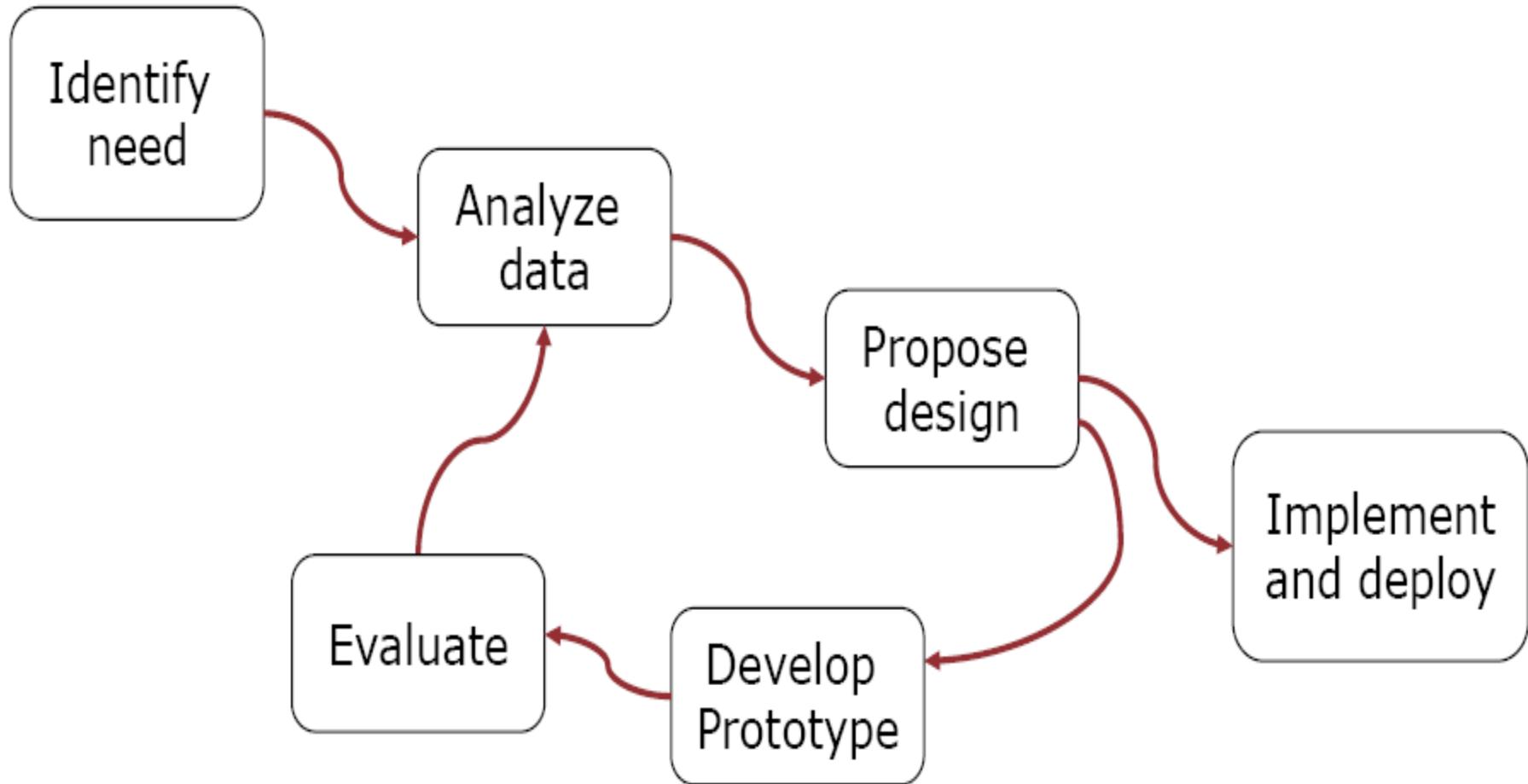
# Participatory Design: Key Points

- Users are first-class (active) members of the design team
  - As opposed to their passive involvement in UCD
- Users are considered subject experts
  - Know all about their work context
- Iterative design process
  - All design stages are subject to revision

# Interactive System Design Life Cycle (ISDLC)

- Key stages
  - Know the user, propose design, evaluate design by the users, refine design
- Iterative design process
  - The above stages are iterated till an acceptable solution (determined from the user feedback) is obtained

# ISDLC: A Consolidated Diagram



# Life Cycle Stage: Identify Needs

- What is wanted – identify users and their needs
- Designers make use of one or more methods to identify the needs and requirements
- Such methods include the following:
  - Interview (structured, semi-structured, unstructured)
  - Contextual inquiry
  - Cultural probes
  - Ethnography
  - User models

# Life Cycle Stage: Analyze Data

- ❖ Analysis of the data collected
- ❖ Two types of analysis are performed
  - Scenario Analysis: Analyze the data collected from the user on one or more usage scenario of the system
  - Task Analysis: Analyze the tasks required to be carried out by the user so as enable them to operate the system
    - System level task analysis: analysis of external tasks required to operate the system
    - Cognitive task analysis: analysis of tasks performed in the mind of the user

# Life Cycle Stage: Propose Design

- Design proposal arrived at from the analysis of collected data
  - Guidelines and principles help in the development of initial design
  - Several sets of guidelines (both general and specific) are there to cater to the needs of a specific interface design context

# Life Cycle Stage: Develop Prototype

- Implement a prototype of the design for collecting the user feedback
- A spectrum of techniques is used in developing prototypes
  - Paper prototype (one extreme)
  - Complete software (other extreme)
  - Lots in between ...

# Life Cycle Stage: Evaluate Design

- Evaluation of the design by the users
- In the initial design phase, evaluation is done on the prototypes: (i) Cost effective and easier to perform, (ii) Suitable for iterative design process where the evaluation is performed many times
- The full system is typically evaluated at the end
  - (i) Full system evaluation is costly in terms of money, manpower, time and effort (ii) Hence, typically done once or a limited number of times

# Life Cycle Stage: Evaluate Design

- Several evaluation methods are available
  - Checklist/guideline based evaluation
    - Heuristic evaluation, cognitive walkthrough
  - Model-based evaluation: employs models (of the system or user or hybrid) for evaluation
    - Hybrid models are essentially models that combines the features of both the system and the user
  - Empirical evaluation – evaluate with real users
    - Involve implementation of the interactive computing system with full functionalities

# HCI: Interactive System Design (GUI Design and Aesthetics)

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

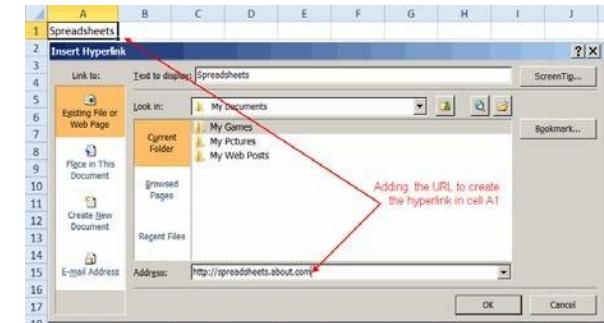
# GUI: Graphic User Interface

The interface through which a user operates a program or an application or a device

Consists of individual or group of ICONS, buttons, scroll bars, menus, widgets, boxes, status lamps, labels, instructions, visuals etc. - arranged on the screen in a pattern that is visually pleasing as well as ergonomically useable.

Very important and critical component in facilitating user interaction with the software & hardware inside the device / product.

GUI determines the Usability Index of the product as a whole. Gives the product an identity, personality & character.



# Requirements of a GUI

## **FUNCTIONAL:**

Useable - Easy to operate; locate what is required & where it is required on the screen; and do what is expected of it – without need for learning or training

## **AESTHETIC:**

Pleasing to the eye; Highest Visual Quality; Identifiable; Distinct; Recognizable, Recallable

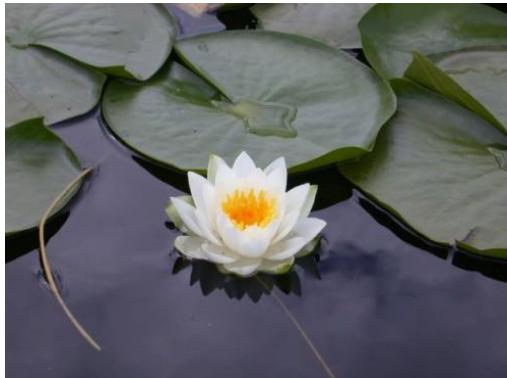
## **COMMUNICABLE:**

Express what it represents; how it is to be operated; Unambiguous; Meaningful; Culturally & Contextually compatible

# In GUI Design Aesthetics is about **Sensory + Empirical + Taste + Judgment**

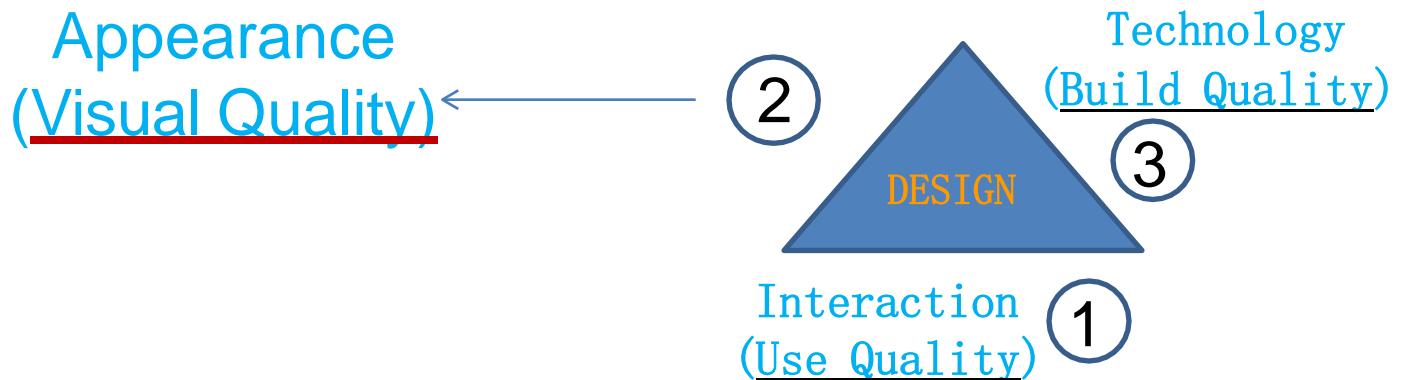
The Philosophical argument of aesthetics shown below is incorporated into Interfaces through Graphic designing

**Simplicity + Infinity + Eternity + Serenity = Beauty**



**Aesthetics is both Art as well as Mathematics.  
It is both rational as well as emotional at the same time.**

# Aesthetics is a medium for User Experience



**Aesthetics (Look & Feel) +  
Communication + Use ability**

**= Total UI Experience**

# Role of Aesthetics – often misunderstood & underestimated

- Aesthetics is not mere beautification.
- It has as much to do with FUNCTION as with beauty
- Aesthetics is not the surface characteristics of a GUI It is not decoration. It is not cosmetic
- A ‘good looking’ GUI needs also **to function**  
**to communicate**  
**to express**  
**to instruct**  
**to perform**



While the judgment of Aesthetics is subjective  
the construction / configuration is not.

There are elements/principles of good aesthetic configuration

## **ELEMENTS**

Line, Shape, Space, Color, Form, Texture, Light

## **PRINCIPLES**

Balance, Emphasis, Rhythm, Unity, Contrast, Movement

# Principles of Design in Visuals

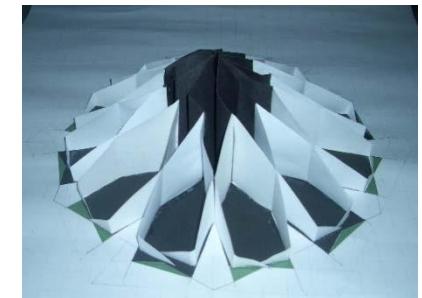
Design is composed of manipulating the physical characteristics of size, shape, texture, proportion, scale, mass and color.

Order & composition is the arrangement and organization of elements in relation to each other.

**Form follows function** is a design approach wherein the form (overall layout / composition/geometric shape) of a GUI is determined by what function it does.

Ex: An arrow has a Form having a sharp angular face at one end  
expressing the function of pointing to a direction.

**Composition**      **Orderly arrangement of elements using the principles of design**



# Principles of Design

**Grammar of the visual language.**

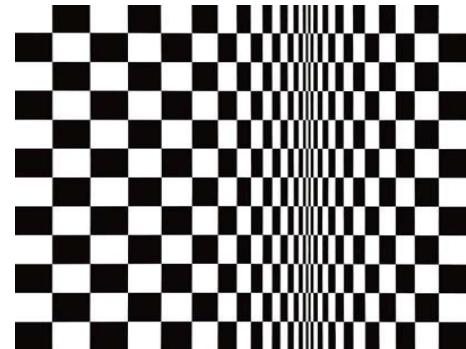
**Rules for composing with the elements**

- The Principles of Design can be thought of as what we do with the elements of design to express and communicate a predetermined message of Usability, Reliability, & Functionality in a harmonious way.
  - **Balance**
  - **Unity**
  - **Proportion**
  - **Harmony**
  - **Direction**
  - **Rhythm**
  - **Symmetry**
  - **Pattern**
  - **Emphasis**
  - **Contrast**
  - **Movement**
- Notice that many of the terms on the right figure are also used in Mathematics.
- Design, therefore has both Aesthetics and Mathematics, underlying it.

# Description of some of the Principles

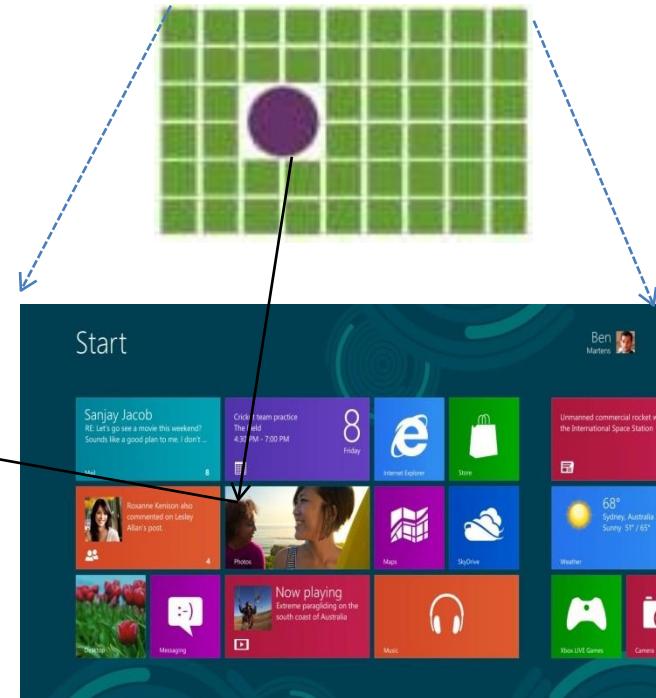
## Rhythm & Movement

- **Movement** is the path the viewers eye takes through the artwork, often to focal areas. Animation is often used.
- **Rhythm** is created when one or more elements of design are used repeatedly to create a feeling of organized movement / direction.



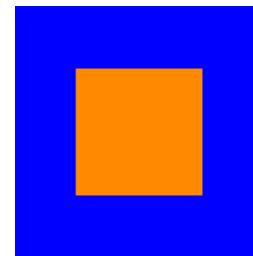
# Emphasis

This is the part of the design that catches the viewer's attention. Usually the Designer will make one area stand out by using the elements of design in a contrasting way. There will be a play with different sizes, colors, textures, shapes etc.

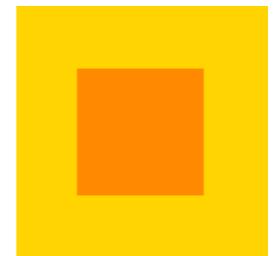


# Contrast

- Differences and Diversities.
- Highlighting similarities



High contrast



Low contrast

# Unity

Unity is an overall “sameness” throughout a screen. How harmoniously all the elements blend together.



Ex: Windows 11 GUI

Ex: Windows 8 GUI

**Balance** Visual balance. Are the various elements visually balanced in terms of their Size, shape, weight, and placement. Can the rhythmic order be visually discernable ?

# Proportion

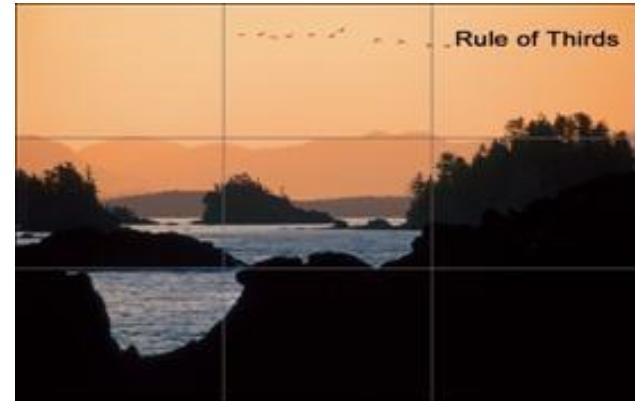
Size relationships found within an object or design. Also a comparison in terms of ratio of size, shape, etc with neighboring elements.

Example see proportions of various buttons within Windows 8 screen



# Proportion & Rule of Thirds Division of a Screen

Proportion refers to the size relationship of visual elements to each other and to the whole picture. One of the reasons proportion is often considered important in composition is that viewers respond to it emotionally.



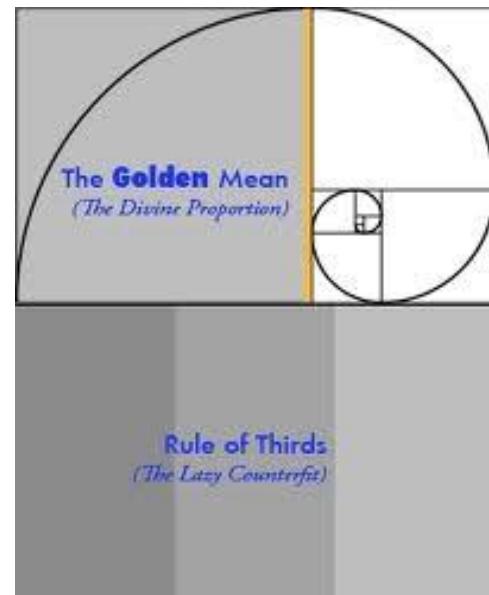
Aspect Ratios



Horizontal



Vertical



# COLOUR

- Colour is a vast subject of both Physics and Fine Arts.
- Graphic Designers use metrics to specify colours.

**Hue:** refers to the names of the primary colours. (red, green and blue).

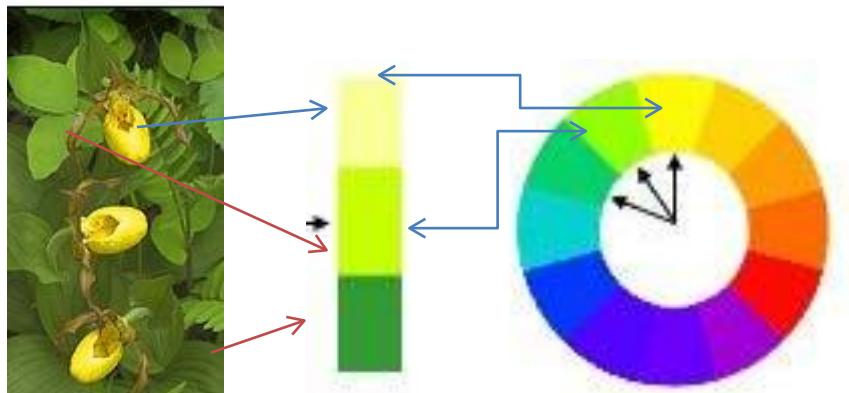
**Value:** lightness and darkness of the hue –

**Shade:** amount of white or black added.

**Intensity:** the purity or saturation of the colour

**Monochromatic :** use of one colour where only the value of the colour changes

**Analogous colours:** colours that are adjacent to each other on the colour wheel, e.g. yellow and green are analogous.



## Limitations of Technology

The Visible spectrum consists of billions of colours, a computer monitor can display millions, a high quality printer is only capable of producing thousands, and older computer systems may be limited to 216 cross-platform colours.

# The Psychology of Colours



**WARM colours include:**  
yellow, red and orange and  
we associate these with  
blood, sun and fire.

# The Psychology of Colours

**COOL** colours include: violet, blue and green because of our association with sky, water.

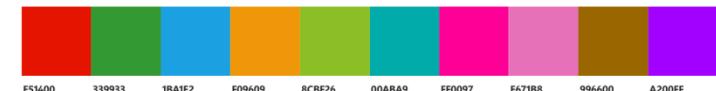
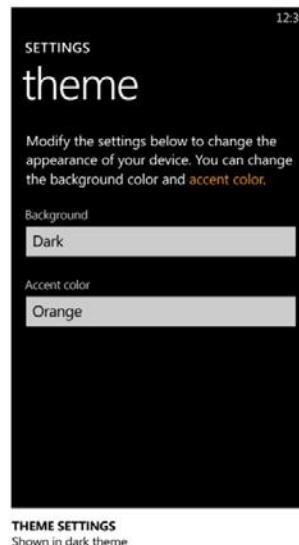
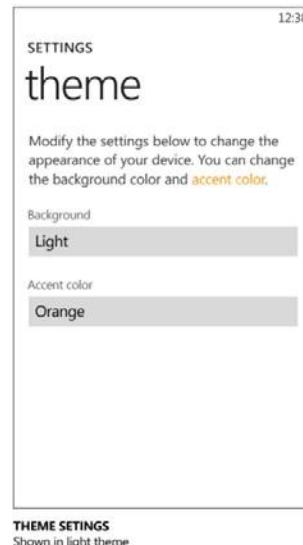


# Colour Theme

Choices of colour given to the User. Simple Pick & Choose does not confuse the user with 100s of colours to chose from.

A set of colours are carefully decided upon by a designer which form a ‘theme’.

All screens have visual elements from the theme.



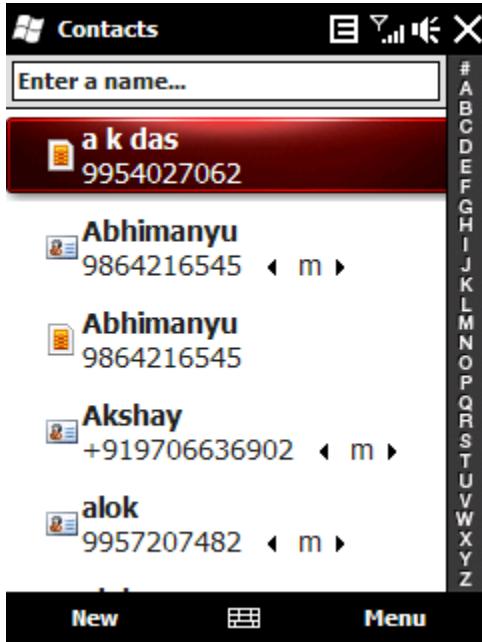
# Graphic Design Principles: Example: Mobile Screen

## The Clustering Principle:

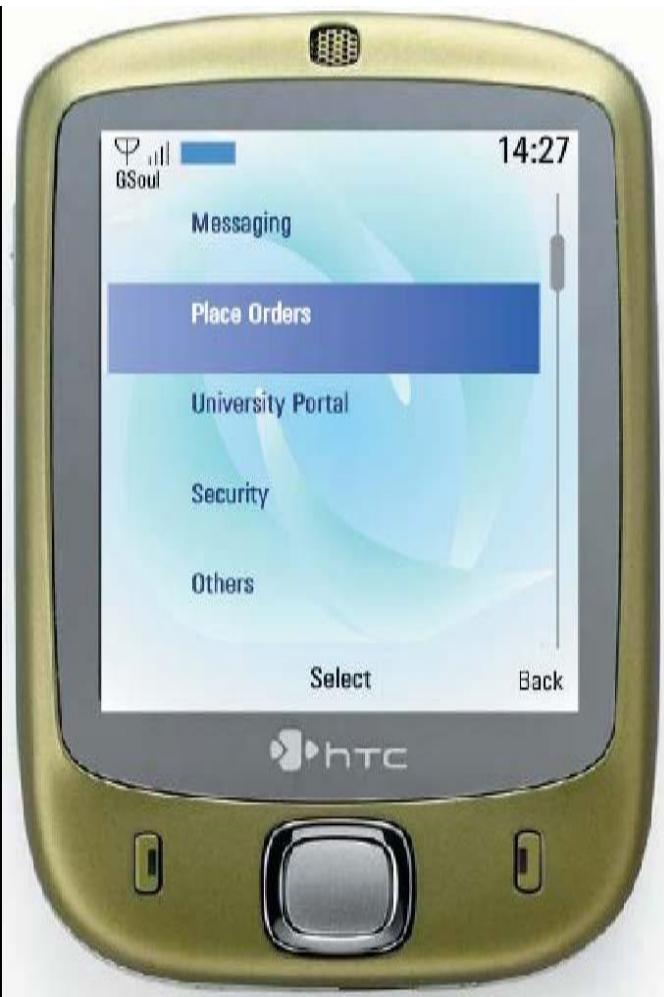
Organizing the screen into visually separate blocks of similar controls, preferably with a title for each block. Modern WIMP (Windows-Icons-Menus- Pointer) systems are a natural expression of the Clustering Principle



Information on a screen which is not categorised into some order (right hand screen in the above figure) can be confusing. GRIDS are therefore used to not only to align & please aesthetically but also categorize UI elements according to functions .



Type size and font, for example:  
the Reduced Clutter Principle  
would suggest that one or two  
type styles are sufficient.



Poor Font readability

# Avoid fancy fonts totally

## Safe Fonts

- Arial, Helvetica, sans-serif
- Courier New, Courier, mono
- Verdana, Arial, Helvetica, sans-serif
- Geneva, Arial, Helvetica, sans-serif

ଜନେରେ ଜନଭୂତ ନାମ ଲୋକେ ଜନନୀ

ଜନମଭୂ ମଶ୍ଚ ସ୍ଵଗାର୍ଦ୍ଧ ପ ଗର୍ୟସୀ

ଜୀନ୍ନ ଜୀନ୍ନମ ମିଶ୍ରସ

ସୁବର୍କାତୁପ କ ଯ

Weight of font matters

BOLD – some times , results in poor smudged readability on mobile screens - even on AMOLED

Regional Fonts still have unresolved problems when used in low resolution & small displays screens.

ଜନେରେ ଜନଭୂତ ନାମରେ ଲୋକେ

ଜନନୀ ଜନମଭୂ ମଶ୍ଚ ସ୍ଵଗାର୍ଦ୍ଧ ପ ଗର୍ୟସୀ

ଜୀନ୍ନ ଜୀନ୍ନମ ମିଶ୍ରସ

ସୁବର୍କାତୁପ କ ଯ

ଜ ନ ଜ ନ୍ତୁ ଭ ଶ୍ଚ ସ୍ଵ ଗର୍ଭାଦ ହେ ଗଯ  
କୁ

ଜନନୀ ଜନମଭୂତିଶୁ ନ

ର୍ଗାତପି ଗରୀଯାଣୀ ଜନନୀ

ଜନମ ତ୍ରୀମ ସ୍ଵଗାର୍ଦ୍ଧ ପ ଗର୍ୟସୀ ଜନନୀ

ଜନମଭୂତିଶୁ ମରାଦିପ ଗରୀଯାଣୀ

# Screen Resolution & Aesthetics

Aspect Ratios



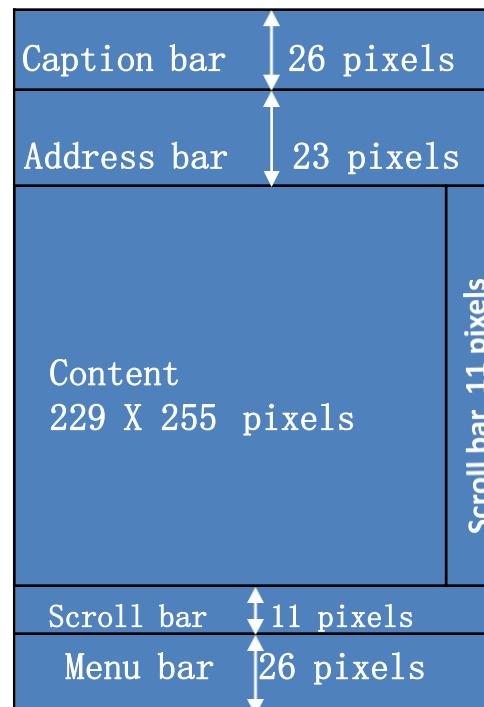
Square



Horizontal



Vertical

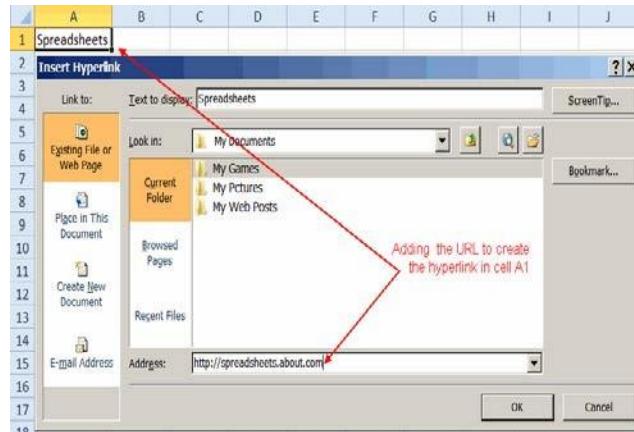


**25~30% area is taken by buttons etc. Hence 229 X 255 is available from 320X240 display.**

# Some Unsatisfactory GUIs



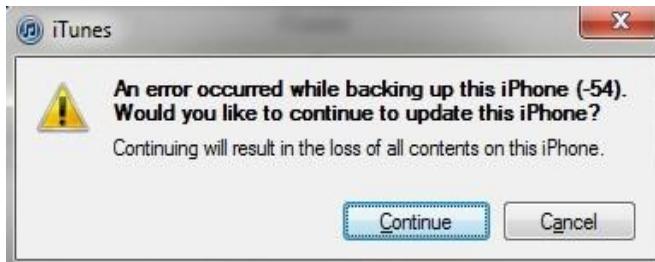
Too many similar elements.  
No colour contrast.  
Monochrome colour scheme  
is visually too heavy. No  
differentiation . No identity.  
Functional confusion. Poor  
communication.  
Difficult to use. Error prone.



No colour contrast.  
Too many data fields  
undistinguishable.  
Use of same colour  
(orange) to for two  
different tasks.



Icon to depict ‘  
security’ has two  
humans. Not  
representational  
& meaningful.



Confused verbal statement label.  
The two buttons offer a dead end. By  
executing the action the user is not  
visually informed as to what to do.

# Case Study 1: Windows GUI

## Aesthetic and Minimalist Design:

The system is not cluttered with excessive use of icons and buttons. Tabs are used to separate different functionalities. A simple rectangle composition arrangement is used to model information.



**Recognition rather than Recall:** The use of colour schemes and icons act to denote functionalities. Example: 'Head Phone icon'. This design feature promotes recognition of rather than recall of system functionalities

# Case Study 2: Icon Design

Two simple icons communicating an activity in progress.

Both the icons are graphically simple, do the function of informing the status, and not complicated to understand. They use gradient in colorus (monochrome) to depict time progress through animation. The circular form express the abstract concept of time.

The state of 'please wait' is expressed in a pleasant peaceful unhurried manner.

In terms of construction, icons do not take expensive screen real estate; need very less computing memory; are amiable to both pixel as well as vector graphics.

The icon has achieved this by employing aesthetic principles in their form, colour, shape, configuration, motion & composition – all of them put together holistically resulting in a simple 'design'.



# Graphic Design – Website Layout

HCI-Designers besides being Engineers are Artists in the sense that they have to become sensitive to the visual language and master the use of visual elements in accordance to Principles



Too much of order. Not Interesting



Rhythmic spacing.  
Different shapes create interesting lay out.

# Graphic Design Case Study 3

A case study of a website's visual quality

The principles of Cognitive Science – Gestalt laws govern aesthetics.

## Aesthetics

Ordered Grid –  
Rows and Columns  
Good composition –  
Position w. r. t. area

Visual Balance –  
Symmetry / Asymmetry

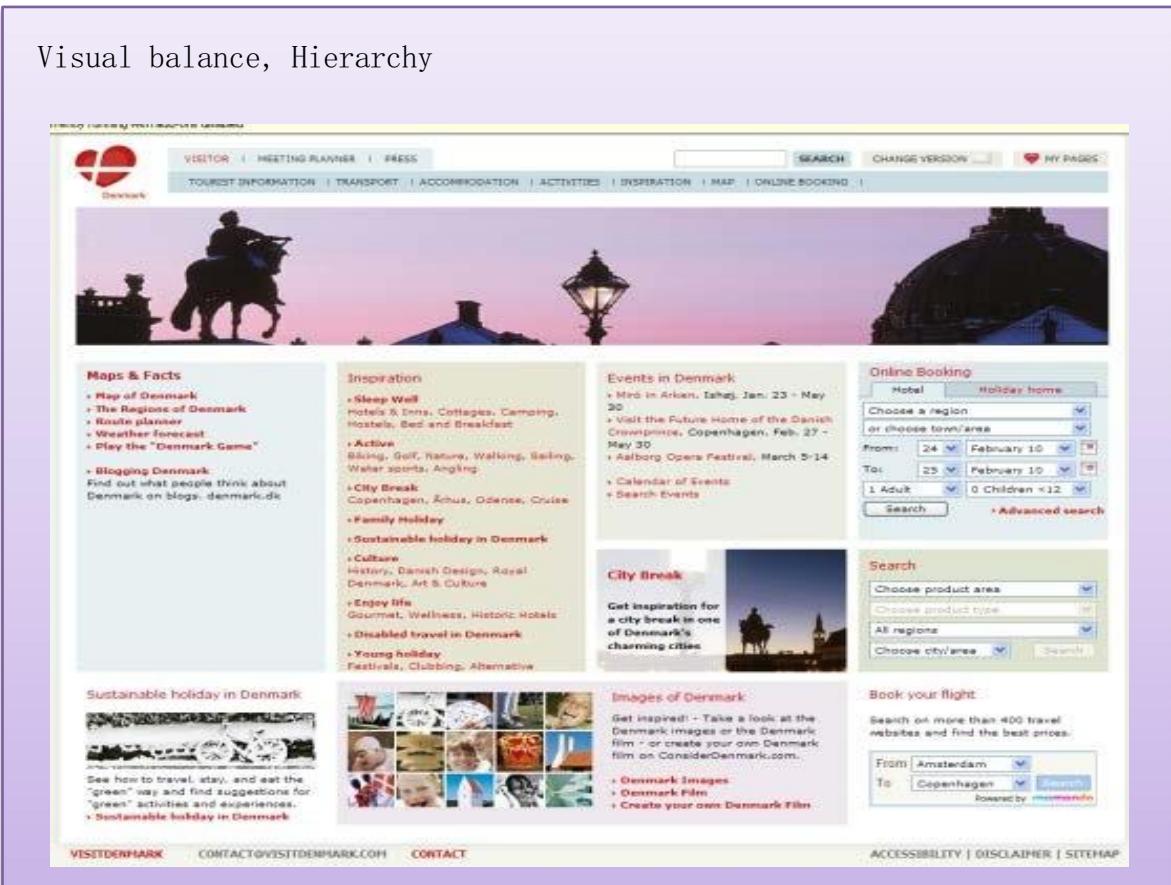
Low visual noise –

No clutter or crowding

## Color & Graphics –

Simple plain light reflective and absorbing colors with no fancy labels.

Visual balance, Hierarchy



*The UI -interface of a product is not a canvas for art nor a surface for advertising.*

- Aesthetics is a specialized discipline.
- It has both Science & Technology and Art.
- It is Qualitative and Quantitative judgment.
- Creative Designers are best equipped to decide on aesthetics as they are trained professionally.

# Home Work

From any computer or mobile screen, you are asked to pick one GUI which you do not like and another GUI which you like very much.

Analyze their constituting graphic / visual elements by applying the principles of aesthetics and find if you can attribute any aesthetic reasons for your ‘likes’ and ‘dislikes’. Keep aside the functional and the usability aspects for the time being.

# Home Work

Sketch as many alternatives as you can visualise for the two icons that depict activity progress happening in the background .

Conduct a quick survey from amongst your friends as to which of the icon concepts, you have come up with, are 'liked' by them.

You can ask them to rate each design for 10 points and empirically find the one that is most likely to be accepted in terms of aesthetics and the function representation.

You can also ask them to point out one visual element from your design that if changed will improve your design.



# HCI: Interactive System Design (Prototype Techniques)

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Learning Objective

- In the previous lecture, we learned about a method (contextual inquiry) to gather requirements for a design
- Designer can come up with ideas on the basis of this data
  - Typically more than one designs are proposed
- It is necessary to evaluate the alternative designs to find out the most appropriate one

# Learning Objective

- Interactive systems are designed following a user-centered design approach
  - Evaluation of the alternative design proposals should be done from the user's perspective
- Employing end users in evaluating designs is not easy
  - It is costly in terms of money, time, effort and manpower

# Learning Objective

- In the initial design phase, when the proposed design undergoes frequent changes, it is not advisable to even feasible to carry out evaluation with real users
- An alternative way to collect feedback on proposed design is to develop and evaluate “prototypes”

# Learning Objective

- In this lecture, we shall learn about the prototyping techniques used in interactive system design
- In particular, we will learn the following:
  - Why we need prototyping (already discussed in the previous slides)?
  - What are the techniques available (overview)?
  - How these techniques are used (details)?

# Prototyping

- A prototype is essentially a model of the system
  - The prototype (model) can have limited or full range of functionalities of the proposed system
- A widely used technique in engineering where the novel products are tested by testing a prototype

# Prototyping

- Prototypes can be “throw away” (e.g., scale models which are thrown away after they serve their purpose) or can go into commercial use
- In software development prototypes can be
  - Paper-based: likely to be thrown away after use
  - Software-based: can support few or all functionalities of the proposed system. May develop into full-scale final product

# Prototyping in HCI

- Essential element in user centered design
  - Is an experimental and partial design
  - Helps involving users in testing design ideas without implementing a full-scale system
- Typically done very early in the design process
  - Can be used throughout the design life cycle

# What to Prototype?

- Any aspect of the design that needs to be evaluated
  - Work flow
  - Task design
  - Screen layout
  - Difficult, controversial, critical areas

# Prototypes in HCI

- In HCI, prototypes take many forms
  - A storyboard (cartoon-like series of screen sketches)
  - A power point slide show
  - A video simulating the use of a system
  - A cardboard mock-up
  - A piece of software with limited functionality
  - Even a lump of wood

# Prototypes in HCI

- We can categorize all these different forms of prototypes in the three following groups
  - Low fidelity prototypes
  - Medium fidelity prototypes
  - High fidelity prototypes

# Low Fidelity Prototypes

- Basically paper mock-up of the interface look, feel, functionality
  - Quick and cheap to prepare and modify
- Purpose
  - Brainstorm competing designs
  - Elicit the user reaction (including any suggestions for further modifications)

# Low Fidelity Prototypes

**What to do**  
Touch a different color, or scan another item.



**What you selected**

**JPG Stroller**  
For children between 1-3 years old ...\$98.

Green  
 Blue  
 Red (out of stock)

Item	Style	Cost
JPG Stroller	Green	98.00 <span style="border: 1px solid black; padding: 2px;">Delete</span>

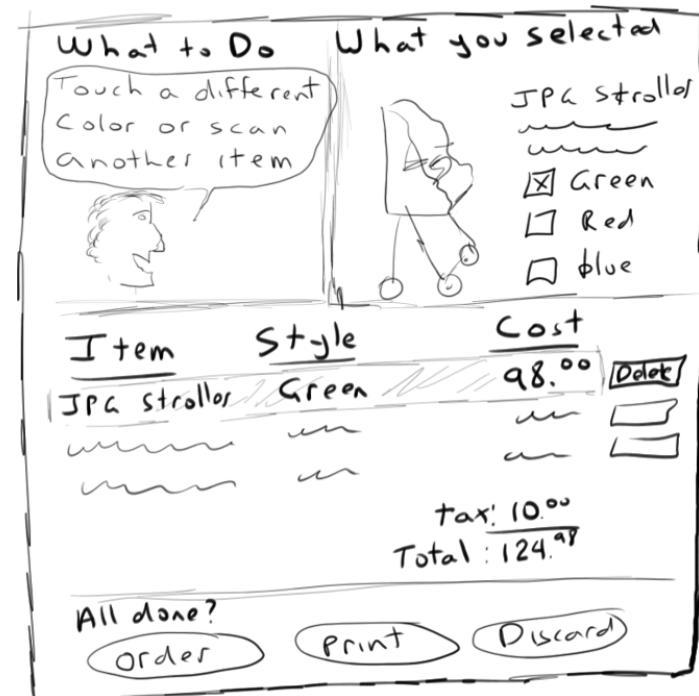
tax: 6.98

**Total:** \$104.98

**All done?**

Place your order Print this list Throw this list away

Interface of a proposed system



A sketch of the interface

# Low Fidelity Prototypes

- In a sketch, the outward appearance of the intended system is drawn
  - Typically a crude approximation of the final appearance
- Such crude approximation helps people concentrate on high level concepts
  - But difficult to visualize interaction (dialogue's progression)

# Low Fidelity Prototypes: Storyboarding

- Scenario-based prototyping
- Scenarios are scripts of particular usage of the system
- The following slides show an example storyboarding of a scenario of stroller-buying using an e-commerce interface

# Low Fidelity Prototypes: Storyboarding

Initial screen. Shows the layout of interface options.

<b>What to do</b> Find the item you want in the catalog and scan the bar code next to it. 	<b>What you selected</b>	
<u>Item</u>	<u>Style</u>	<u>Cost</u>
		tax: _____
		<b>Total:</b> \$ 0.00
<b>All done?</b>		
<input type="button" value="Place your order"/>	<input type="button" value="Print this list"/>	<input type="button" value="Throw this list away"/>

# Low Fidelity Prototypes: Storyboarding

Once a stroller is selected by the customer, its tag is scanned with a hand-held scanner. The JPG stroller details are displayed on the interface if the scanning is successful. Also, the option buttons become active after a successful scan.

**What to do**  
Touch a different color, or scan another item.



**What you selected**  
**JPG Stroller**  
For children between 1-3 years old ...\$98.  
 Green  
 Blue  
 Red (out of stock)

Item	Style	Cost
JPG Stroller	Green	98.00

tax: 6.98

**Total:** \$104.98

**All done?**

Place your order      Print this list      Throw this list away

# Low Fidelity Prototypes: Storyboarding

However, the customer can choose a different product at this stage and the same procedure is followed. For example, the customer may choose a stroller with a different color.

**What to do**  
Touch a different color, or scan another item.



**What you selected**

**JPG Stroller**  
For children between 1-3 years old ...\$98.

Green  
 Blue  
 Red (out of stock)

<u>Item</u>	<u>Style</u>	<u>Cost</u>
JPG Stroller	Blue	98.00

tax: 6.98

**Total:** \$104.98

**All done?**

Place your order      Print this list      Throw this list away

# Low Fidelity Prototypes: Storyboarding

Once the customer finalizes a product, the bill (invoice) is generated and displayed on the interface. The option buttons become inactive again.

**What to do**

To get your items, bring your printout to the front counter.



<u>Item</u>	<u>Style</u>	<u>Cost</u>
JPG Stroller	Blue	98.00

tax: 6.98

**Total:** \$104.98

**All done?**

Place your order      Print this list      Throw this list away

# Low Fidelity Prototypes: Storyboarding

- Here, a series of sketches of the *keyframes* during an interaction is drawn
  - Typically drawn for one or more typical interaction scenarios
  - Captures the interface appearance during specific instances of the interaction
  - Helps user evaluate the interaction (dialog) unlike sketches

# Low Fidelity Prototypes: Pictiv

- Pictiv stands for “plastic interface for collaborative technology initiatives through video exploration”
- Basically, using readily available materials to prototype designs
  - Sticky notes are primarily used (with plastic overlays)
  - Represent different interface elements such as icons, menus, windows etc. by varying sticky note sizes

# Low Fidelity Prototypes: Pictiv

- Interaction demonstrated by manipulating sticky notes
  - Easy to build new interfaces “on the fly”
- Interaction (sticky note manipulation) is videotaped for later analysis

# Medium Fidelity Prototypes

- Prototypes built using computers
  - More powerful than low fidelity prototypes
  - Simulates some but not all functionalities of the system
  - More engaging for end users as the user can get better feeling of the system
  - Can be helpful in testing more subtle design issues

# Medium Fidelity Prototypes

- Broadly of two types
  - **Vertical prototype** where in-depth functionalities of a limited number of selected features are implemented. Such prototypes help to evaluate common design ideas in depth.
  - Example: working of a single menu item in full

# Medium Fidelity Prototypes

- Broadly of two types
  - **Horizontal prototype** where the entire surface interface is implemented without any functionality. No real task can be performed with such prototypes.
  - Example: first screen of an interface (showing layout)

# Medium Fidelity Prototypes: Scenarios

- Computer are more useful (than drawing on paper as in storyboarding) to implement scenarios
  - Provide many useful tools (e.g., ppt slides, animation)
  - More engaging to end-users (and easier to elicit better response) compared to hand-drawn story-boarding

# Hi-Fidelity Prototypes

- Typically a software implementation of the design with full or most of the functionalities
  - Requires money, manpower, time and effort
  - Typically done at the end for final user evaluations

# Prototype and Final Product

- Prototypes are designed/used in either of the following:
  - ❖ **Throw-away:** prototypes are used only to elicit user reaction. Once their purpose is served, they are thrown away.
    - Typically done with low and some medium fidelity prototypes
  - ❖ **Incremental:** Product is built as separate components (modules). After each component is prototyped and tested, it is added to the final system
    - Typically done with medium and hi fidelity prototypes

# Prototype and Final Product

- Prototypes are designed/used in either of the following:
  - ❖ **Evolutionary:** A single prototype is refined and altered after testing, iteratively, which ultimately “evolve” towards the development of a final product
    - Typically done with hi-fidelity prototypes

# Prototyping Tools

- For (computer-based) medium and hi fidelity prototype developed, several tools are available
  - **Drawing tools**, such as Adobe Photoshop, MS Visio can be used to develop sketch/storyboards
  - **Presentation software**, such as MS Power Point with integrated drawing support are also suitable for low fidelity prototypes

# Prototyping Tools

- For (computer-based) medium and hi fidelity prototype developed, several tools are available
  - **Media tools**, such as Adobe flash can be used to develop storyboards. Scene transition is achieved by simple user inputs such as key press or mouse clicks

# Prototyping Tools

- For (computer-based) medium and hi fidelity prototype developed, several tools are available
  - **Interface builders**, such as VB, Java Swing with their widget libraries are useful for implementing screen layouts easily (horizontal prototyping). The interface builders also supports rapid implementation of vertical prototyping through programming with their extensive software libraries

# The Wizard of Oz Technique

- A technique to test a system that does not exist
- First used to test a system by IBM called the Listening Typewriter (1984)
  - Listening Typewriter was much like modern day voice recognition systems. User inputs text by uttering the text in front of a microphone. The voice is taken as input by the computer, which then identifies the text from it.

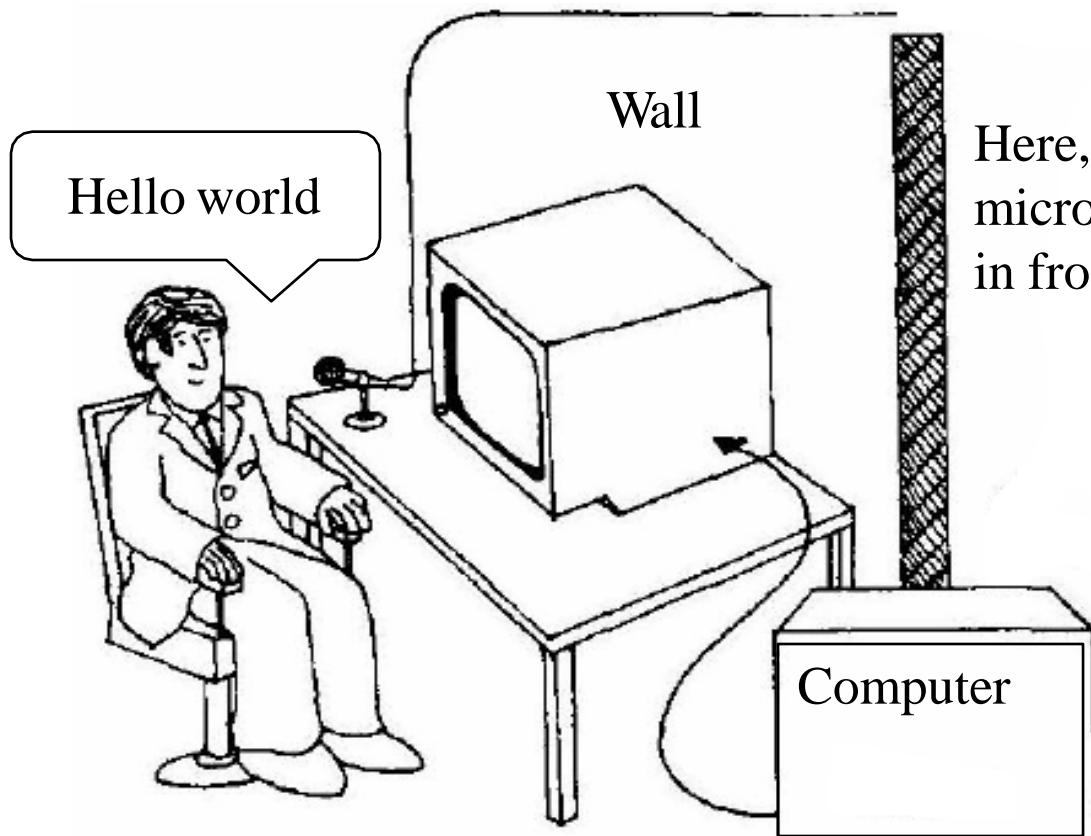
# The Wizard of Oz Technique

- Implementing voice recognition system is too complex and time consuming
- Before the developers embark on the process, they need to check if the “idea” is alright; otherwise the money and the effort spent in developing the system would be wasted
- Wizard of oz provides a mechanism to test the idea without implementing the system

# The Wizard of Oz Technique

- Suppose a user is asked to evaluate the performance of a listening typewriter
- He(She) is asked to sit in front of a computer screen
- A microphone is placed in front of him(her)
- He(She) is told that “whatever he(she) speaks in front of the microphone will be displayed on the screen”

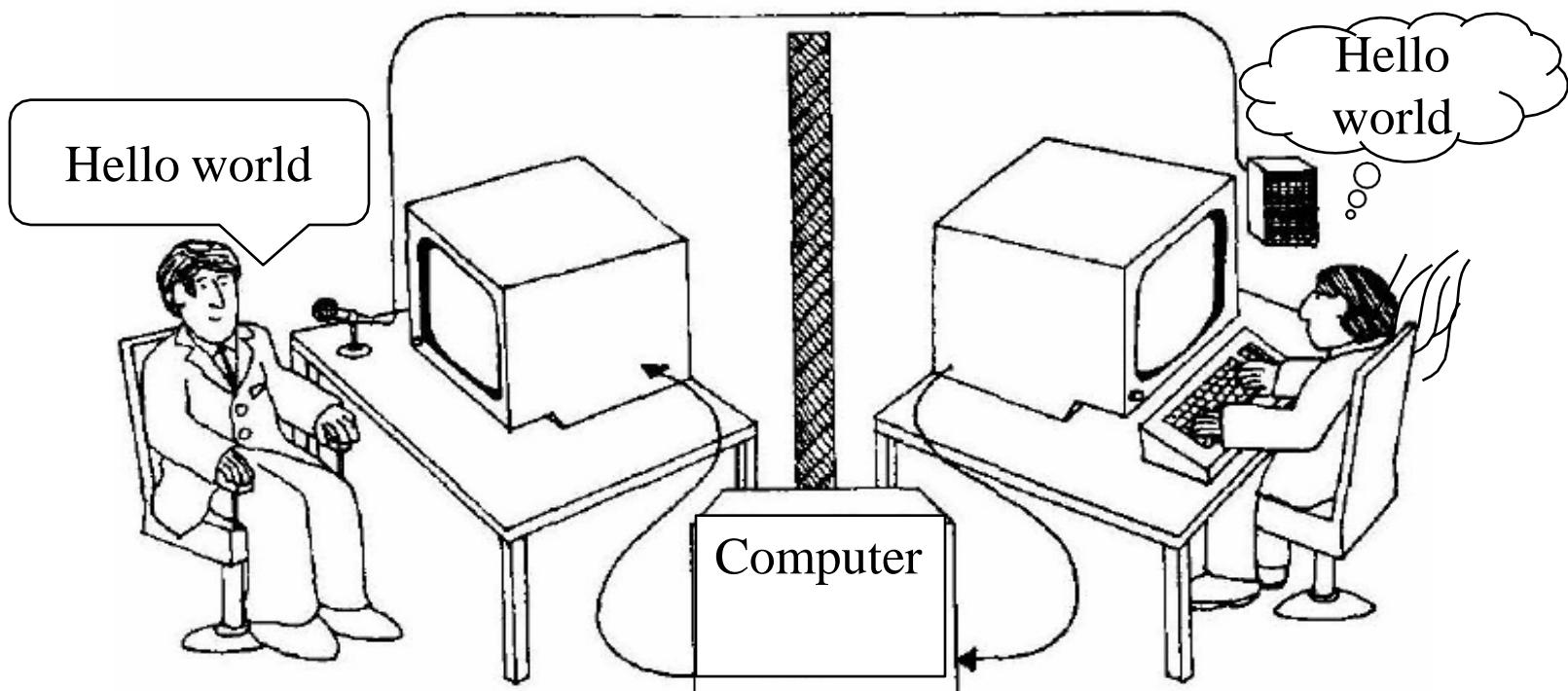
# The Wizard of Oz Technique



Here, the user sees: a screen, a microphone and the “computer” in front of an opaque wall.

# The Wizard of Oz Technique

This is what happens behind the wall. A typist (the wizard) listen to the utterance of the user, types it, which is then displayed on the user's screen. The user thinks the computer is doing everything, since the existence of the wizard is unknown to him.



# The Wizard of Oz Technique

- Human ‘wizard’ simulates system response
  - Interprets user input
  - Controls computer to simulate appropriate output
  - Uses real or mock interface
  - Wizard is typically hidden from the user; however, sometimes the user is informed about the wizard’s presence

# The Wizard of Oz Technique

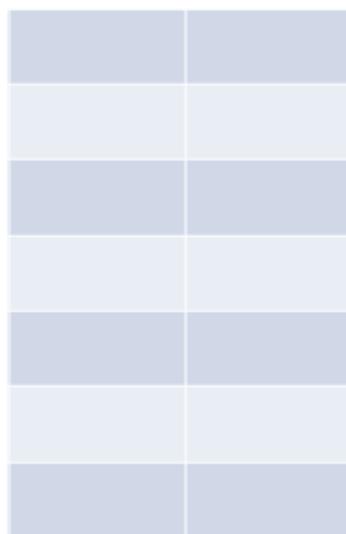
- The technique is very useful for
  - Simulating complex vertical functionalities of a system
  - Testing futuristic ideas

## HCI: Interactive System Design (Self Evaluation)

1. Consider the two word processing system: MS Word and Latex. Highlight their key differences from usability point of view.
2. Explain the concept of user centered design.
3. Discuss participatory design approach. How it is different from UCD?
4. Mention with brief explanation the main stages of an interactive system design life cycle. Why the iterative approach is required?
5. What is a prototype? Why we require it in interactive system design?
6. Explain the different prototyping techniques used in HCI. In which stages of the design cycle these are used and why?
7. Discuss the difference between vertical and horizontal prototypes. Suppose you are asked to evaluate the power point interface (in which you are viewing this slide). Discuss the evaluation approach in light of the vertical and horizontal prototypes.
8. List any three tools for prototype building. How each of these helps in developing prototypes?
9. Discuss the wizard of oz approach. Can we evaluate any design with this approach?
10. Suppose you want to test the touch-based interaction of a smart phone. Propose a wizard of oz approach for doing the same.
11. Conduct a quick Usability evaluation of your mobile phone & Compare it with the evaluation of your friends phone.

Rating out of 10

Effective to use - Functional



Efficient to use - Efficient

Error free in use - Safe

Easy to use - Friendly

Enjoyable in use - Pleasurable

Total :

12. Take a GUI which you do not like & one which you like.  
Analyses elements visually by applying principles of aesthetics to and find out if you can attribute aesthetic reasons for your like & dislike. Keep functional aspects out of the judgment for the time being.

13. Sketch as many alternatives as you can visualise for the two icons that depict activity progress happening in the background.

Conduct a quick survey from amongst your friends as to which of the icon concepts, you have come up with, are 'liked' by them.

You can ask them to rate each design for 10 points and empirically find out the one that is most likely to be accepted in terms of aesthetics & function representation.

You can also ask them to point out one visual element from your design that if changed will improve your design.



# Human Computer Interaction

## (Interactive System Design: Interviews and Questionnaires)

# Interviews & questionnaires: intro

- **Interviews** are “*conversations with a purpose*” (Kahn and Cannell, 1957)
- **Questionnaires** are like interviews
  - Can be given to lots of people to get wide general opinions
- Both techniques involve careful question design and planning
- There are no concrete rules to designing these, just guidelines to help us along the way

# **Interviews & questionnaires: goal**

- **Prior knowledge**
- **Skills and abilities**
- **Beliefs**
- **Personality traits**
- **Attitudes**
- **User satisfaction**

# Interviews & questionnaires: guidelines

- **Order of questions**
  - General before specific
- **Avoid complex/compound questions**
  - **Bad:** “How do you like this hair removal product compared to the ones you’ve owned?”
  - **Better:** “Why do you like this hair removal product? … Have you owned others before? … If so, did you like them?”
- **Short and sweet**  **Clear and concise**
  - **Bad:** Wouldn’t you agree that exceedingly long queries exemplify poor question formation design because it is difficult to remember each part and what’s more, it facilitates the blurring of one’s train of thought - especially if it’s worded terribly poorly?
- **Each question should contribute to the evaluation goal**
  - Think about what you would do with the answer; YES/NO responses aren’t that useful as they don’t convey WHY
  - Ask open-ended questions at the end to gather info. you may have missed

# Interviews & questionnaires: response bias

- **Beware of response bias**
  - When answers received don't reflect the truth, the answers become useless!
- **Types of response errors**
  - Motivated errors: *hiding info to create a good impression*
  - Memory errors: *not being able to remember*
  - Communication errors: *questions are not clear OR answers not clear*

# Interviews & questionnaires: response bias

- Response bias example 1: *probing*
  - **Bad:**

“Have you ever driven a car while legally drunk?”
  - **Better:**

“There are times when it’s impossible to find alternative transportation after drinking with friends at a party. Have you ever been in such a situation and had to drive home?”
- Response bias example 2: *embarrassing*
  - **Bad:**

“How much time do you spend reading the newspaper?”
  - **Better:**

“Did you have a chance to read the newspaper yesterday?”  
(If respondent says yes)  
“About how much time did you spend reading it yesterday?”

# Interviews & questionnaires: response bias

- Response bias example 3: *asking ppl to organize info*
  - **Bad:**

“How many hours did you use a word processor yesterday?”
  - **Better:**

“Below we list the hours for yesterday in half hour slots. Please mark with X those half hour slots in which you used a word processor.

6:00 AM \_\_

6:30 AM \_\_ ...”

# Interviews & questionnaires: response bias

- **Avoiding bias:**
  - Aim to be neutral – biases can be introduced unconsciously!
  - Take care of wording!
  - Users may be embarrassed to ask the meanings of gobbledegook
  - Avoid directing and leading questions
    - **Bad:** “Your ‘Treasure Troll’-like hairstyle looks great! What do you think of the hairstyles of toys these days?”

# Questionnaires: response formats

- **Check boxes**
  - Gender
- **Ranges**
  - age ranges
- **Likert scales**
  - opinions, attitudes, beliefs, user satisfaction
- **Other response scales**
  - Semantic differential scales for bipolar attitudes

# Questionnaires: response formats - ranges

- **Ensure that ranges:**
  - Don't overlap
  - Are appropriate
  - “How many hours do you spend on the Internet per week?”
  - **Bad:**            0-1        1-2        2-3        3-40        40+
- **Ordering of scales should be:**
  - Consistent with other questions
  - Intuitive

# Questionnaires: Response Formats – Likert Scales

- **Pick a number from a range of numbers**
  - “I’m falling asleep in this stuffy classroom.”

Strongly agree	Agree	Neutral	Disagree	Strong disagree
1	2	3	4	5

- **Steps:**
  - Create statements about features to be evaluated
  - Place statements into groups
  - Choose proper scale
  - Select statements for final questionnaire

# Questionnaires: Response Formats – Semantic Differential Scales

- Less popular than Likert scales
- Explores bipolar attitudes
  - Each attitude pair represented as adjectives
  - Participant chooses between extremes
- Example:
  - “How often do you watch trashy soap operas on TV?”

Never	+	+	+	+	Very often
1	2	3	4	5	6

# Case Studies on Model-Based Design

Professor Ram Mohana Reddy Gudde  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Learning Objective

- In the previous lectures, we discussed the idea of models and the principles of model-based design
  - We discussed the different types of models used in HCI
  - We learned in details of four different models, namely: KLM, (CMN)GOMS, Fitts' Law and Hick-Hyman Law

# Learning Objective

- In this lecture, we shall see a specific case study on model-based design, namely: the design of virtual keyboards, to understand the idea better

# Virtual Keyboards

- Before going into the design, let us first try to understand the virtual keyboard (VK)
- We know what a physical keyboard is
  - The input device through which you can input characters
- Although the physical keyboards are ubiquitous and familiar, sometimes it is not available or feasible

# Virtual Keyboards

- Suppose we want to input the characters in a mobile device (e.g., our mobile phone or iPad)
  - Physical keyboards make the system bulky and thus reduces the mobility
- Sometimes the users may not have the requisite motor control to operate the physical keyboards
  - For example, persons with cerebral palsy, paraplegia etc.

# Virtual Keyboards

- In such scenario, the Virtual Keyboards (VKs) are useful
  - A VK is an on-screen representation of the physical keyboard (see the image which shows the text input in an iPad with a VK)



# VK Design Challenge

- The iPad example in the previous slide shows a QWERTY layout (i.e., key arrangement)
  - That's because the typing is two-hand and QWERTY layout is suitable for two-hand typing
- However, in many cases, VK is used with single-hand typing (particularly for small devices where one hand holds the device)

# VK Design Challenge

- Since the QWERTY layout is good for the two-hand typing, we have to find out the alternative “efficient” layout
  - Efficiency, in the context of keyboards in general and the VK in particular, is measured in terms of character entry speed (characters/sec or CPS, words/min or WPM etc)

# VK Design Challenge

- Thus, what we want is a VK layout for a single hand typing that allows the user to input characters with high speed and accuracy
- Mathematically, for a  $N$  character keyboard, we have to find the best among  $N!$  possible key arrangements

# VK Design Challenge

- Thus, it is known as a typical “search” problem
  - We want to search for a solution in a search space of size  $N!$
  - Note the “huge” size of the search space (for ex., if  $N = 26$  letters of English alphabet + 10 numerals = 36, then the size of the search space is  $36!$ )

# What We Can Do

- We can apply the standard design life cycle
- Drawbacks
  - We can not check all the alternatives in the search space  
(that will in fact take millions of years!)
- If the designer is experienced, he(she) can chose a small subset from the search space based on intuition

# What We Can Do

- The alternatives in the subset can be put through the standard design life cycle for comparison
  - However, an empirical comparison still requires great time and effort
- Alternatively, we can use model-based approach to compare the alternatives

# GOMS Analysis

- We can compare the designs in the subset using a GOMS analysis (also called CTA or cognitive task analysis)
- In order to do so, we first need to identify one or a set of “representative tasks”

# GOMS Analysis

- What is a task here?
  - To input a series (string) of characters with the VK
- Remember, we should have a representative task
  - That means, the string of characters that we chose should represent the language characteristics

# GOMS Analysis

- How to characterize a language?
- There are many ways
  - One simple approach is to consider unigram character distribution, which refers to the frequency of occurrence of characters in any arbitrary sample of the language (text)
  - Bigram distribution, which refers to the frequency of occurrence of character pairs or bigrams in any arbitrary sample, is another popular way to characterize a language

# GOMS Analysis

- In order to perform GOMS analysis, we need to have character string(s) having language characteristics (say, the unigram distribution of characters in the string(s) match(es) to that of the language)
  - How to determine such string(s)?
- We can use a language corpus for the purpose

# Corpus

- Corpus (of a language) refers to a collection of texts sampled from different categories (genres)
  - Stories, prose, poem, technical articles, newspaper reports, mails ...
- It is assumed that a corpus represents the language (by capturing its idiosyncrasies through proper sampling)

# Corpus

- However, corpus development is not trivial (requires great care to be truly representative)
- The good news is, already developed corpora are available for many languages (e.g., British National Corpus or BNC for English)
  - We can make use of those

# Corpus-based Approach

- How to use a corpus to extract representative text?
  - Get hold of a corpus
  - Extract the representative text through some statistical means (for example, cross-entropy based similarity measure)

# Cross-Entropy Based Similarity Measure

- Let  $X$  be a random variable which can take any character as its value
- Further, let  $P$  be the probability distribution function of  $X$  [i.e.,  $P(x_i) = P(X = x_i)$ ]
- We can calculate the “entropy”, a statistical measure, of  $P$  in the following way

$$H(P) = -\sum_i P(x_i) \log_2 P(x_i)$$

# Cross-Entropy Based Similarity Measure

- Now, suppose there are two distributions, P and M
- We can calculate another statistical measure, called “cross-entropy”, of the two distributions

$$H(P, M) = - \sum_i P(x_i) \log_2 M(x_i)$$

# Cross-Entropy Based Similarity Measure

- The cross-entropy measure can be used to determine similarity of the two distributions
  - Closer  $H(P,M)$  is to  $H(P)$ , the better approximation  $M$  is of  $P$  (i.e.,  $M$  is similar to  $P$ )
- We can use this idea to extract representative text from a corpus

# Cross-Entropy Based Similarity Measure

- Let  $P$  be the unigram probability distribution of the language
  - This can be determined from the corpus. Simply calculate the character frequencies in the corpus. Since the corpus is assumed to represent the language, the character frequencies obtained from the corpus can be taken as representative of the language
  - Calculate  $H(P)$

# Cross-Entropy Based Similarity Measure

- Take random samples of texts from the corpus and determine the unigram character distribution of the sample text, which is  $M$
- Next, calculate  $H(P, M)$
- The sample text for which  $H(P, M)$  is closest to  $H(P)$  will be our representative text

# Problem with GOMS-based CTA

- Thus, we can perform GOMS analysis
- However, there is a problem
  - The text is usually large (typically  $>100$  characters to make it *reasonably* representative), which makes it tedious to construct GOMS model

# Problem with GOMS-based CTA

- We need some other approach, which is not task-based, to address the design challenge
  - Task-based approaches are typically tedious and sometimes infeasible to perform
- In the next lecture, we shall discuss one such approach, which is based on the Fitts' law and the Hick-Hyman law

# Learning Objective

- In the previous slides, we discussed the challenge faced by the Virtual Keyboards (VK) designers
  - The objective of the VK designer is to determine an efficient layout
  - The challenge for VK designer is to identify the layout from a large design space
  - We saw the difficulties in using the standard design life cycle
  - We explored the possibility of using GOMS in the design and discussed its problems

# Alternative Design Approach

- Here, Let us see another way of addressing the issue, which illustrates the power of model-based design
- We saw the problem with GOMS in VK design
  - The problem arises due to the task-based analysis, since identifying and analyzing tasks is tedious if not difficult and sometimes not feasible
- We need some other approach that is not task based
  - Fitts' Law and Hick-Hyman Law can be useful for the purpose as they do not require task-based analysis

# Fitts'-Digraph Model

- The alternative approach makes use of the Fitts' Diagraph (FD) model
- FD model was proposed to *compute* the user performance for a VK from layout specification
  - Layout in terms of keys and their positions
  - Performance in text entry rate

# Fitts'-Digraph Model

- The FD model has three components
  - **Visual Search Time (RT)**: time taken by a user to locate a key on the keyboard. The Hick-Hyman Law is used to model this time

$$RT = a + b \log_2 N$$

N is the total number of keys, a and b are empirically-determined constants

# Fitts'-Digraph Model

- The FD model has three components
  - **Movement Time (MT)**: time taken by the user to move his hand/finger to the target key (from its current position). This time is modeled by the Fitts' Law

$$MT_{ij} = a' + b' \log_2 \left( \frac{d_{ij}}{w_j} + 1 \right)$$

$MT_{ij}$  is the movement time from the source (i-th) to the target (j-th) key,  $d_{ij}$  is the distance between the source and target keys,  $w_j$  is the width of the target key and  $a'$  and  $b'$  are empirically-determined constants

# Fitts'-Digraph Model

- The FD model has three components
  - **Digraph Probability:** It is the probability of occurrence of character pairs or digraphs, which is determined from a corpus

$$P_{ij} = f_{ij} / \sum_{i=1}^N \sum_{j=1}^N f_{ij}$$

- $P_{ij}$  is the probability of occurrence of the  $i$ -th and  $j$ -th key, whereas  $f_{ij}$  is the frequency of the key pair in the corpus

# Fitts'-Digraph Model

- Using the movement time formulation between a pair of keys, an average (mean) movement time for the whole layout is computed

$$MT_{MEAN} = \sum_{i=1}^N \sum_{j=1}^N MT_{ij} \times P_{ij}$$

- The mean movement time is used, along with the visual search time, to compute the user performance for the layout

# Fitts'-Digraph Model

- Users' Performance is measured in terms of the characters/second (CPS) or words/minute (WPM)
- Performances for two categories of users, namely: novice and expert users, are computed

# Fitts'-Digraph Model

- Novice User Performance: Users are assumed to be unfamiliar with the layout. Hence, such users require time to search for the desired key before selecting the key

$$CPS_{Novice} = \frac{1}{RT + MT_{MEAN}}$$

$$WPM = CPS \times (60 / W_{AVG})$$

$W_{AVG}$  is the average number of characters in a word. For example, English words have 5 characters on average

# Fitts'-Digraph Model

- Expert User Performance: An expert user is assumed to be thoroughly familiar with the layout. Hence, such users don't require the visual search time

$$CPS_{Expert} = \frac{1}{MT_{MEAN}}$$

$$WPM = CPS \times (60 / W_{AVG})$$

$W_{AVG}$  is the average number of characters in a word. For example, English words have 5 characters on average

# Using the FD Model

- If you are an expert designer
  - You have few designs in mind (experience and intuition helps)
  - Compute WPM for those designs
  - Compare the performance of competing designs

# Using the FD Model

- Otherwise
  - Perform *design space exploration* – search for a good design in the design space using algorithm
- Many algorithms have been developed for design space exploration such as dynamic simulation, Metropolis algorithm and genetic algorithm
  - We shall discuss one such algorithm (Metropolis algorithm) to illustrate the idea

# Metropolis Algorithm

- A “Monte Carlo” method widely used to search for the minimum energy (stable) state of molecules in statistical physics
- We map our problem (VK design) to a minimum-energy state finding problem in statistical physics

# Metropolis Algorithm

- We map a layout to a molecule (keys in the layout serves the role of atoms)
- We redefine performance as the average movement time, which is mapped to the energy of the molecule
- Thus, our problem is to find a layout with minimum energy

# Metropolis Algorithm

- Steps of the algorithm
  - Random walk: pick a key and move in a random direction by a random amount to reach a new configuration (called a *state*)
  - Compute energy (average movement time) of the state
  - Decide whether to retain new state or not and iterate

# Metropolis Algorithm

- The decision to retain/ignore the new state is taken on the basis of the decision function, where  $\Delta E$  indicates the energy difference between the new energy and old energy states ( $\Delta E = \text{energy of new state} - \text{energy of old state}$ )

$$W(O - N) = \begin{cases} e^{-\frac{\Delta E}{kT}} & \Delta E > 0 \\ 1 & \Delta E \leq 0 \end{cases}$$

# Metropolis Algorithm

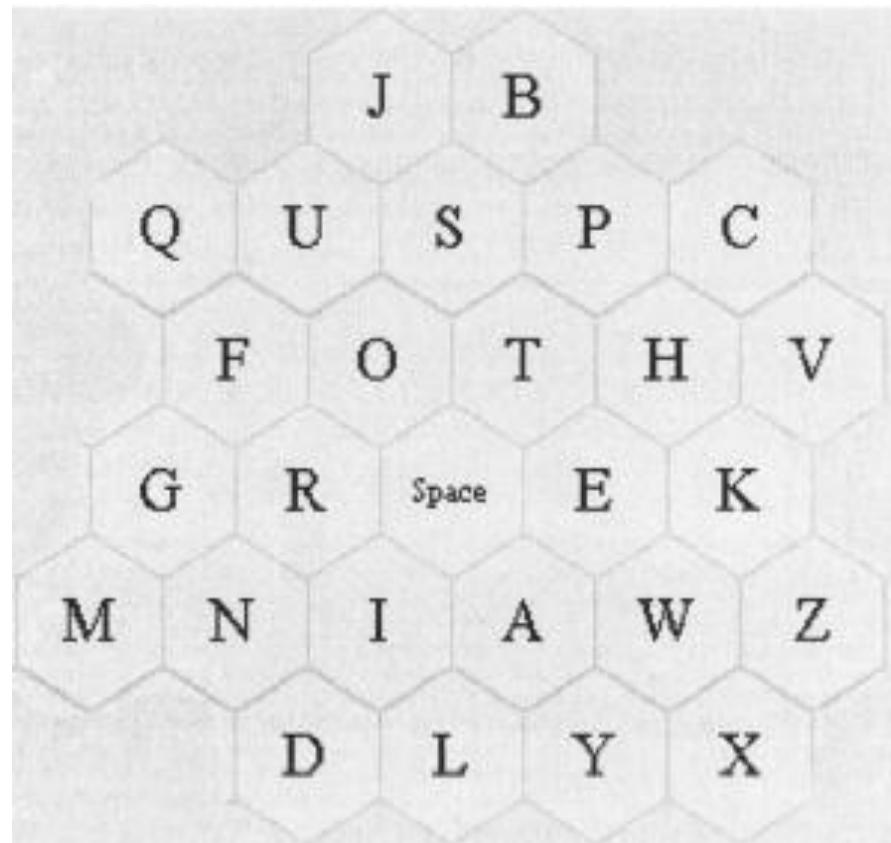
- $W$  is probability of changing from old to new configuration
- $k$  is a coefficient
- $T$  is “temperature”
- Initial design: a “good” layout stretched over a “large” space

# Metropolis Algorithm

- Note the implications of the decision function
  - If energy of the new state is less than the current state, retain the new state
  - If the new state is having more energy than the current state, don't discard the new state outright. Instead, retain the new state if the probability  $W$  is above some threshold value. This steps helps to avoid local minima
- To reduce the chances of getting stuck at the local minima further, “Simulated Annealing” is used
  - Bringing “temperature” through several up & down cycles

# Metropolis Algorithm

An example VK layout, called the Metropolis layout, is shown, which was designed using the Metropolis algorithm



# Some VK Layouts with Performance

- QWERTY
  - 28 WPM (novice)
  - 45.7 WPM (expert)
- FITALY
  - 36 WPM (novice)
  - 58.8 WPM (expert)

FITALY Keyboard

Z	V	C	H	W	K
F	I	T	A	L	Y
		N	E		
G	D	O	R	S	B
Q	J	U	M	P	X

# Some VK Layouts with Performance

- QWERTY
  - 28 WPM (novice)
  - 45.7 WPM (expert)
- FITALY
  - 36 WPM (novice)
  - 58.8 WPM (expert)
- OPTI II
  - 38 WPM (novice)
  - 62 WPM (expert)

OPTI II Keyboard

Q	K	C	G	V	J
	S	I	N	D	
W	T	H	E	A	M
	U	O	R	L	
Z	B	F	Y	P	X

# Some VK Layouts with Performance

- The layouts mentioned before were not designed using models
- They were designed primarily based on designer's intuition and empirical studies
- However, the performances shown are computed using the FD model

# Some VK Layouts with Performance

- ATOMIK – a layout will be designed using the slightly modified Metropolis algorithm
- Performance of the ATOMIK layout
  - 41.2 WPM (novice)
  - 67.2 WPM (expert)



# Some VK Layouts with Performance

- Note the large performance difference between the ATOMIK and other layouts
- This shows the power of model-based design, namely: a (significant) improvement in the performance without increasing the design time and effort (since the design can be mostly automated)

# HCI: Model-based Design (GOMS Family of Models)

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Learning Objective

- In the previous lectures, we learnt the design process involved in interactive system design
- We learnt that the interactive systems are designed using the Interactive System Design Life Cycle (ISDLC)
  - Consisting of the stages for requirement identification, design, prototyping and evaluation
  - Highly iterative (iterative life cycle is time consuming and also requires money (for coding and testing))
- It is always good if we have an alternative method that reduces the time and effort required for the design life cycle
- Model-based design provides one such alternative method

# Learning Objective

- In this lecture, we will learn the model-based design
- In particular, we will learn the following
  - Motivation for model-based design approach
  - The idea of models
  - Types of models used in HCI

# Idea of a Model

- A ‘model’ in HCI refers to “a representation of the user’s interaction behavior under certain assumptions”
- The representation is typically obtained from empirical studies (collecting and analyzing data from end users)
  - The model represents behavior of average users, not individuals

# Motivation

- Suppose we are trying to design an interactive system
- First, let us identify requirements (“know the user”) using the methods such as contextual inquiry
  - Time consuming and tedious process
- Instead of going through the process, it would have been better if we have a “model of the user”
- By encompassing information about user behaviour, a model helps in alleviating the need for requirement identification process
- Such requirements are already known from the model
- Once the requirements are identified, designer ‘propose’ design(s)

# Motivation Contd...

- Typically, more than one designs are proposed
  - The competing designs need to be evaluated
- This can be done by evaluating either the prototypes (in the early design phase) or the full (final) system (at the final stages of the design) with end users
  - End user evaluation is a must in user centered design
- Like requirement identification stage, the continuous evaluation with end users is also money and time consuming
- If we have a model of end users as before, we can employ the model to evaluate the design
  - Because the model already captures the end user characteristics, no need to go for real users

# Summary

- A model is assumed to capture behavior of an average user of an interactive system
- User behavior and responses are what we are interested in knowing during ISDLC
- Thus by using these models, we can fulfill the key requirement of interactive computing system design (without actually going to the end user)
  - Saves lots of time, effort and money

# Types of Models

- In this lecture, we shall discuss two broad categories of the models used in HCI
- Descriptive/Prescriptive Models: Some models in HCI are used to explain/describe user behavior during interaction in qualitative terms. An example is the Norman's model of interaction (to be discussed later). These models help in formulating (prescribing) guidelines for interface design
- Predictive Engineering Models: These models can “predict” the behaviour of a user in quantitative terms. An example is the GOMS model (to be discussed later), which can predict the task completion time of an average user for a given system. We can actually “compute” such behaviour.

# Predictive Engineering Models

- The predictive engineering models used in HCI are of three types, namely:
  - **Formal (System) Models**
  - **Cognitive (User) Models**
  - **Syndetic (Hybrid) Model**

# Formal (System) Model

- In these models, the interactive system (interface and interaction) is represented using ‘formal specifications’
  - For example, the interaction modeling using state transition networks
- Essentially models of the ‘external aspects’ of interactive system (what is seen from outside)

# Formal (System) Model

- Interaction is assumed to be a transition between states in a ‘system state space’
  - A ‘system state’ is characterized by the state of the interface (what the user sees)
- It is assumed that certain state transitions increase usability while the others do not
- The models try to predict if the proposed design allows the users to make usability-enhancing transitions
  - By applying ‘reasoning’ (manually or using tools) on the formal specification.

# Cognitive (User) Models

- These models capture the user's thought (cognitive) process during interaction
  - For example, a GOMS model tells us the series of cognitive steps involved in typing a word
- Essentially models are the ‘internal aspects’ of interaction (what goes on inside user’s mind)
- Usability is assumed to depend on the ‘complexity’ of the thought process (cognitive activities)
  - Higher complexity implies less usability

# Cognitive (User) Models

- Cognitive activities involved in interacting with a system is assumed to be composed of a series of steps (serial or parallel)
  - More the number of steps (or more the amount of parallelism involved), the more complex the cognitive activities are
- These models try to predict the number of cognitive steps involved in executing the ‘representative’ tasks with the proposed designs
  - Which leads to an estimation of usability of the proposed design

# Syndetic (Hybrid) Model

- HCI literature mentions one more type of model, called ‘Syndetic’ model
- In this model, both the system (external aspect) and the cognitive activities (internal aspect) are combined and represented using formal specification
- This model is rather complex and rarely used, hence it is outside the scope of this IT351: HCI Course.

# Cognitive Models in HCI

- Although we said before that the cognitive models are models of human thinking (thought) process, they are not exactly treated as the same in HCI
- Since interaction is involved, cognitive models in HCI not only model the human cognition (thinking) alone, but the perception and motor actions also (as interaction requires ‘perceiving what is in front’ and ‘acting’ after the decision making).

# Cognitive Models in HCI

- Thus cognitive models in HCI should be considered as the models of human perception (perceiving the surrounding), cognition (thinking in the ‘mind’) and motor action (result of thinking such as the hand movement, the eye movement etc.)
- In HCI, broadly three different approaches are used to model the cognition
  - Simple models of human information processing
  - Individual models of human factors
  - Integrated cognitive architectures

# Simple Models of Human Information Processing

- These are the earliest cognitive models used in HCI
- These model complex cognition as a series of simple (primitive/atomic) cognitive steps
  - Most well-known and widely used models based on this approach is the GOMS family of models
- Due to its nature, the application of such models to identify the usability issues is also known as the “Cognitive Task Analysis (CTA)”

# Individual Models of Human Factors

- In this approach, individual human factors such as manual (motor) movement, eye movement, decision time in the presence of visual stimuli etc. are modeled
  - These models are analytical expressions to compute the task execution times in terms of interface and cognitive parameters
- Examples are: the Fitts' Law, the Hick-Hyman Law

# Integrated Cognitive Architectures

- In this approach, the whole human cognition process (including perception and motor actions) is modelled
  - These models capture the complex interaction between different components of the cognitive mechanism unlike the first approach
  - Combines all human factors in a single model unlike the second approach
- Examples are MHP, ACT-R/PM, SOAR

# Model-based Design Limitations

- As we mentioned before, model-based design reduce the need for real users in ISDLC
- However, they can not completely eliminate the role played by the real users
- We still need to evaluate the designs with real users, albeit during the final stages
  - Model-based design can be employed in the initial stages

# Model-based Design Limitations

- The following are the key limitations:
  - The present models are not complete in representing average end user (they are very crude approximations only)
  - The models can not capture individual user's characteristics (only models the average user behavior)

# HCI:Model-based Design (GOMS Family of Models)

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Learning Objective

- Earlier, we learned the idea of model-based design in HCI
- We also discussed the type of models used in HCI
  - The concepts of prescriptive and predictive models
  - Different types of predictive engineering models
- Here, we will be dealing with a type of predictive engineering models known as “simple model of human information processing”
- GOMS family of models is the best known examples of the above type of predictive engineering model used in the “Interface Design”
  - GOMS stands for **G**oals, **O**perators, **M**ethods and **S**election Rules

# GOMS Family of Models

- GOMS is a modeling technique (more specifically, a family of modeling techniques) that analyzes the user complexity of interactive systems design. It is used by the software designers to model the user's behaviour. The user's behaviour is modelled in terms of Goals, Operators, Methods and Selection rules
- A GOMS model consists of *Methods that are used to achieve Goals*.
- A *Method is a sequential list of Operators that the user performs and (sub)Goals that must be achieved*
- If there is more than one *Method which may be employed to achieve a Goal, a Selection rule is invoked to determine what Method to choose, depending on the context.*

# GOMS Family of Models

- **What is GOMS?**
  - Description of the knowledge that a user must have to carry out tasks on a device or system
  - Representation of the “how to do it” knowledge that is required by a system in order to get the intended tasks accomplished.
- **What does a GOMS task analysis involve?**
  - Involves defining and then describing the user’s
    - Goals:
      - Something that the user tries to accomplish (action-object pair, e.g. delete word)
      - Include context
    - Methods:
      - Well learned sequence of steps that accomplish a task
      - How do you do it on this system? (could be long and tedious...)
    - Selection Rules:
      - Only when there are clear multiple methods for the same goal.
    - Operators:
      - Elementary perceptual, cognitive and motor acts that cause change (external vs. mental)
      - Also uses action-object pair (e.g. press key, select menu, make gesture, speak command...)
      - mostly defined by hardware and lower-level software.

# GOMS: An Example

- **File & Directory Operations:**
  - Delete a file, move a file, delete a directory, move a directory.
- **GOMS Analysis – File & Directory Operations:**
  - Method for goal: delete a file.
    - Step 1. drag file to trash.
    - Step 2. Return with goal accomplished.
  - Method for goal: move a file.
    - Step 1. drag file to destination.
    - Step 2. Return with goal accomplished.
  - Method for goal: delete a directory.
    - Step 1. drag directory to trash.
    - Step 2. Return with goal accomplished.
  - Method for goal: move a directory.
    - Step 1. drag directory to destination.
    - Step 2. Return with goal accomplished.

# GOMS: An Example

- **GOMS Analysis – File & Directory Operations - A Better Version:**
  - Method for goal: delete an object.
    - Step 1. drag object to trash.
    - Step 2. Return with goal accomplished.
  - Method for goal: move an object.
    - Step 1. drag object to destination.
    - Step 2. Return with goal accomplished.
- **GOMS Analysis – the Drag Operation**
  - Method for goal: drag item to destination.
    - Step 1. Locate icon for item on screen.
    - Step 2. Move cursor to item icon location.
    - Step 3. Hold mouse button down.
    - Step 4. Locate destination icon on screen.
    - Step 5. Move cursor to destination icon.
    - Step 6. Verify the destination icon.
    - Step 7. Release mouse button.
    - Step 8. Return with goal accomplished.

# **Learning Objective**

- The GOMS family consists of **FOUR** models
  - **Keystroke Level Model or KLM**
  - Original GOMS proposed by **Card, Moran and Newell**, popularly known as **(CMN) GOMS**
  - **Natural GOMS Language or NGOMSL**
  - **Cognitive Perceptual Motor or (CPM)GOMS**  
[also known as **Critical Path Method GOMS**]

# Keystroke Level Model (KLM)

- The KLM - GOMS model was proposed way back in 1980 by Card, Moran and Newell; retains its popularity even today
- This is the earliest model to be proposed in the GOMS family (and one of the first predictive models used in HCI)
- The KLM model provides a quantitative tool (like other predictive engineering models)
  - The model allows a designer to ‘predict’ the time it takes for an average user to execute a task using an interface and interaction method
  - For example, the model can predict how long it takes to close this PPT using the “close” menu option

# How KLM Works

- In KLM, it is assumed that any decision-making task is composed of a series of ‘elementary’ cognitive (mental) steps, that are executed in sequence
- These ‘elementary’ steps essentially represent low-level cognitive activities, which can not be decomposed any further
- The method of breaking down a higher-level cognitive activity into a sequence of elementary steps is simple to understand, provides a good level of accuracy and enough flexibility to apply in practical design situations

# How KLM Works

- In KLM, we build a model for task execution in terms of operators
  - That is why KLM belongs to the cognitive task analysis (CTA) approach to design
- For this, we need to choose one or more representative task scenarios for the proposed design
- Next, we need to specify the design to the point where keystroke (operator)-level actions can be listed for the specific task scenarios
- Then, we have to figure out the best way to do the task or the way the users will do it
- Next, we have to list the keystroke-level actions and the corresponding physical operators involved in doing the task

# How KLM Works

- If necessary, we may have to include operators when the user must wait for the system to respond (as we discussed before, this step may not be ignored most of the times for modern-day computing systems)
- In the listing, we have to insert mental operator M when user has to stop and think (or when the designer feels that the user has to think before taking next action)
- Once we list in proper sequence all the operators involved in executing the task, we have to do the following:
  - Look up the standard execution time for each operator
  - Add the execution times of the operators in the list

# How KLM Works

- The total of the operator times obtained in the previous step is “the time estimated for an average user to complete the task with the proposed design”
- If there are more than one design, then we can estimate the completion time of the same task with the alternative designs
  - The design with the least estimated task completion time will be the best design

# The Idea of Operators

- To understand how the model works, we first have to understand this concept of ‘elementary’ cognitive steps
- These elementary cognitive steps are known as *operators*
  - For example, a key press, mouse button press and release etc.
- Each operator takes a pre-determined amount of time to perform
- The operator times are determined from the empirical data (i.e., data collected from several users over a period of time under different experimental conditions)
  - That means, operator times represent average user behaviour (not the exact behaviour of an individual)

# The Idea of Operators

- The empirical nature of operator values indicate that, we can predict the behavior of average user with KLM
  - The model can not predict individual traits
- There are 7 operators defined, belonging to three broad groups
  - Physical (motor) operators
  - Mental operator
  - System response operator

# Physical (Motor) Operators

- There are five operators, that represent five elementary motor actions with respect to an interaction

Operator	Description
<b>K</b>	The motor operator representing a key-press
<b>B</b>	The motor operator representing a mouse-button press or release
<b>P</b>	The task of pointing (moving some pointer to a target)
<b>H</b>	Homing or the task of switching hand between mouse and keyboard
<b>D</b>	Drawing a line using mouse ( <b>not used much nowadays</b> )

# Mental Operator

- Unlike physical operators, the core thinking process is represented by a single operator **M**, known as the “mental operator”
- Any decision-making (thinking) process is modeled by **M**

# System Response Operator

- KLM originally defined an operator **R**, to model the system response time (e.g., the time between a key press and appearance of the corresponding character on the screen)
- When the KLM model was first proposed (1980), **R** was significant. However, it is no longer used since we are accustomed to almost instantaneous system response, unless we are dealing with some networked system where network delay may be an issue

# KLM- GOMS

- **Calculates the task execution time using pre-established keystroke-level primitive operators** (each operator in KLM refers to an elementary cognitive activity that takes a pre-determined amount of time to perform).
- **Seven Types of Operators:**
  - K: to press a key or a button
  - B: to click (press or release) a mouse button
  - P: to point with a mouse to a target on a display
  - H: to home hands on keyboard or other device
  - D: to draw a line segment on a grid
  - M: to mentally prepare to do an action or closely related series of primitive actions.
  - R: to symbolize the system response time during which the user has to wait for the system.
- **Each of these operators has an estimate time or simple approximation function.**
  - Time to execute is empirically defined:
    - $T_{execute} = T_K + T_B + T_P + T_H + T_D + T_M + T_R$
- **Heuristics for adding (handling) M**

# KLM- GOMS: Operator Times

Operator	Description	Time (sec)
<b>K</b>	press a key or button (shift or control key count separately) best typist (135 wpm) good typist (90 wpm) average typist (55 wpm) average typist (40 wpm) typing complex codes	.08 .12 .22 .28 .75
<b>B</b>	click (press or release) a mouse button	.10/.20
<b>P</b>	point with mouse to target on display (Fitts's Law)	1.10
<b>H</b>	home hand on keyboard or device	.40
<b>D(n,l)</b>	draw $n$ straight-line segments of total length $l$ cm (calculated for a square .56 cm grid)	$.9n + .16l$
<b>M</b>	mentally prepare/respond	1.35

# KLM: Additional Operator Times

Operator	Description	Time (sec)
	Move eyes to location on screen	2.3
	Retrieve item from memory	12
	Select among methods	12

# KLM- GOMS: An Example

- **Closing a Window**
  - Either use the close button, or press Ctrl+W

GOAL: ICONISE-WINDOW

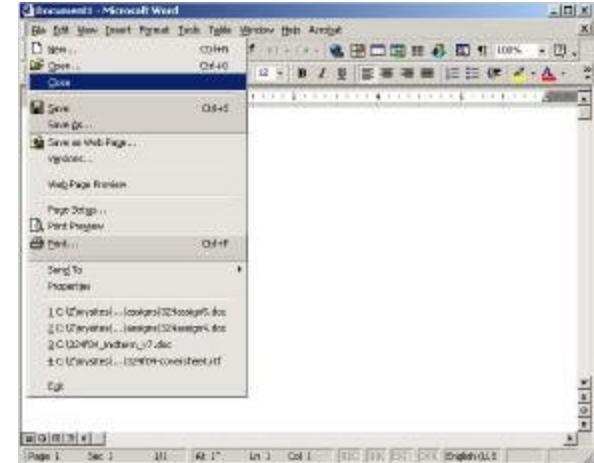
[select

    GOAL: USE-CLOSE-METHOD

- MOVE-.MOUSE-TO- FILE-MENU
- PULL-DOWN-FILE-MENU
- CLICK-OVER-CLOSE-OPTION

    GOAL: USE-CTRL-W-METHOD

        PRESS-CONTROL-W-KEY ]

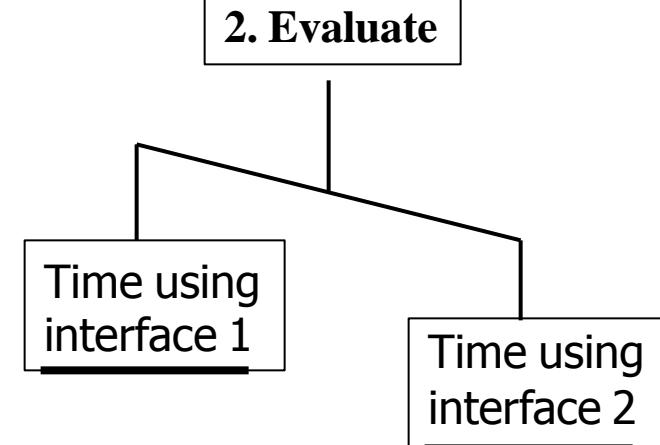


- Comparing both techniques (assuming hand starts on mouse)

## 1. Predict

	USE-CTRL-W-METHOD	USE-CLOSE-METHOD	
H[to kbd]	0.40	P[to menu]	1.1
M	1.35	B[LEFT down]	0.1
K[ctrlW key]	0.28	M	1.35
		P[to option]	1.1
		B[LEFT up]	0.1
Total	2.03 s	Total	3.75 s

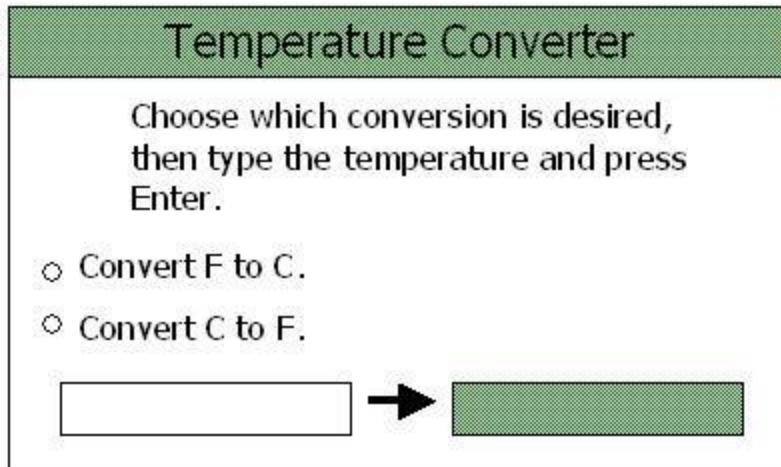
## 2. Evaluate



# KLM- GOMS: Handling M

- **Rule 0: initial insertion of candidate's M's**
  - Insert M before K
  - Insert M before P iff P selects a command
- **Rule 1: deletion of anticipated M's**
  - If an operator following an M is fully anticipated, delete that M
- **Rule 2: deletion of M's within cognitive units**
  - If a string of MK's belongs to a cognitive unit, delete all Ms but the first
- **Rule 3: deletion of M's before consecutive terminators**
  - If a K is a redundant delimiter, delete the M before it.
- **Rule 4: deletion of M's that are terminator of commands**
  - If K is a delimiter that follows a constant string, delete the M in front of it.
- **Rule 5: deletion of overlapped M's**
  - Don't count any M that overlaps an R

# KLM: Handling M – An Example



---

K	0.2
B	.10/.20
P	1.1
H	0.4
D	-
M	1.35
R	-

---

HPBHKKKKK

Apply Rule 0

HMPMBHMKMKMKMKMKM

Apply Rules 1 and 2

HMPBHMKKKKMK

Convert to numbers

$$.4+1.35+1.1+.20+.4+1.35+4(.2)+1.35+.2$$

$$=7.15$$

# KLM: Handling M – An Example

To convert temperatures,  
Type in the numeric temperature,  
Followed by °C for Celicious or  
°F for Fahrenheit. The converted  
Temperature will be displayed.

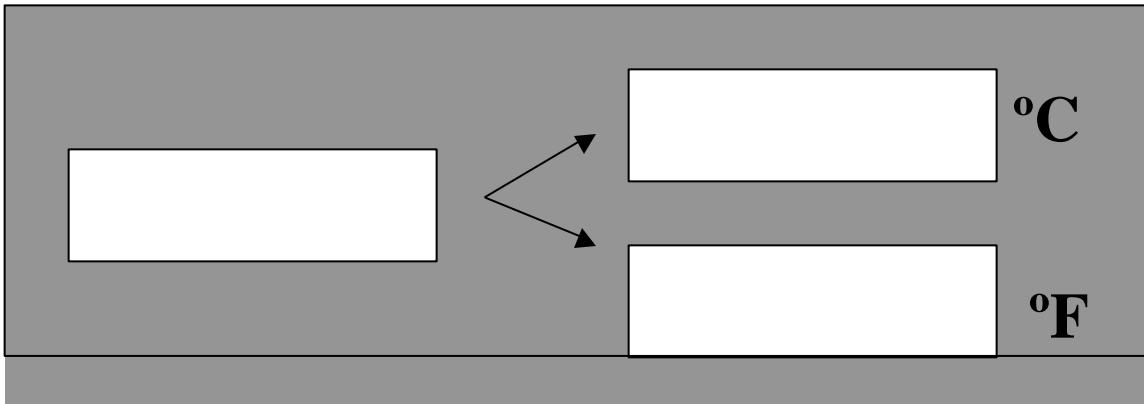
---

K	0.2
B	.10/.20
P	1.1
H	0.4
D	-
M	1.35
R	-

---

MKKKKMK = 3.7 sec

# KLM: Handling M – An Example



---

K	0.2
B	.10/.20
P	1.1
H	0.4
D	-
M	1.35
R	-

---

MKKKK = 2.15 sec

# KLM Limitations

- Although KLM provides an easy-to-understand-and-apply predictive tool for interactive system design, it has few significant constraints and limitations
  - It can model only “expert” user behavior
  - User errors can not be modeled
  - Analysis should be done for “representative” tasks; otherwise, the prediction will not be of much use in design. Finding “representative” tasks is not easy

# Learning Objective

- In the previous slides, we discussed the KLM
- In KLM, we list the basic (cognitive) steps or operators required to carry out a complex interaction task
  - The listing of operators implies a linear and sequential cognitive behavior
- In this lecture, we shall discuss another model in the GOMS family, referred to as the (CMN)GOMS model
  - CMN stands for **C**ard, **MN**ewell (the surname of the three researchers who proposed it)

# KLM vs (CMN)GOMS

- In (CMN)GOMS, a hierarchical cognitive (thought) process is assumed, as opposed to the linear thought process of KLM
- Both assumes error-free and ‘logical’ behavior
  - A logical behavior implies that we think logically, rather than driven by emotions

# (CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the task and user actions in terms of four constructs (goals, operators, methods, selection rules)
  - **Goals:** represents what the user wants to achieve, at a higher cognitive level. This is a way to structure a task from cognitive point of view
  - The notion of Goal allows us to model a cognitive process hierarchically

# (CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the task and user actions in terms of four constructs (goals, operators, methods, selection rules)
  - **Operators:** elementary acts that change user's mental (cognitive) state or task environment. This is similar to the operators we have encountered in KLM, but here the concept is more general

# (CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the task and user actions in terms of four constructs (goals, operators, methods, selection rules)
  - **Methods:** these are sets of goal-operator sequences to accomplish a sub-goal

# (CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the task and user actions in terms of four constructs (Goals, Operators, Methods, Selection Rules)
  - **Selection Rules:** sometimes there can be more than one method to accomplish a goal. Selection rules provide a mechanism to decide among the methods in a particular context of interaction

# Operator in (CMN)GOMS

- As mentioned earlier, the operators in (CMN)GOMS Model are conceptually similar to the operators in KLM
- The major difference is that in KLM, only seven operators are defined. In (CMN)GOMS, the notion of operators is not restricted to those seven operators
  - The modeler has the freedom to define any “elementary” cognitive operation and use it as operator
- The operator can be defined
  - At the keystroke level (as in KLM)
  - At higher levels (for example, the entire cognitive process involved in “closing a file by selecting the close menu option” can be defined as operator)

# Operator in (CMN)GOMS

- (CMN)GOMS gives the flexibility of defining operators at any level of cognition and different parts of the model can have operators defined at various levels
- **Example:** Suppose we want to find out the definition of a word from an online dictionary. How can we model this task with (CMN)GOMS?
- **Answer:** We shall list the goals (high level tasks) first
  - Goal: Access online dictionary (first, we need to access the dictionary)
  - Goal: Lookup definition (then, we have to find out the definition)

# Answer to the Example (Contd...)

- Next, we have to determine the methods (operator or goal-operator sequence) to achieve each of these goals
  - Goal: Access online dictionary
    - # Operator: Type URL sequence
    - # Operator: Press Enter
- Next, we have to determine the methods (operator or goal-operator sequence) to achieve each of these goals
  - Goal: Lookup definition
    - # Operator: Type word in entry field
    - # Goal: Submit the word
      - Operator: Move cursor from field to Lookup button
      - Operator: Select Lookup
    - # Operator: Read output

# Answer to the Example (Contd...)

- Thus, the complete model for the task is
  - Goal: Access online dictionary
    - Operator: Type URL sequence
    - Operator: Press Enter
  - Goal: Lookup definition
    - Operator: Type word in entry field
    - Goal: Submit the word
      - Operator: Move cursor from field to Lookup button
      - Operator: Select Lookup button
    - Operator: Read output

# Answer to the Example (Contd...)

- Notice that there is a hierarchical nature of the model
- Note that there are operators of use
  - The operator “type URL sequence” is a high-level operator defined by the modeler
  - “Press Enter” is a keystroke level operator
- Note that how both the low-level and high-level operators co-exist in the same model
- Note that there are methods of use
  - For the first goal, the method consisted of two operators
  - For the second goal, the method consisted of two operators and a sub-goal (which has a two-operators method for itself)

# Another Example

- The previous example illustrates the concepts of goals and goal hierarchy, operators and methods
- The other important concept in (CMN)GOMS is the selection rules
- **Example:** Suppose we have a window interface that can be closed in either of the two methods: by selecting the ‘close’ option from the file menu or by selecting the Ctrl key and the F4 key together. How we can model the task of “closing the window” for this system?

# Another Example

- Here, we have the high level goal of “close window” which can be achieved with either of the two methods: “use menu option” and “use Ctrl+F4 keys”
  - This is unlike the previous example where we had only one method for each goal
- We use the “Select” construct to model such situations (please see the next slide)

# Another Example

Goal: Close window

- [Select

Goal: Use menu method  
Operator: Move mouse to file menu

Operator: Pull down file menu

Operator: Click over close option

Goal: Use Ctrl+F4 method

Operator: Press Ctrl and F4 keys together]

# Another Example

- The select construct implies that “selection rules” are there to find a method among the alternatives for a particular usage context
- Example selection rules for the window closing task can be
  - Rule 1: Select “use menu method” unless another rule applies
  - Rule 2: If the application is GAME, then select “use Ctrl+F4 method”
- The rules state that, if the window appears as an interface for a game application, it should be closed using the Ctrl+F4 keys. Otherwise, it should be closed using the close menu option

# Steps for Model Construction

- A (CMN)GOMS model for a task is constructed according to the following steps:
  - Determine high-level user goals
  - Write method and selection rules (if any) for accomplishing goals
  - This may invoke sub-goals, write methods for sub-goals
  - This is recursive. Stop when operators are reached

# Use of the Model

- Like KLM, (CMN)GOMS model also makes quantitative prediction about user performance
  - By adding up the operator times, total task execution time can be computed
- However, if the modeler uses operators other than those in KLM, the modeler has to determine the operator times

# Use of the Model

- The task completion time can be used to compare the performance of competing designs
- In addition to the task completion times, the task hierarchy itself can be used for comparison
  - The deeper the hierarchy (keeping the operators same), the more complex the interface is (since it involves more thinking to operate the interface)

# (CMN) GOMS Model Limitations

- Like KLM, (CMN)GOMS also models only skilled (expert) user behavior
  - That means user does not make any errors
- Can not capture the full complexity of human cognition such as learning effect, parallel cognitive activities and emotional behavior

# HCI: Model-based Design (Individual Models of Human Factors)

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Learning Objective

- In the previous lectures, we discussed the two popular models belonging to the GOMS family models, namely: KLM and (CMN)GOMS
  - Those models, as we mentioned before, are simple models of human information processing
- These models are one of three cognitive modeling approaches used in the HCI System Design

# Learning Objective

- A second type of cognitive models used in HCI is the individual models of human factors
- To recap, these are the models of human factors such as the motor movement, choice-reaction, eye movement etc.
  - The models provide analytical expressions to compute the values associated with the corresponding factors, such as movement time, movement effort etc.

# Learning Objective

- In this lecture, let us discuss two well known models belonging to this category
  - **Fitts' Law:** This law governs the manual (motor) movement
  - **Hick-Hyman Law:** This law governs the decision making process in the presence of choice

# Fitts' Law

- It is one of the earliest predictive models used in HCI (the most well known models in HCI also)
- Proposed by P M Fitts (hence the name) in 1954

Fitts, P. M. (1954), "The information capacity of the human motor system in controlling the amplitude of movement", *Journal of Experimental Psychology*, 47, 381-391.

# Fitts' Law

- As we noted before, the Fitts' law is a model of human motor performance
  - It mainly models the way we move our hand and fingers
- A very important thing to note that this law is not generalized one; it models the motor performance under certain constraints (see the next slide)

# Fitts' Law - Characteristics

- This law models the human motor performance with the following characteristics
  - The movement is related to some “*target acquisition task*” (i.e., the human wants to acquire some target at some distance from the current hand/finger position)
  - The movement is *rapid* and *aimed* (i.e., no decision making is involved during the movement)
  - The movement is *error-free* (i.e. the target is acquired at the very first attempt)

# Nature of the Fitts' Law

- Another important thing about the Fitts' law is that, it is both a descriptive and a predictive model
- Why it is a descriptive model?
  - Because it provides “throughput”, which is a descriptive measure of human motor performance
- Why it is a predictive model?
  - Because it provides a prediction equation (an analytical expression) for the time to acquire a target, given the distance and size of the target

# Task Difficulty

- The key concern in this Fitts' law is to measure “task difficulty” (i.e., how difficult it is for a person to acquire, with his hand/finger, a target at a distance 'D' from the hand/finger's current position)
  - Note that the movement is assumed to be rapid, aimed towards the target and error-free

# Task Difficulty

- Fitts, in his experiments, noted that the difficulty of a target acquisition task is related to two factors
  - Distance (D): the distance by which the person needs to move his hand/finger. This is also called *amplitude* (A) of the movement
  - The larger the 'D' is, the harder the task becomes
  - Width (W): the difficulty also depends on the width of the target to be acquired by the person
  - As the width increases, then the task becomes easier

# Measuring Task Difficulty

- The qualitative description of the relationships between the task difficulty and the target distance (D) and width (W) can not help in “measuring” how difficult a task is
- Fitts’ proposed a ‘concrete’ measure of task difficulty, called the “index of difficulty” (ID)

# Measuring Task Difficulty

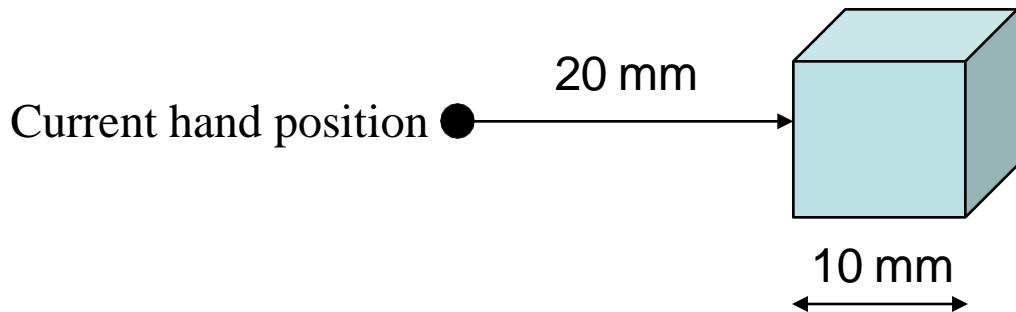
- From the analysis of empirical data, Fitts' law proposed the following relationship between three factors, namely: ID, D and W

$$ID = \log_2(D/W+1) \text{ [unit is } bits\text{]}$$

(Note: the above formula was not what Fitts originally proposed. It is a refinement of the original formulation over time. Since this is the most common formulation of ID, we shall follow this rather than the original one)

# ID - Example

- Suppose a person wants to grab a small cubic block of wood (side length of 10 mm) at a distance of 20 mm. What is the difficulty for this task?



- Here  $D = 20 \text{ mm}$ ,  $W = 10 \text{ mm}$
- Thus,  $\text{ID} = \log_2(20/10+1)$   
=  $\log_2(2+1)$   
=  $\log_2 3 = 1.57 \text{ bits}$

# Throughput

- Fitts' also proposed a measure called the *index of performance* (IP), now called *throughput* (TP)
  - Computed as the difficulty of a task (ID, in bits) divided by the movement time (MT, in sec) to complete the task
- Thus,  $TP = ID/MT$  bits/sec (bps)

# Throughput - Example

- Consider our previous example (on ID). If the person takes 2 sec to reach for the block, what is the throughput of the person for the task

Here ID = 1.57 bits, MT = 2 sec

Thus TP =  $1.57/2$

= 0.785 bits/sec (bps)

# Implication of Throughput

- The concept of throughput is very important
- It refers to a measure of performance for rapid, aimed, error-free target acquisition task (as implied by its original name “The Index of Performance”)
  - Taking the human motor behavior into account
- In other words, throughput should be relatively constant for a test condition over a wide range of task difficulties; i.e., over a wide range of target distances and target widths

# Examples of a Test Condition

- Suppose a user is trying to point to an icon on the screen using a mouse
  - The task can be mapped to a rapid, aimed, error-free target acquisition task
  - The mouse is the test condition here
- If the user is trying to point with a touchpad, then touchpad is the test condition
- Suppose we try to determine the target acquisition performance for a group of persons (say, workers in a factory) after the lunch
  - The “taking of lunch” is the test condition here

# Throughput – Design Implication

- The central idea is - Throughput provides a means to measure the user performance for a test condition
  - We can use this idea in design
- We collect throughput data from a set of users for different task difficulties
  - The mean throughput for all users over all task difficulties represents the average user performance for a test condition

# Throughput – Design Implication

- Example— suppose we want to measure the performance of a mouse. We employ 10 participants in an experiment and gave them 6 different target acquisition tasks (where the task difficulties varied). From the data collected, we can measure the mouse performance by taking the mean throughput over all participants and the tasks (next slide)

# Throughput – Design Implication

D	W	ID (bits)	MT (sec)	TP (bits/sec)
8	8	1.00	0.576	1.74
16	8	1.58	0.694	2.28
16	2	3.17	1.104	2.87
32	2	4.09	1.392	2.94
32	1	5.04	1.711	2.95
64	1	6.02	2.295	2.62
<b>Mean</b>				<b>2.57</b>

The 6 tasks with varying difficulty levels

Throughput = 2.57 bits/sec (bps)

Each value indicates mean of 10 participants

# Throughput – Design Implication

- In the example, note that the mean throughputs for each task difficulty is relatively constant (i.e., not varying widely)
  - This is one way of checking the correctness of our procedure (i.e., whether the data collection and analysis was proper or not)

# Summary

- In this lecture, we discussed the concept of throughput and how to measure it
- In the next topic, let us discuss more design implications of throughput
- Let us discuss the predictive nature of the Fitts' law
- And, finally let us discuss the Hick-Hyman law

# Learning Objective

- So far, we got introduced to the Fitts' law
  - The Fitts' law models the human motor behavior for rapid, aimed, error-free target acquisition task
- The law allows us to measure the task difficulty using the index of difficulty (ID)

# Learning Objective

- Using ID and task completion time (MT), we can compute the throughput (TP), which is a measure of task performance

$$TP = ID/MT$$

Unit of ID is bits, unit of MT is sec.

Thus, unit of TP is bits/sec (bps)

# Learning Objective

- We saw how TP helps in design
  - We can estimate the user performance under a test condition by estimating the TP
  - The TP is estimated by taking the mean of the TP achieved by different persons tested with varying task difficulty levels under the same test condition

# Learning Objective

- In this lecture, we shall extend this knowledge further and learn about the following
  - How TP can help in comparing designs?
  - How the Fitts' law can be used as a predictive model?
- Also, let us study the Hick-Hyman law, another model of human factor (models the choice-reaction time)

# Throughput – Design Implication

- In the case of Fitts' law, we discussed one design implication of throughput in HCI
  - That is, to estimate user's motor performance in a given test condition
- We can extend this idea further to compare the performance of competing designs

# Throughput – Design Implication

- Suppose you have designed two input devices: a mouse and a touchpad. You want to determine which of the two is better in terms of user performance, when used to acquire targets (e.g., for point and select tasks). How can you do so?

# Throughput – Design Implication

- You are asked to set up two experiments for two test conditions: one with the mouse and the other with the touchpad
- Determine throughput for each test condition as we have already done before (i.e., collect throughput data from a group of users for a set of tasks with varying difficulty level and take the overall mean)

# Throughput – Design Implication

- Suppose we got the throughputs  $TP_1$  and  $TP_2$  for the mouse and the touchpad experiments, respectively
- You are asked to Compare  $TP_1$  and  $TP_2$ 
  - If  $TP_1 > TP_2$ , the mouse gives better performance
  - The touchpad is better if  $TP_1 < TP_2$
  - Performance-wise they are the same if  $TP_1 = TP_2$  (this is very unlikely as we are most likely to observe some difference)

# Predictive Nature of Fitts' Law

- The throughput measure, derived from the Fitts' law, is found to be descriptive
  - We need to determine its value empirically
- Fitts' law also allows us to predict the performance
  - We can “compute” the performance rather than determining its value empirically

# Predictive Nature of Fitts' Law

- Although not originally proposed by Fitts, it is now common practice to build a prediction equation (analytical expression) in the Fitts' law research
- The predictive equation is obtained by a method known as Linear Regression of MT (Movement Time) against the ID (Index of Difficulty) in a MT-ID plot
- The equation is of the form  $MT = a + b.ID$

Where, 'a' and 'b' are constants for a test condition (empirically derived)

# Predictive Nature of Fitts' Law

- As we can see, the equation allows us to predict the time to complete a target acquisition task (with known 'D' and 'W')
- How we can use predictive equation in the system design?
  - Find the constants ('a' and 'b') empirically, for a test condition
  - Use these constant values in the predictive equation to determine the MT for a representative target acquisition task under the test condition
  - Compare the values of MTs for different test conditions to decide (as with throughput)

# Speed-Accuracy Trade-off

- Suppose, we try to select an icon by clicking on it. The icon width is 'D'
  - Suppose each click is called a “hit”. In a trial involving several hits, we are most likely to observe that not all hits lie within 'D' (some may be just outside)
  - If we plot the *hit distributions* (i.e., the coordinates of the hits), we shall see that about 4% of the hits are outside the target boundary

# Speed-Accuracy Trade-off

- This is called the speed-accuracy trade-off
  - When we try to make rapid movements, then we can not avoid the mistakes (errors)
- However, in the measures (ID, TP and MT), we have used 'D' only, without taking into account the trade-off
  - We assumed all hits will be inside the target boundary

# Speed-Accuracy Trade-off

- We can resolve this in two-ways
  - Either we proceed with our current approach, with the knowledge that the measures will have 4% error rates
  - Or we take the effective width  $D_e$  (the width of the region enclosing all the hits) instead of  $D$
- The second approach requires us to empirically determine  $D_e$  for each test condition

# The Hick-Hyman Law

- While Fitts' law relates task performance to motor behavior, there is another law popularly used in HCI, which tell us the “reaction time” (i.e., the time to react to a stimulus) of a person in the presence of “choices”
- The law is called the Hick-Hyman law, named after its inventors

# Example

- A telephone call operator has 10 buttons. When the light behind one of the buttons comes on, the operator must push the button and answer the call
  - When a light comes on, how long does the operator takes to decide which button to press?
- In the example,
  - The “light on” is the stimulus
  - We are interested to know the operator’s “reaction time” in the presence of the stimulus
  - The operator has to decide among these 10 buttons (further these buttons represent the set of choices)

# The Hick-Hyman Law

- As we discussed in the example (previous slides), the Hick-Hyman law can be used to predict the reaction times in such situations
- Thus, this law models the human's reaction time (also known as the *choice-reaction time*) under *uncertainty conditions* (the presence of choices)
  - The law states that the reaction (decision) time 'T' increases with uncertainty about the judgment or decision to be made

# The Law

- We know that a measure of uncertainty is referred to as the entropy ( $H$ )  
Thus,  $T \propto H$  (or equivalently),  $T = kH$ , where ' $k$ ' is the proportionality constant (empirically determined)
- We can calculate ' $H$ ' in terms of the choices in the following way  
let,  $p_i$  be the probability of making the  $i^{\text{th}}$  choice  
Then, 
$$H = \sum_{i=1} p_i \log_2(1/p_i)$$

# The Law

- Therefore,

$$T = k \sum_{i=1} p_i \log_2(1/p_i)$$

- When all the probabilities of making the choices becomes equal, we have  $H = \log_2 N$  ( $N = \text{no of choices}$ )
  - In such cases,  $T = k \log_2 N$

# Example Revisited

- Then, what will be the operator's reaction time in our example?
  - Here  $N = 10$
  - A button can be selected with a probability of  $1/10$  and all probabilities are equal (equally likely)
  - Thus,  $T = k \log_2 10$   
 $= 0.66 \text{ ms}$  (assuming  $a = 0, b = 0.2$ )

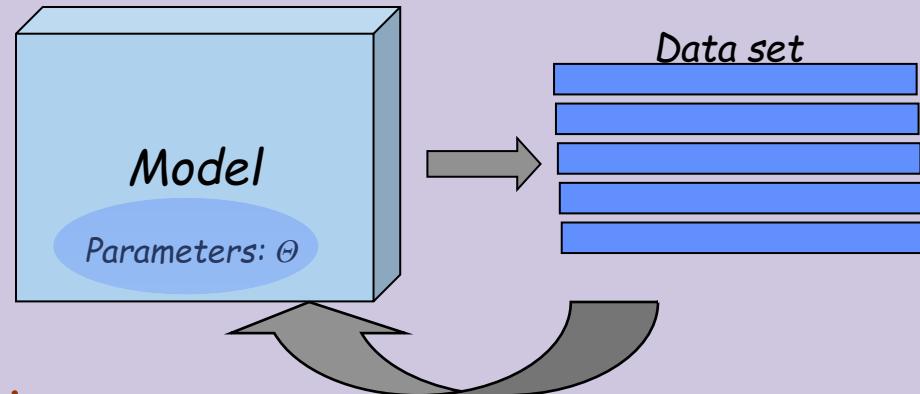
# An Introduction to Markov Chain Monte Carlo

# The Problem

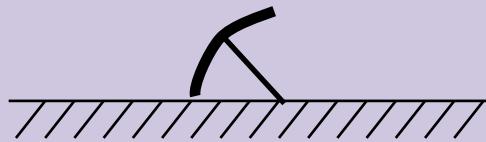
- ◆ To solve search problems (search for optima of functions, search for NN structures, search for solution to various problems)

# Statistical Parameter Estimation

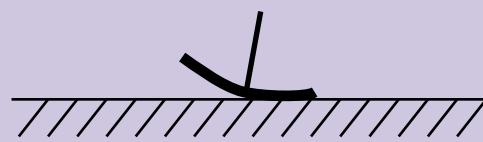
- The Basic Paradigm:



- MLE / Bayesian Approach
- Input Data: Series of observations  $X_1, X_2 \dots X_t$ 
  - We assumed observations were i.i.d (independent identical distributed)



Heads -  $P(H)$

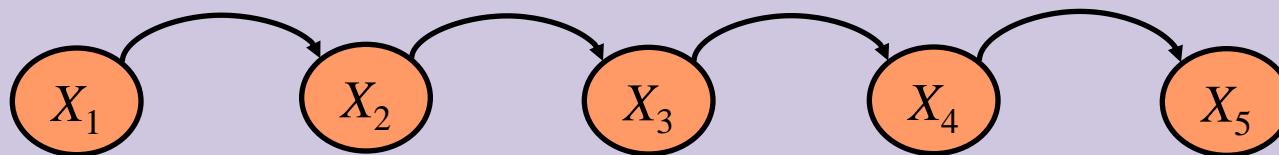


Tails -  $1-P(H)$

# Markov Process

- **Markov Property:** The state of the system at time  $t+1$  depends only on the state of the system at time  $t$

$$\Pr[X_{t+1} = x_{t+1} / X_1 \Lambda X_t = x_1 \Lambda x_t] = \Pr[X_{t+1} = x_{t+1} / X_t = x_t]$$



- **Stationary Assumption:** Transition probabilities are independent of time ( $t$ )

$$\Pr[X_{t+1} = b / X_t = a] = p_{ab}$$

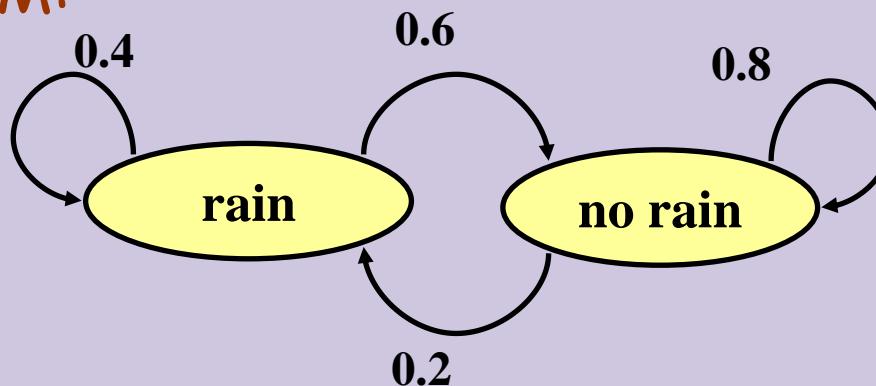
Bounded memory transition model

# Markov Process: Simple Example

Weather:

- raining today
  - 40% rain tomorrow
  - 60% no rain tomorrow
- not raining today
  - 20% rain tomorrow
  - 80% no rain tomorrow

Stochastic FSM:



# Markov Process: Simple Example

Weather:

- raining today  40% rain tomorrow  
 60% no rain tomorrow
- not raining today  20% rain tomorrow  
 80% no rain tomorrow

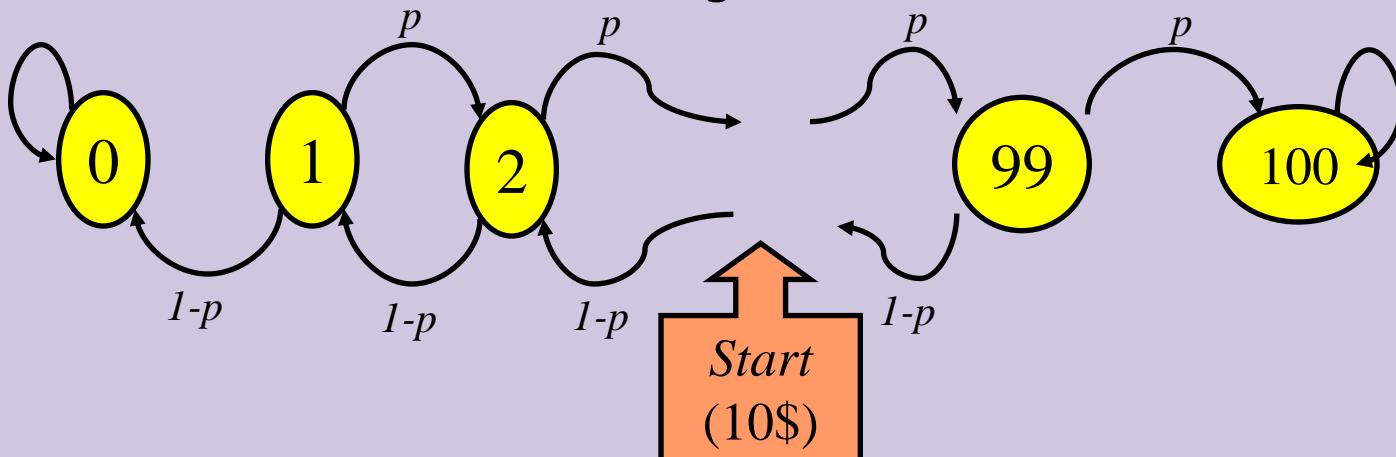
The transition matrix:

$$P = \begin{pmatrix} 0.4 & 0.6 \\ 0.2 & 0.8 \end{pmatrix}$$

- Stochastic matrix:  
Rows sum up to 1
- Double stochastic matrix:  
Rows and columns sum up to 1

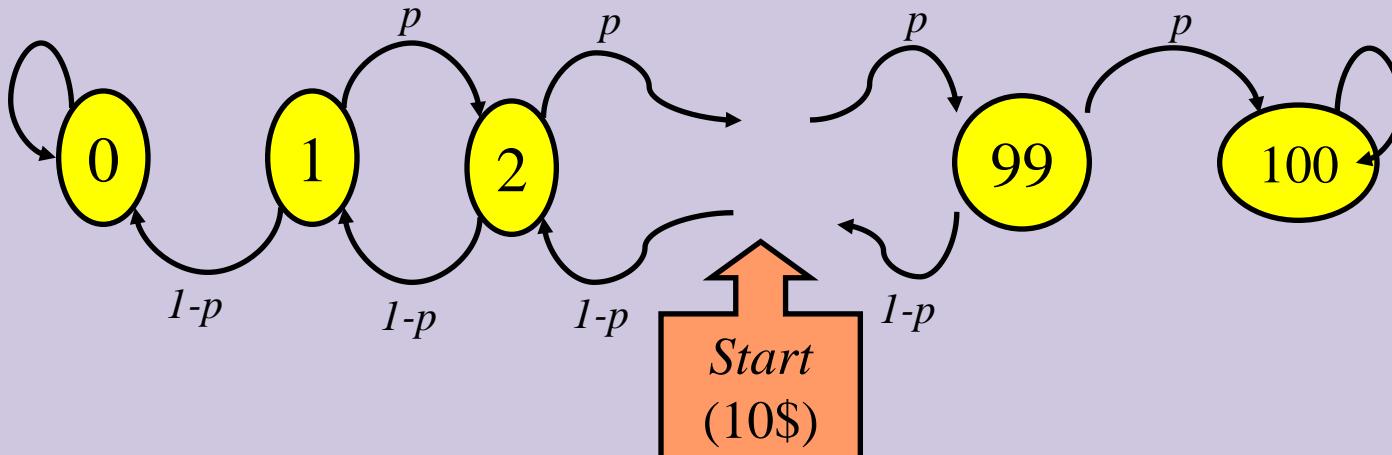
# Markov Process: Gambler's Example

- Gambler starts with \$10
- At each play we have one of the following:
  - Gambler wins \$1 with probability  $p$
  - Gambler loses \$1 with probability  $1-p$
- Game ends when gambler goes broke, or gains a fortune of \$100  
(Both 0 and 100 are absorbing states)



# Markov Process

- **Markov Process** - It is described by a Stochastic FSM
- **Markov Chain** - A Random Walk on this Graph
  - (distribution over paths)
- Edge-weights give us  $\Pr[X_{t+1} = b / X_t = a] = p_{ab}$
- We can ask more complex questions, like  $\Pr[X_{t+2} = a / X_t = b] = ?$

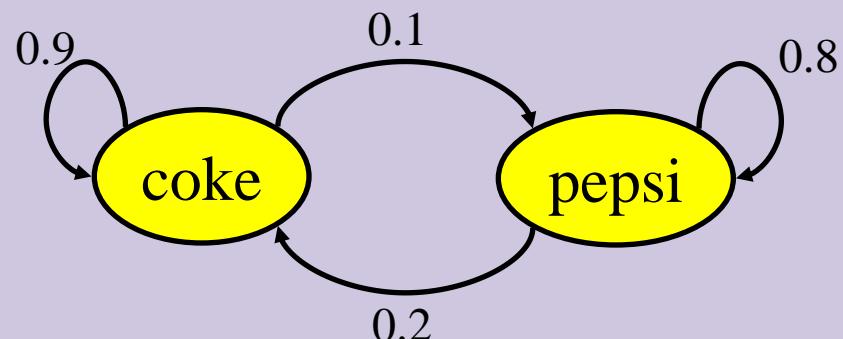


# Markov Process: Coke vs. Pepsi Example

- Given that a person's last cola purchase was **Coke**, there is a **90%** chance that his next cola purchase will also be **Coke**.
- If a person's last cola purchase was **Pepsi**, there is an **80%** chance that his next cola purchase will also be **Pepsi**.

transition matrix:

$$P = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$$



# Markov Process: Coke vs. Pepsi Example

Given that a person is currently a **Pepsi** purchaser, what is the probability that he will purchase **Coke** two purchases from now?

$$\Pr[\text{Pepsi} \rightarrow ? \rightarrow \text{Coke}] =$$

$$\Pr[\text{Pepsi} \rightarrow \text{Coke} \rightarrow \text{Coke}] + \Pr[\text{Pepsi} \rightarrow \text{Pepsi} \rightarrow \text{Coke}] =$$

$$0.2 * 0.9 + 0.8 * 0.2 = 0.34$$

$$P = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} \begin{bmatrix} 0.9 \\ 0.2 \end{bmatrix} = \begin{bmatrix} 0.83 & 0.17 \\ 0.34 & 0.66 \end{bmatrix}$$

$\uparrow$   
 $\text{Pepsi} \rightarrow ? \quad ? \rightarrow \text{Coke}$

# Markov Process: Coke vs. Pepsi Example

Given that a person is currently a **Coke** purchaser, what is the probability that he will purchase **Pepsi three** purchases from now?

$$P^3 = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} \begin{bmatrix} 0.83 & 0.17 \\ 0.34 & 0.66 \end{bmatrix} = \begin{bmatrix} 0.781 & 0.219 \\ 0.438 & 0.562 \end{bmatrix}$$

# Markov Process: Coke vs. Pepsi Example

- Assume each person makes one cola purchase per week
- Suppose 60% of all people now drink Coke, and 40% drink Pepsi
- What fraction of people will be drinking Coke three weeks from now?

$$P = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$$

$$P^3 = \begin{bmatrix} 0.781 & 0.219 \\ 0.438 & 0.562 \end{bmatrix}$$

$$\Pr[X_3 = \text{Coke}] = 0.6 * 0.781 + 0.4 * 0.438 = 0.6438$$

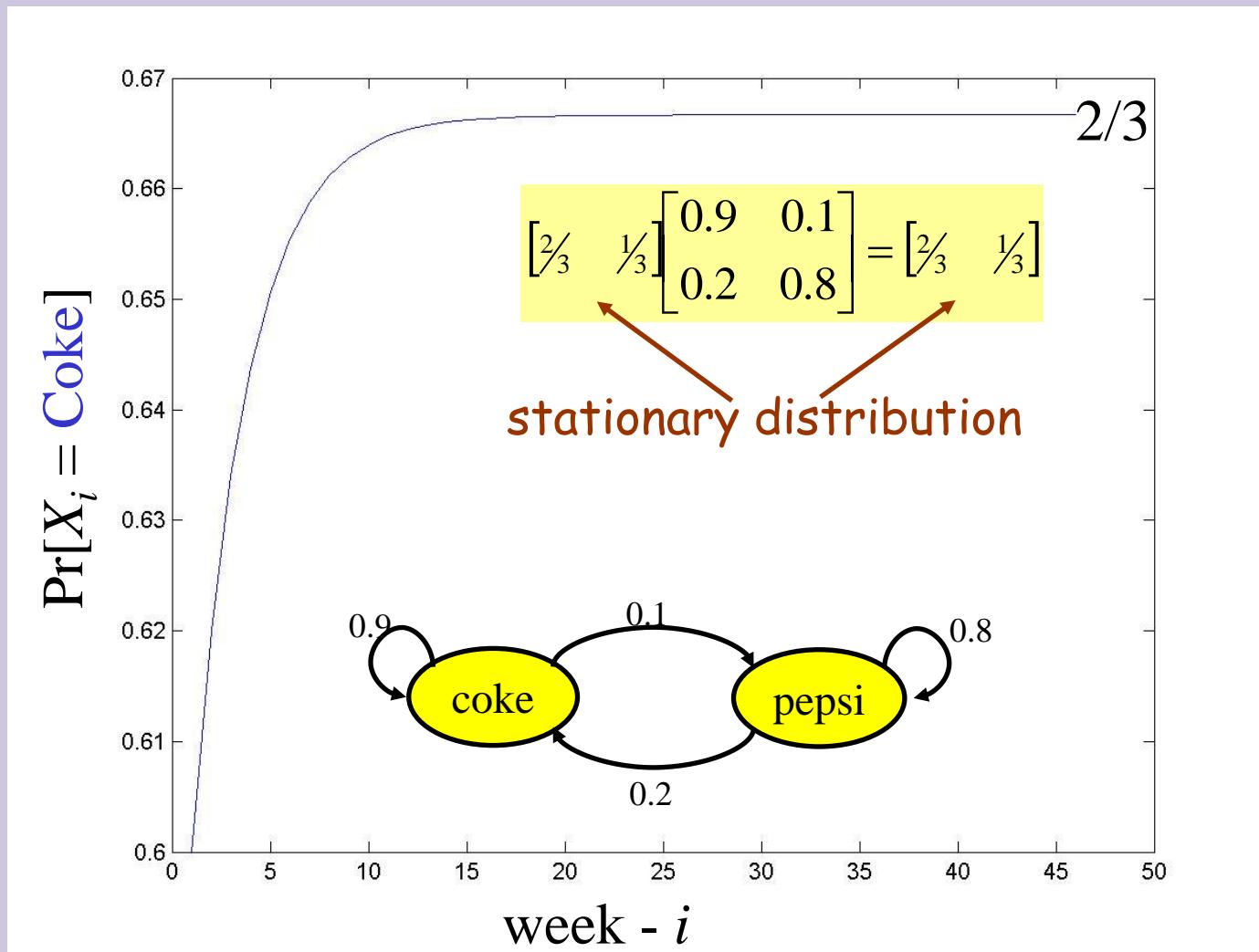
$Q_i$  - the distribution in week  $i$

$Q_0 = (0.6, 0.4)$  - initial distribution

$$Q_3 = Q_0 * P^3 = (0.6438, 0.3562)$$

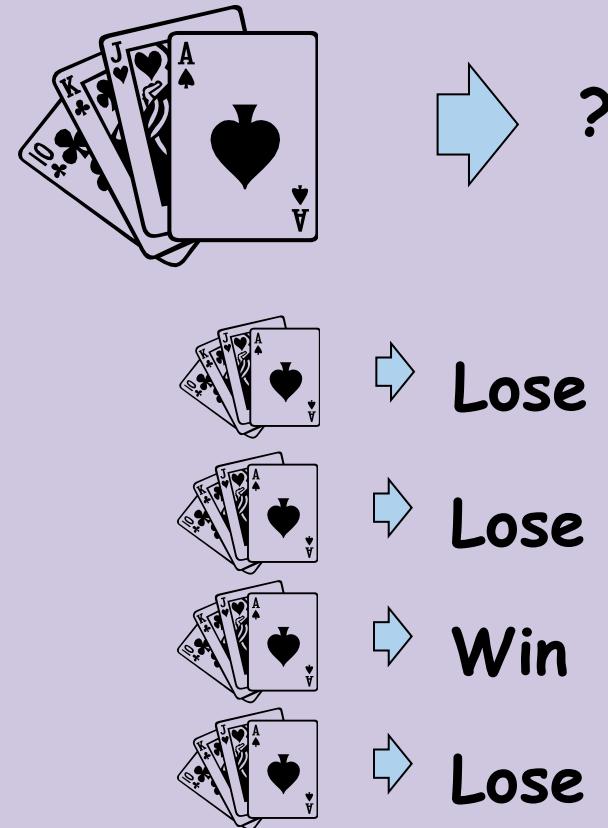
# Markov Process: Coke vs. Pepsi Example

Simulation:



# Monte Carlo Principle

- ◆ Consider the game of solitaire: what's the chance of winning with a properly shuffled deck?
- ◆ Hard to compute analytically because winning or losing depends on a complex procedure of reorganizing cards
- ◆ Insight: why not just *play a few hands*, and see empirically how many do in fact win?
- ◆ More generally, can approximate a probability density function using only samples from that density



Chance of winning is 1 in 4!

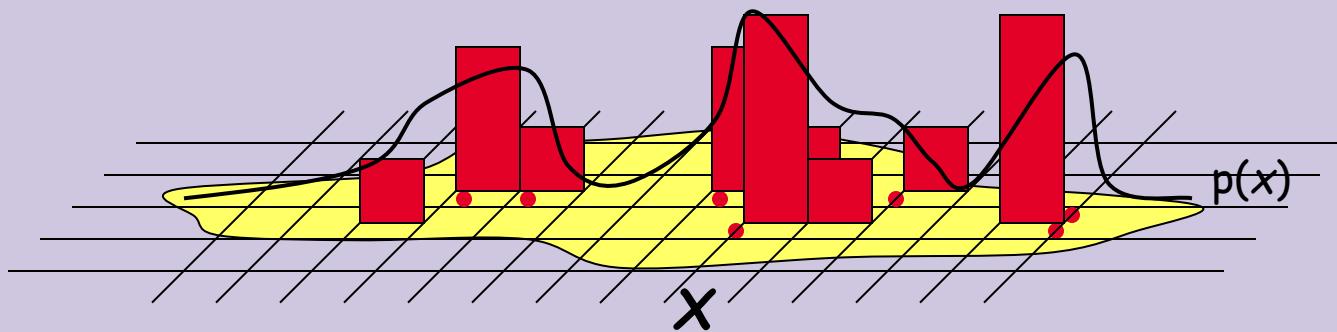
# The Monte Carlo Principle

- ◆  $p(x)$ : It is a target density defined over a high-dimensional space (ex: the space of all possible configurations of a system under study)
- ◆ The idea of Monte Carlo techniques is to draw a set of i.i.d samples  $\{x_1, \dots, x_N\}$  ( $N$  samples) from  $p(x)$  in order to approximate  $p(x)$  with the empirical distribution
- ◆ Using these samples we can approximate integrals  $I(f)$  (or very large sums) with tractable sums that converge (as the number of samples grows) to  $I(f)$

$$I(f) = \int f(x)p(x)dx \approx \frac{1}{N} \sum_{i=1}^N f(x^{(i)}) \xrightarrow[N \rightarrow \infty]{} I(f)$$

# The Monte Carlo Principle

- ◆ Given a very large set  $X$  and a distribution  $p(x)$  over it
- ◆ We draw i.i.d. a set of  $N$  samples
- ◆ We can then approximate the distribution using these samples



$$p_N(x) = \frac{1}{N} \sum_{i=1}^N \mathbf{1}(x^{(i)} = x) \xrightarrow{N \rightarrow \infty} p(x)$$

# The Monte Carlo Principle

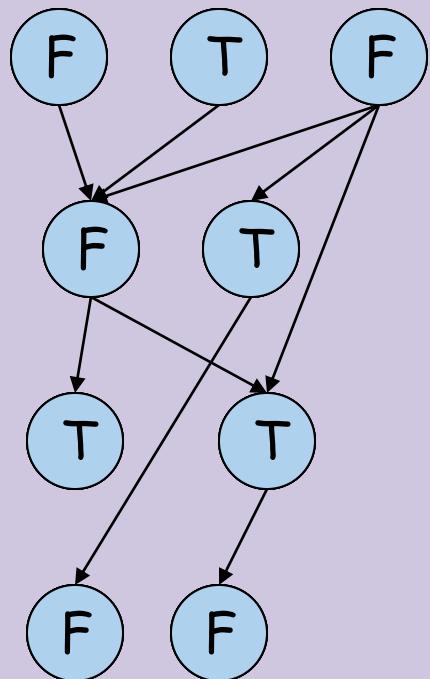
- ◆ We can also use these samples to compute expectations

$$E_N(f) = \frac{1}{N} \sum_{i=1}^N f(x^{(i)}) \xrightarrow{N \rightarrow \infty} E(f) = \sum_x f(x) p(x)$$

- ◆ And even use them to find a maximum

$$\hat{x} = \arg \max_{x^{(i)}} [p(x^{(i)})]$$

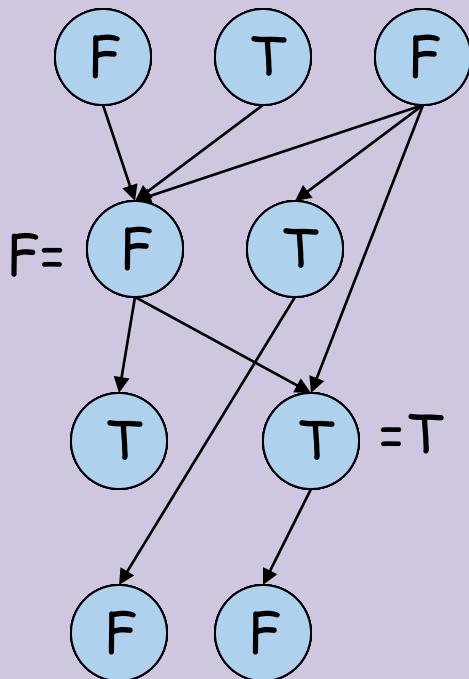
# Example: Bayes Net Inference



Sample 1: FTFTTTFFFT  
Sample 2: FTFFFTTTFF  
etc.

- ◆ Suppose we have a Bayesian network with variables  $X$
- ◆ Our state space is the set of all possible assignments of values to variables
- ◆ Computing the joint distribution is in the worst case NP-hard
- ◆ However, note that you can draw a sample in time that is linear in the size of the network
- ◆ Draw  $N$  samples, use them to approximate the joint

# Rejection Sampling



Sample 1: FTFTTTFFFT **Reject**  
Sample 2: FTFFTTTFFF **Accept**  
etc.

- ◆ Suppose we have a Bayesian network with variables  $X$
- ◆ We wish to condition on some evidence  $Z \not\in X$  and compute the posterior over  $Y = X - Z$
- ◆ Draw samples, rejecting them when they contradict the evidence in  $Z$
- ◆ Very inefficient if the evidence is itself improbable, because we must reject a large number of samples

# Rejection Sampling

- ◆ More generally, we would like to sample from  $p(x)$ , but it's easier to sample from a *proposal distribution*  $q(x)$
- ◆  $q(x)$  satisfies  $p(x) \leq M q(x)$  for some  $M < \infty$
- ◆ Procedure:
  - Sample  $x^{(i)}$  from  $q(x)$
  - Accept with probability  $p(x^{(i)}) / Mq(x^{(i)})$
  - Reject otherwise
- ◆ The accepted  $x^{(i)}$  are sampled from  $p(x)$ !
- ◆ Problem: if  $M$  is too large, we will rarely accept samples
  - In the Bayes network, if the evidence  $Z$  is very unlikely then we will reject almost all samples

# Markov Chain Monte Carlo (MCMC) Idea

- ◆ Design a Markov Chain on finite state space

state space:  $x^{(i)} \in \{x_1, x_2, \dots, x_s\}$

Markov property:  $p(x^{(i)} | x^{(i-1)}, \dots, x^{(1)}) = T(x^{(i)} | x^{(i-1)})$

...such that when simulating a trajectory of states from it; will explores the state space spending more time in the most important regions (where  $p(x)$  is large)

# Markov Chain Monte Carlo (MCMC) Idea

- ◆ Recall again the set  $X$  and the distribution  $p(x)$  we wish to sample from
- ◆ Suppose that it is hard to sample  $p(x)$  but that it is possible to “walk around” in  $X$  using only local state transitions
- ◆ Insight: we can use a “random walk” to help us draw random samples from  $p(x)$



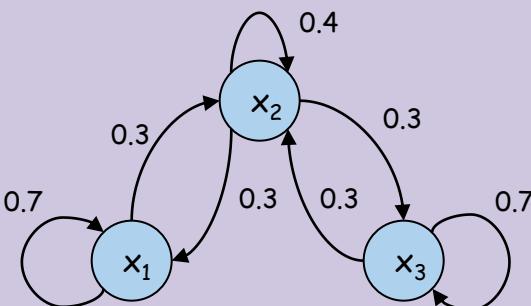
# The Markov Chains

- ◆ Markov chain (MC) on a space  $X$  with transitions  $T$  is a random process (infinite sequence of random variables)  $(x^{(0)}, x^{(1)}, \dots, x^{(t)}, \dots)$  in  $X^\infty$  that satisfy

$$p(x^{(t)} | x^{(t-1)}, \dots, x^{(1)}) = T(x^{(t-1)}, x^{(t)})$$

- ◆ That is, the probability of being in a particular state at time  $t$  given the state history depends only on the state at time  $t-1$
- ◆ If the transition probabilities are fixed for all  $t$ , the chain is considered *homogeneous*

$$T = \begin{pmatrix} 0.7 & 0.3 & 0 \\ 0.3 & 0.4 & 0.3 \\ 0 & 0.3 & 0.7 \end{pmatrix}$$



# The Markov Chains for Sampling

- ◆ In order for a Markov chain (MC) to be useful for sampling  $p(x)$ , we require that for any starting state  $x^{(1)}$

$$p_{x^{(1)}}^{(t)}(x) \xrightarrow[t \rightarrow \infty]{} p(x)$$

- ◆ Equivalently, the **stationary distribution** of the Markov chain must be  $p(x)$

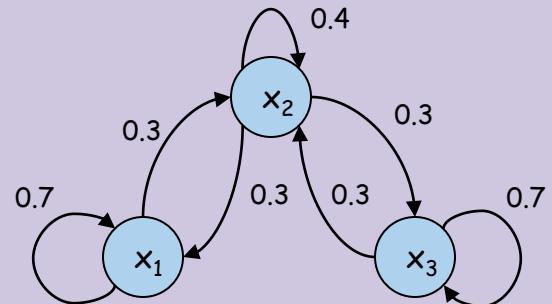
$$[pT](x) = p(x)$$

- ◆ If this is the case, we can start in an arbitrary state, use the Markov chain to do a random walk for a while, and stop and output the current state  $x^{(t)}$
- ◆ The resulting state will be sampled from  $p(x)$ !

# The Stationary Distribution

- ◆ Consider the Markov chain given above:

$$T = \begin{pmatrix} 0.7 & 0.3 & 0 \\ 0.3 & 0.4 & 0.3 \\ 0 & 0.3 & 0.7 \end{pmatrix}$$



- ◆ The stationary distribution is

$$\begin{pmatrix} 0.33 & 0.33 & 0.33 \end{pmatrix} \times \begin{pmatrix} 0.7 & 0.3 & 0 \\ 0.3 & 0.4 & 0.3 \\ 0 & 0.3 & 0.7 \end{pmatrix} = \begin{pmatrix} 0.33 & 0.33 & 0.33 \end{pmatrix}$$

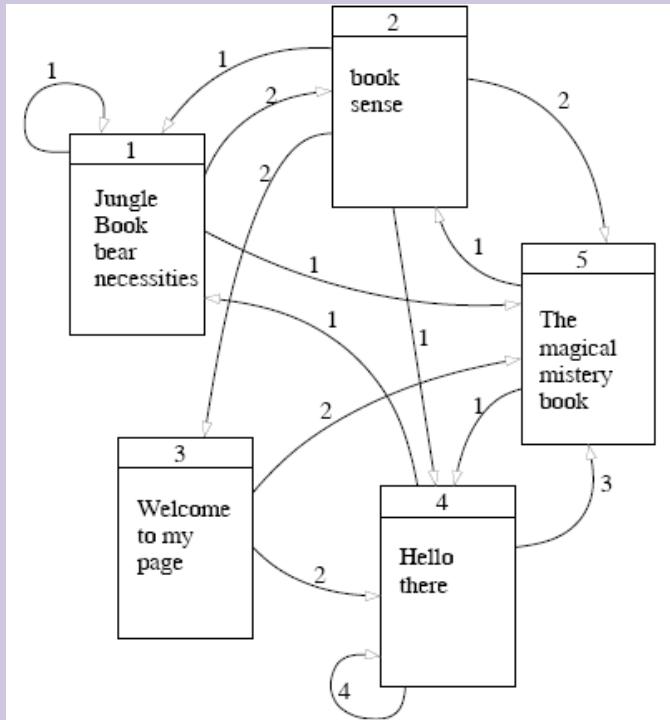
- ◆ Some samples:

1,1,2,3,2,1,2,3,3,**2**  
1,2,2,1,1,2,3,3,**3**  
1,1,1,2,3,2,2,1,1,**1**  
1,2,3,3,3,2,1,2,2,**3**  
1,1,2,2,2,3,3,2,1,**1**  
1,2,2,2,3,3,3,2,2,**2**

Empirical Distribution:

$$\begin{pmatrix} 0.33 & 0.33 & 0.33 \end{pmatrix}$$

# Stationary Distribution of a MC



- ◆ Supposing you browse this for infinitely long time, what is the probability to be at page  $x_i$ .
- ◆ No matter where you started off.  
=>Page-Rank (Google)

$$p(x^{(i)} | x^{(i-1)}, \dots, x^{(1)}) = T(x^{(i)} | x^{(i-1)}) \equiv \mathbf{T}$$

$$((\mu(x^{(1)})\mathbf{T})\mathbf{T})\dots\mathbf{T} = \mu(x^{(1)})\mathbf{T}^n = p(x), \quad s.t. \quad p(x)\mathbf{T} = p(x)$$

# Google vs. MCMC

$$p(x)\mathbf{T} = p(x)$$

- ◆ Google is given  $\mathbf{T}$  and finds  $p(x)$
- ◆ MCMC is given  $p(x)$  and finds  $\mathbf{T}$ 
  - But it also needs a ‘proposal (transition) probability distribution’ to be specified.
- ◆ Q: Do all MCs have a stationary distribution?
- ◆ A: No.

# Conditions for Existence of an Unique Stationary Distribution

## ◆ Irreducibility

- The transition graph is connected (any state can be reached)

## ◆ Aperiodicity

- State trajectories drawn from the transition don't get trapped into cycles

## ◆ MCMC samplers are irreducible and aperiodic MCs that converge to the target distribution

## ◆ These 2 conditions are not easy to impose directly

# The Ergodicity

- ◆ Claim: To ensure that the chain converges to a unique stationary distribution the following conditions are sufficient:
  - *Irreducibility*: every state is eventually reachable from any start state; for all  $x, y$  in  $\mathbf{X}$  there exists a  $t$  such that
$$p_x^{(t)}(y) > 0$$
  - *Aperiodicity*: the chain doesn't get caught in cycles; for all  $x, y$  in  $\mathbf{X}$  it is the case that
$$\gcd\{t : p_x^{(t)}(y) > 0\} = 1$$
- ◆ The process is *ergodic* if it is both irreducible and aperiodic
- ◆ This claim is easy to prove, but involves eigenstuff!

# Reversibility

- ◆ Reversibility (also called ‘detailed balance’) is a sufficient (but not necessary) condition for  $p(x)$  to be the stationary distribution.

$$p(x^{(i)})T(x^{(i-1)}|x^{(i)}) = p(x^{(i-1)})T(x^{(i)}|x^{(i-1)}).$$

Summing both sides over  $x^{(i-1)}$ , gives us

$$p(x^{(i)}) = \sum_{x^{(i-1)}} p(x^{(i-1)})T(x^{(i)}|x^{(i-1)}).$$

- ◆ It is easier to work with this condition.

# The Markov Chains for Sampling

- ◆ Claim: To ensure that the stationary distribution of the Markov chain is  $p(x)$  it is sufficient for  $p$  and  $T$  to satisfy the *detailed balance (reversibility)* condition:

$$p(x)T(x, y) = p(y)T(y, x)$$

- ◆ Proof: for all  $y$  we have

$$[p T](y) = \sum_x p(x)T(x, y) = \sum_x p(y)T(y, x) = p(y) \sum_x T(y, x) = p(y)$$

- ◆ And thus  $p$  must be a stationary distribution of  $T$

# The MCMC Algorithms

- ◆ Metropolis-Hastings algorithm
- ◆ Metropolis algorithm
  - Mixtures and blocks
- ◆ Gibbs sampling
- ◆ Sequential Monte Carlo & Particle Filters

# The Metropolis-Hastings and the Metropolis Algorithm as a Special Case

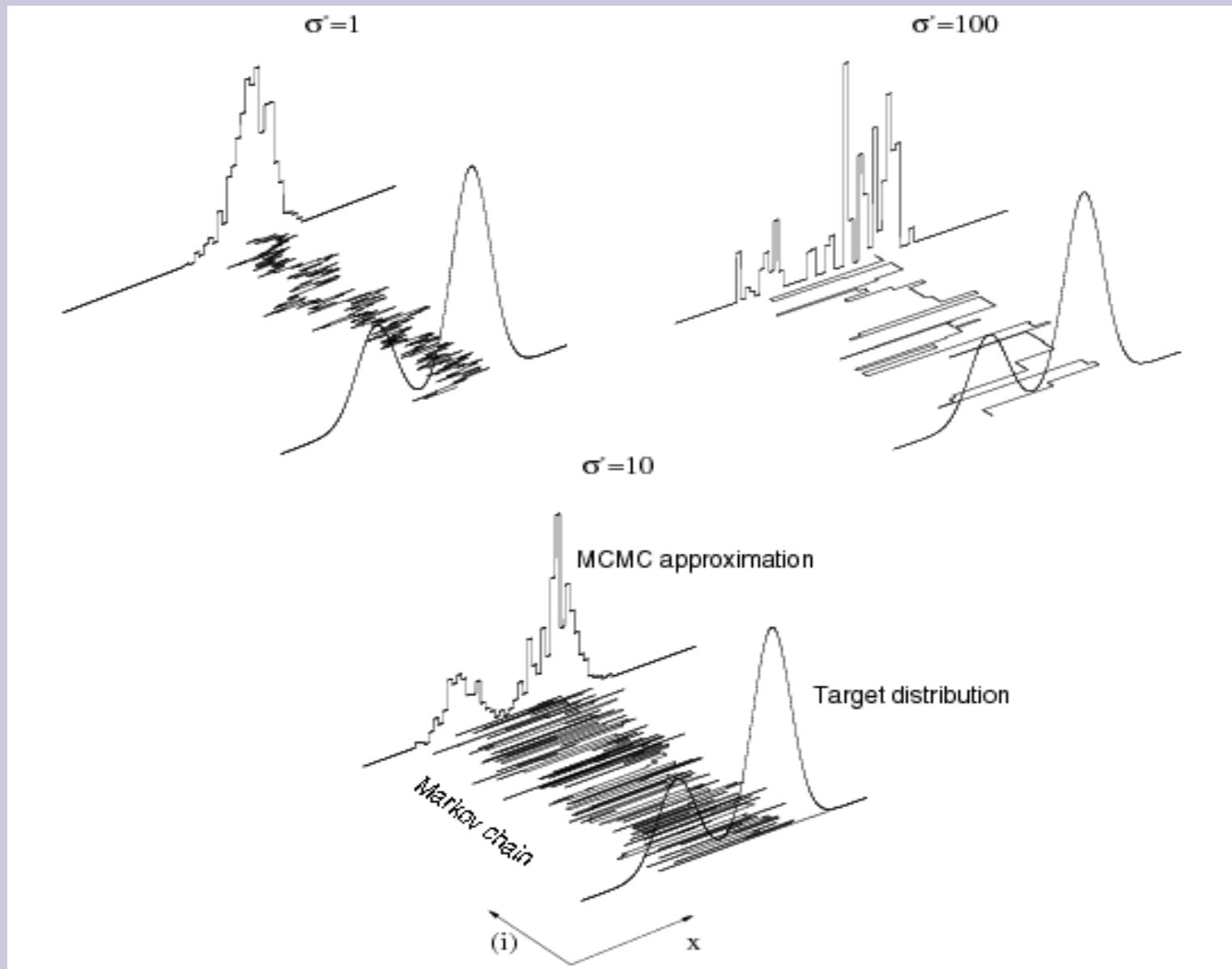
1. Initialise  $x^{(0)}$ .
2. For  $i = 0$  to  $N - 1$ 
  - Sample  $u \sim \mathcal{U}_{[0,1]}$ .
  - Sample  $x^* \sim q(x^*|x^{(i)})$ .
  - If  $u < \mathcal{A}(x^{(i)}, x^*) = \min\left\{1, \frac{p(x^*)q(x^{(i)}|x^*)}{p(x^{(i)})q(x^*|x^{(i)})}\right\}$ 
$$x^{(i+1)} = x^*$$
  - else
$$x^{(i+1)} = x^{(i)}$$

The Metropolis algorithm assumes a symmetric random walk proposal  $q(x^*|x^{(i)}) = q(x^{(i)}|x^*)$  and, hence, the acceptance ratio simplifies to

$$\mathcal{A}(x^{(i)}, x^*) = \min\left\{1, \frac{p(x^*)}{p(x^{(i)})}\right\}.$$

Note: The target distrib  $p(x)$  is only needed up to normalisation.

# M-H Simulations with a Gaussian with Variance $\sigma^2$



# The Metropolis Algorithm

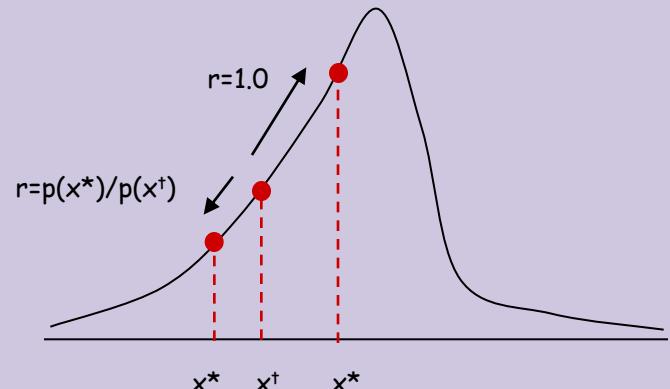
- ◆ How to pick a suitable Markov chain for our distribution?
- ◆ Suppose our distribution  $p(x)$  is easy to sample, and easy to compute *up to a normalization constant*, but hard to compute exactly
  - e.g. a Bayesian posterior  $P(M|D) \propto P(D|M)P(M)$
- ◆ We define a Markov chain with the following process:
  - Sample a candidate point  $x^*$  from a *proposal distribution*  $q(x^*|x^{(t)})$  which is *symmetric*:  $q(x|y)=q(y|x)$
  - Compute the *importance ratio* (this is easy since the normalization constants cancel)

$$r = \frac{p(x^*)}{p(x^{(t)})}$$

- With probability  $\min(r, 1)$  transition to  $x^*$ , otherwise stay in the same state

# The Metropolis Intuition

- ◆ Why does the Metropolis Algorithm Work?
  - Proposal distribution can propose anything it likes (as long as it can jump back with the same probability)
  - Proposal is always accepted if it's jumping to a more likely state
  - Proposal accepted with the importance ratio if it's jumping to a less likely state
- ◆ The acceptance policy, combined with the reversibility of the proposal distribution, makes sure that the algorithm explores states in proportion to  $p(x)$ !



# The Metropolis Convergence

- ◆ Claim: The Metropolis algorithm converges to the target distribution  $p(x)$ .
- ◆ Proof: It satisfies detailed balance

For all  $x, y$  in  $X$ , wlog assuming  $p(x) < p(y)$ , then

$$T(x, y) = q(y | x) \quad \text{candidate is always accepted, since the } r=1$$

$$T(y, x) = q(x | y) \frac{p(x)}{p(y)} \quad \text{Since, w generate } x \text{ with prob } q(x|y) \text{ and accept with prob } r = \text{the ratio} < 1.$$

$q$  is symmetric

**Hence:**

$$\begin{aligned} p(x)T(x, y) &= p(x)q(y | x) = p(x)q(x | y) \\ &= p(y)q(x | y) \frac{p(x)}{p(y)} = p(y)T(y, x) \end{aligned}$$

# The Metropolis-Hastings

- ◆ The symmetry requirement of the Metropolis proposal distribution can be hard to satisfy
- ◆ Metropolis-Hastings is the natural generalization of the Metropolis algorithm, and the most popular MCMC algorithm
- ◆ We define a Markov chain with the following process:
  - Sample a candidate point  $x^*$  from a proposal distribution  $q(x^*|x^{(t)})$  which is not necessarily symmetric
  - Compute the importance ratio:

$$r = \frac{p(x^*) q(x^{(t)} | x^*)}{p(x^{(t)}) q(x^* | x^{(t)})}$$

- With probability  $\min(r, 1)$  transition to  $x^*$ , otherwise stay in the same state  $x^{(t)}$

# The Metropolis-Hastings Convergence

- ◆ Claim: The Metropolis-Hastings algorithm converges to the target distribution  $p(x)$ .
- ◆ Proof: It satisfies detailed balance

For all  $x, y$  in  $X$ , wlog assume  $p(x)q(y|x) < p(y)q(x|y)$ , then

$$T(x, y) = q(y | x) \quad \text{candidate is always accepted, since } r = 1$$

$$T(y, x) = q(x | y) \frac{p(x)q(y | x)}{p(y)q(x | y)} \quad \text{Since, w generate } x \text{ with probability } q(x|y) \text{ and accept with probability } r = \text{the ratio} < 1.$$

$$\begin{aligned} \text{Hence: } p(x)T(x, y) &= p(x)q(y | x) = p(x)q(y | x) \frac{p(y)q(x | y)}{p(y)q(x | y)} \\ &= p(y)q(x | y) \frac{p(x)q(y | x)}{p(y)q(x | y)} = p(y)T(y, x) \end{aligned}$$

# Modeling Visual Search Time for Soft Keyboards (Virtual Keyboards) (Research Case-Study of IIT KGP)

# Topics

- Introduction
- Models of Visual Search
- Our Proposed Model
- Model Validation
- Conclusion and Future Scope

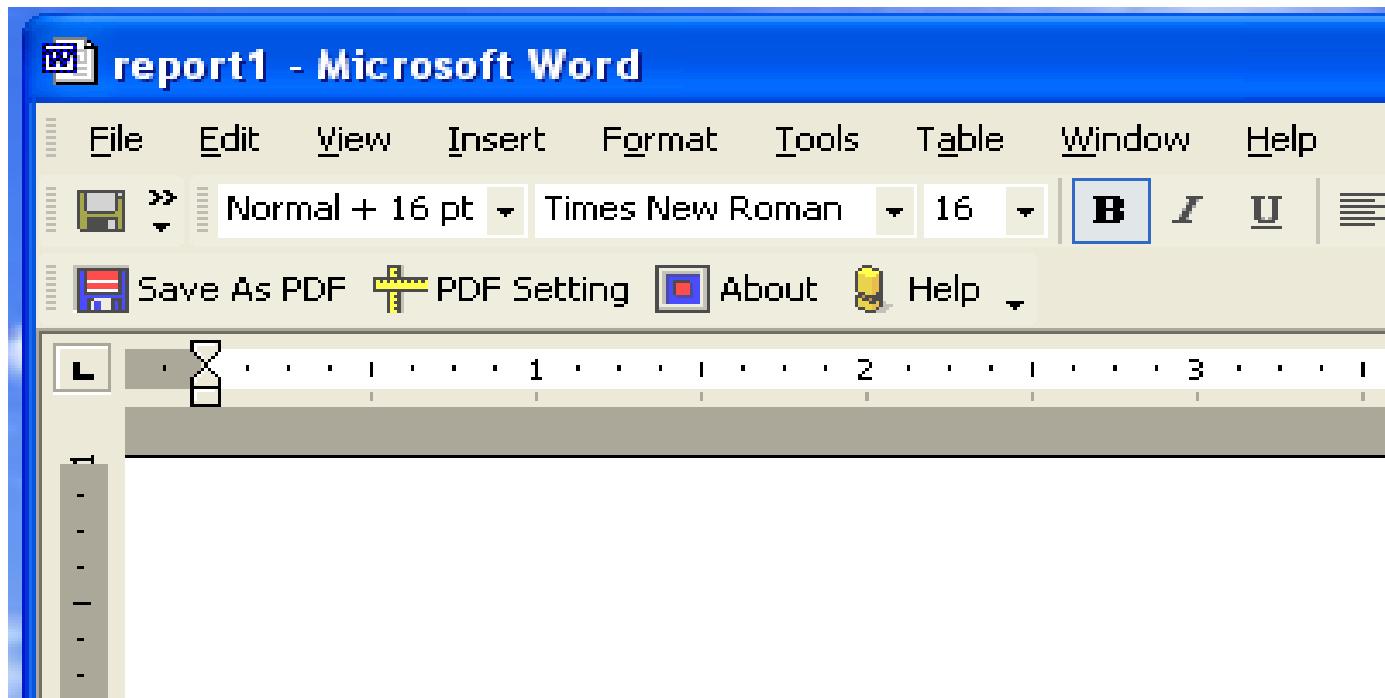
# Introduction

- What is Visual Search?
- Types of Visual Search
- What is a Soft Keyboard (Virtual Keyboard)?
- Why it is important?
- Issues in designing a Soft Keyboard

# Introduction (cont..)

## What is Visual Search?

- Visual search is our ability to detect a target among distractors
- Example: Searching File menu in *MS-WORD*



# Introduction (cont..)

## Types of Visual Search

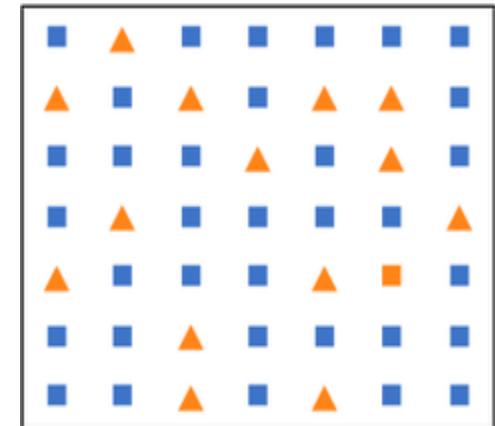
### Feature Search

Feature Search is the process of searching for targets defined by a single visual feature, such as color, size, orientation or shape, sometimes called *Parallel Search*

### Conjunction Search

It occurs when a target is defined not by any single visual feature, but by a combination of two or more features, also called *Serial Search*

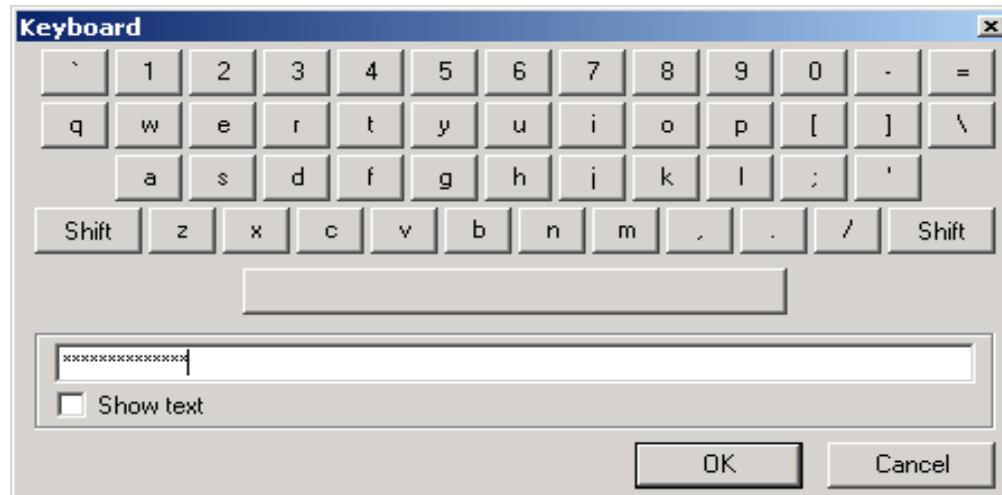
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	O	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X



# Introduction (cont..)

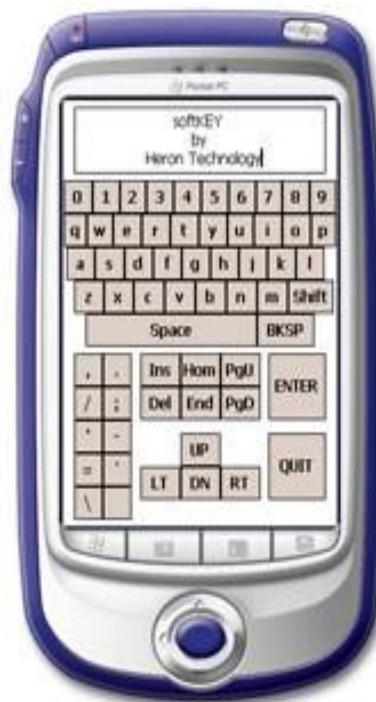
## What is a Soft Keyboard?

- Soft Keyboards (Virtual Keyboards) are on-screen representation of physical keyboards
- The time to locate one key in soft keyboard interface is known as *Visual Search time in Soft Keyboard (Virtual Keyboard)*



# Introduction (cont..)

- Why a Soft Keyboard is important
  - In Hand-held devices
  - For motor-impaired people to use the computer



# Introduction (cont..)

- Issues in designing Soft Keyboard
  - Problems in *user testing*
  - Fitts' Digraph Model
  - Deficiencies in *Hick-Hyman Law*

# Introduction (cont..)

## ■ Fitts' Digraph Model

- Used to model virtual keyboard performance
- There are 3 components in this model
  - *The visual search time component (RT)*: The time to locate the target character key on the interface. The time is modeled using the *Hick-Hyman Law*.
  - *The motor movement time component (MT)*: The time to move the finger or stylus (common input method used on soft keyboards) from the last selected key to the target key. The time is calculated using the *Fitts' Law*.
  - *Digraph probability of character pairs (DP)*: The digraph probability is calculated from a corpus of text.

# Fitts' Digraph Model

Fitts' Digraph Model consists of three parameters

- Movement time using Fitts Law

$$MT_{ij} = A + B \times \log_2(D_{ij}/W_j + 1)$$

- Visual search time using Hick-Hyman Law

$$RT = A' + B' \times \log_2 N$$

- Digraph probability (calculated from a corpus)

$$P_{ij} = f_{ij} / \sum_{i=1}^N \sum_{j=1}^N f_{ij}$$

# Fitts' Digraph Model

- Average movement time

$$MT_{MEAN} = \sum_{i=1}^N \sum_{j=1}^N (MT_{ij} \times P_{ij})$$

- Performance (CPS)

$$Novice = 1/(RT + MT_{MEAN})$$

$$Expert = 1/MT_{MEAN}$$

- Performance (WPM)

$$WPM = CPS \times (60 / W_{AVG})$$

# Introduction (cont..)

## □ Hick-Hyman Law (Hick, Hyman)

- States that the time required to make a choice among  $N$  objects given a target is a logarithmic function of  $N$ 
  - $N$  is the number of possible responses,
  - $c$  and  $d$  are constants

## □ Response Time

$$RT = c + d \log 2N$$

- $d$  was set to the reciprocal of the slowest speed of text entry (i.e. 1/5 or 0.20)

# Fitts' Digraph Model

## □ Drawbacks of Hick-Hyman Law

- No. of alternatives (N) should be based on the no. of keys in a keyboard rather than the no. of letters.
- Visual search time on a soft keyboard interface is affected by various other factors like familiarity of the organization of the letters, number of letters on each key, grouping of letters etc. in addition to the number of elements on the interface

# Models of Visual Search

## 1. Simple Search Model (Neisser)

This model predicts that the response time will increase linearly as a function of the total number of objects that must be examined

### ***Mean Predicted Response Time***

- $RT_a = \alpha_a + \beta(m_s)$  -----when target is absent
- $RT_p = \alpha_p + \beta(m_s + 1)/2$  -----when target is present

Where

- $m_s$  = The total number of items in the display
- $\beta$  = The amount of search time required to examine each item in a serial, non-overlapping search
- $\alpha$  = A constant amount of time required for other processes not related to the number of items in the display (which might change depending on whether the target is absent or present)

# Models of Visual Search (cont...)

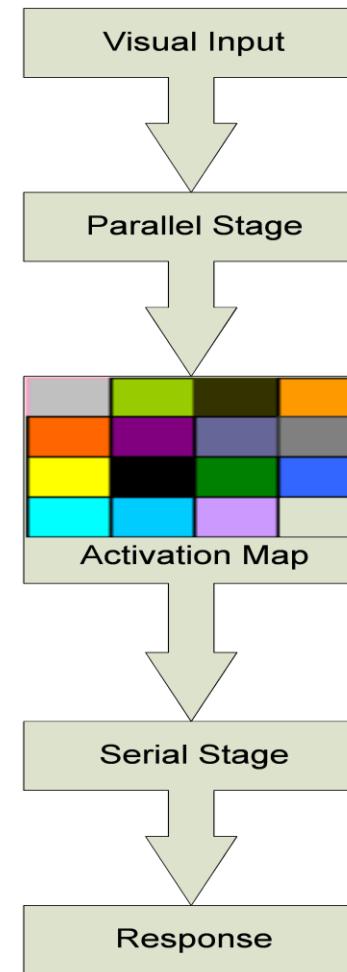
## 2. Feature Integration Model (Treisman)

- Early in the processing of visual information, a separate *feature map* is created for each primary feature (such as size, color, shape, and orientation)
- The feature maps are searched in parallel, pre-attentively, for the presence of a single object that possesses a simple feature corresponding to one of the maps
- If the target item does not possess a distinguishing primary feature, then the model must resort to a serial search process much like simple search model

# Models of Visual Search (cont....)

## 3. Guided Search Model (Cave & Wolfe)

- The Guided Search model had a pre-attentive stage and an attentive stage. Information from the first stage could be used to guide deployments of selective attention in the second stage
- In the parallel stage, the element that resembles the target and differs most from other elements is identified



# Models of Visual Search (cont....)

- Visual Search time is linearly related to the number of steps for locating the target
- The Visual Search Time is then calculated by the following formula

$$RT = \text{No. of Steps} * \text{Cycle Time} + \text{Overhead Time}$$

Where,

- **RT** = It is the reaction time or response time, the time required to search an element in the interface
- **Cycle Time** = It is the constant time required for processing of an individual element in the serial search. Its value can be taken from the cognitive science
- **Overhead Time** = It is the time for key pressing during experiment

# Models of Visual Search (cont....)

- The activation score for each and every element can be calculated by the following equation.

$$A_i = \exp\left( \frac{p}{n-1} \sum_{\substack{j=1 \\ j \neq i}}^n \frac{|f_i - f_j|}{|d_i - d_j|} \right) - q |f_i - t|$$

Where,

$A_i$  = Net activation of element  $i$  for a visual feature

$p$  = Bottom-up coefficient

$q$  = Top-down coefficient

$f_i$  = Activation for element  $i$  for that feature

$f_j$  = Activation for rest of the other elements

$|d_i - d_j|$  = Absolute difference in the distance between the  $i_{th}$  and  $j_{th}$  elements

$t$  = Activation for the target for that particular visual feature

$n$  = No. of elements in the interface

# Identified Factors for Soft Keyboard

1. Grouping
2. Size of the keys
3. Number of groups
4. Distance between the groups
5. Size of the groups
6. Number of elements
7. Shape characteristics

# Proposed Model

- Selecting Model Parameters and Factors
- Collecting Experimental Data from Search Trials
- Parameter Fitting
- Computing Activation Score, Visual Search Time

# Model Parameters and Factors

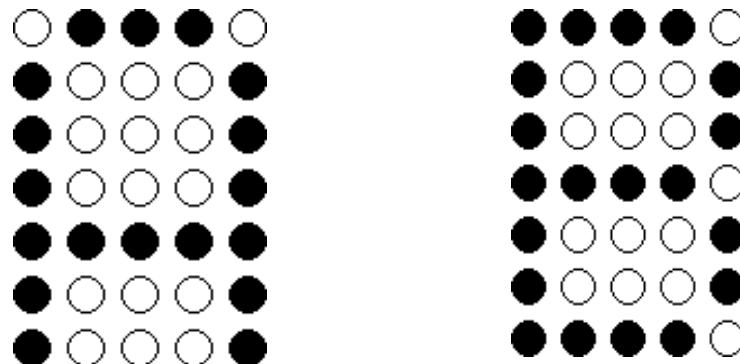
## I Factors

### *Grouping Factor*

Whenever, there are large number of elements present in the interface, grouping and labeling of elements have decrease the visual search time

### *Shape Characteristics*

It describes how much amount one element differs from other elements by shape.



# Model Parameters and Factors (cont..)

## Parameters

- *Activation Scores* for each and every elements for each and every factor
- *Bottom-up coefficient* for the corresponding factor
- *Top-down coefficient* for the corresponding factor
- *Cycle Time*
- *Overhead Time*

# Collecting Experimental Data

## Step 1

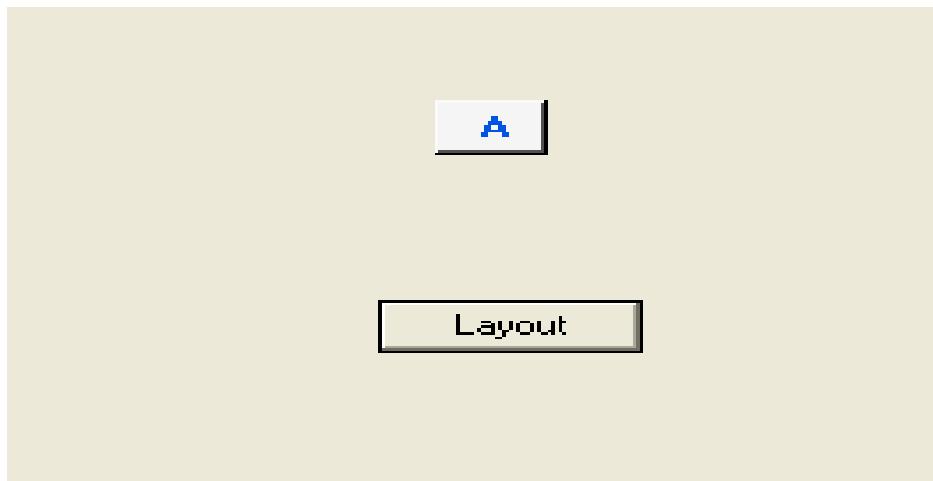
Keyboard hooking program was run by which all key press events were stored in a file in a temporary folder along with the timestamp

## Step 2

- After finishing the first step, the participants were shown a screen which is shown below

## Step 3

- As soon as the user was ready for search, the control was set on to the “layout” button. Then, the user hit the <ENTER> button



# Experiment (cont..)

## □ **Step 4**

After hitting <ENTER>button, a screen which contained keyboard layout came. Then, the participant had to search for the target

## □ **Step 5**

As soon as the participant found the target, he(she) again pressed the <ENTER> button

## □ **Step 6**

Some verifications were also done to investigate whether the participant actually saw the target or not

# Layouts for Experiment

Consonant						Vowel			Numeric			
Q	W	R	T	Y	P	K	E	A	7	8	9	
S	D	F	G	H	J	L	I	O	4	5	6	
Z	X	C	V	B	N	M	U		1	2	3	
Spacebar												
Functional						Punctuation			Operator			
Enter	Shift					,	;	.	+	-		
Caps Lock	Back Space					"	'	:	/	*		
						?	!		%	=		

1	2	3	4	5	6	7	8	9	0	←
Q	W	E	R	T	Y	U	I	O	P	Shift
A	S	D	F	G	H	J	K	L	=	Enter
Z	X	C	V	B	N	M	:	.	'	;
Caps Lock + - / * " , % ?										
Spacebar										

Alphabets						Numeric						
Q	W	E	R	T	Y	U	I	O	7	8	9	
A	S	D	F	G	H	J	K	P	4	5	6	
Z	X	C	V	B	N	M	L		1	2	3	
Spacebar												
Punctuation Operator						Functional						
,	;	.	+	-		Enter	Shift					
"	'	:	/	*		Caps Lock	Back Space					
?	!	%	=									

Alphabets						Numeric						
Q	W	E	R	T	Y	U	I	O	7	8	9	
A	S	D	F	G	H	J	K	P	4	5	6	
Z	X	C	V	B	N	M	L		1	2	3	
Spacebar												
Punctuation Operator						Functional						
,	;	.	+	-		Enter	Shift					
"	'	:	/	*		Caps Lock	Back Space					
?	!	%	=									

# Parameter Fitting

- I The average search times across the 10 participants in all the search trials were used to fit the model predictions
- I With a given set of model parameters, the model can compute a prediction of the visual search time for locating the target
- I The predicted search times were then correlated with the experimentally measured search times, and the *squared correlation coefficient* was computed
- I A single model parameter was then changed, and the predicted search times for all targets were computed again

# Parameter Fitting (cont..)

- These predicted search times were used again to calculate the squared correlation coefficient with the experimental search times
- If the parameter change led to an increase in the squared correlation coefficient, the change was kept otherwise change was undone
- This process iterated until we found no further changes in model parameters led to an increase in the squared correlation coefficient
- The model parameters were first set manually

# Computing Activation Score & Visual Search Time

- After representing the elements in the *pixel matrix* form, initial activation for shape can be calculated as follows
  - Binary values can be calculated by taking the blank pixels as '0' and filled pixels as '1'
  - From the binary representation, decimal values can be calculated for each element
  - From the initial activations, we can calculate the final activations from the activation formula
- After calculating activations for individual factors, total activations can be calculated by the following formula,  
$$\text{Activation (Total)} = \text{Activation (group)} + \text{Activation (shape)}$$

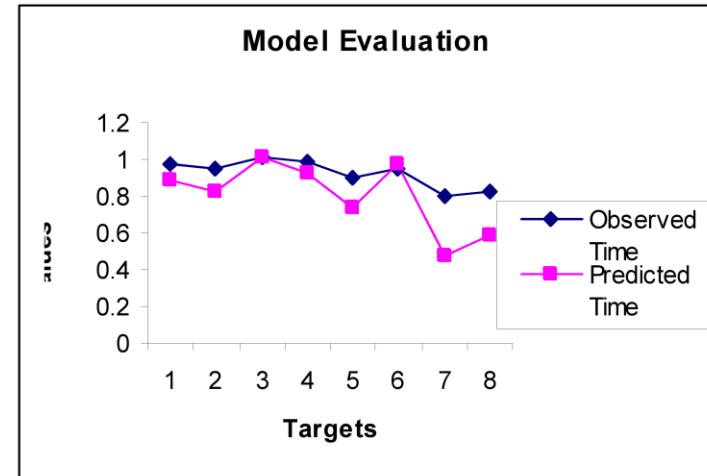
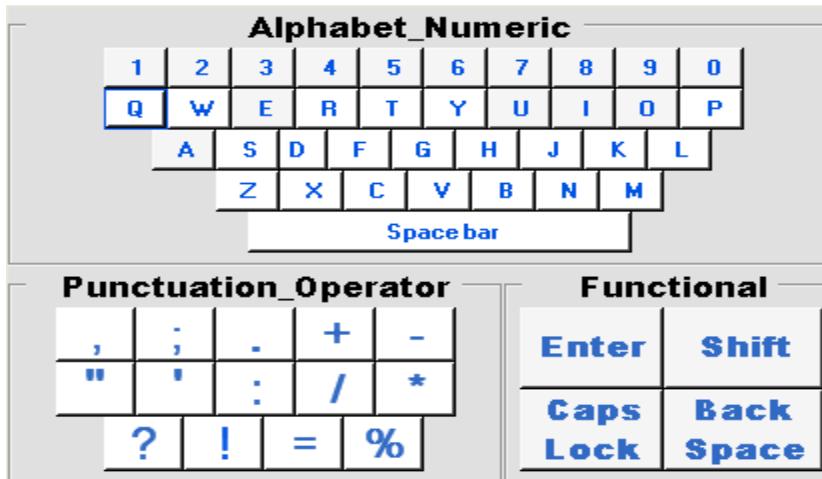
# Computing Activation Score & Visual Search Time

- Distance between two elements has been calculated by the *Euclidian Distance*
- Visual Search Time or Response Time (RT) can be calculated as:

$$RT = \text{No. of Steps} * \text{Cycle Time} + \text{Overhead Time} + \text{System Response Time}$$

- Cycle Time = 30 msec (Taken from *Cognitive Science*)
- Overhead Time = 50 msec
- System Response Time = Time delay between successive <ENTER> key presses = 150 msec  
(calculated)

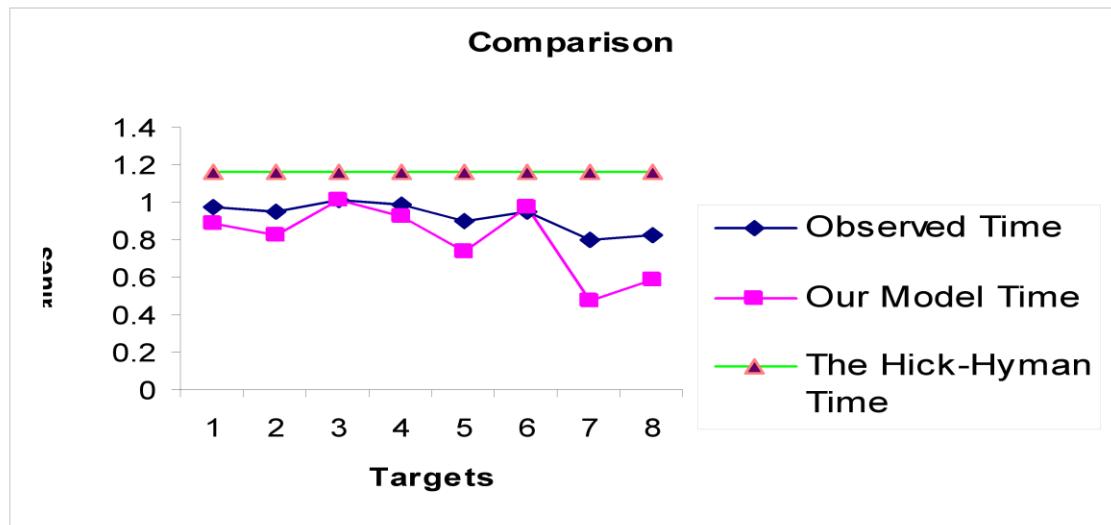
# Proposed Model Validation



Targets	Observed Time	No. of Steps	Predicted Time
A	0.980	23	0.89
B	0.947	21	0.83
division	1.016	27	1.01
G	0.984	24	0.92
I	0.906	18	0.74
S	0.953	26	0.98
Z	0.805	9	0.47
Shift	0.828	13	0.59

# Comparison Between Hick-Hyman Law and the Proposed Model

Targets	Observed time	Predicted time (The Hick-Hyman law)	% Error	Predicted time (Our model)	% Error
A	0.98	1.1563	17.9898	0.89	9.183673
B	0.947	1.1563	22.10137	0.83	12.3548
division	1.016	1.1563	13.80906	1.01	0.590551
G	0.984	1.1563	17.51016	0.92	6.504065
I	0.906	1.1563	27.62693	0.74	18.3223
S	0.953	1.1563	21.33263	0.98	2.833158
Z	0.805	1.1563	43.63975	0.47	41.61491
Shift	0.828	1.1563	39.64976	0.59	28.74396



# Conclusion and Future Scope

- █ Comparison shows that the proposed model predicts visual search time more accurately than the Hick-Hyman Law.
- █ Some more factors need to be considered like
  - █ Size of the groups
  - █ Size of the keys
  - █ Spacing between the groups
  - █ Number of groups



National Institute of Technology Karnataka  
Surathkal, Mangalore, India

# Vision Based Laser Controlled Keyboard System for the Disabled

Hiba Ahsan

Aarti Prabhu

Deeksha S D

Shridhar G Domanal

Ashwin T S

G Ram Mohana Reddy

# Contents

- Introduction
- Background and Related Work
- Proposed Keyboard Layout
- Design Methodology
- Results and Analysis
- Conclusions
- Future Work
- References

# Introduction

- The keyboard is one of the most important means of providing the computer with input for its operation.
- Commonly used keyboards such as QWERTY require the use of both hands.
- However, this poses a difficulty for the the disabled people who have the inability to move their hands.
- The core idea of our method is to design an inexpensive system for such people which can be easily assembled using an ordinary webcam and a laser pointer.

# Introduction

- We have designed a Unistroke keyboard system with optimized character placement based on frequently used Digraphs and Trigraphs.
- A webcam is positioned to monitor the keyboard and the characters are identified based on the laser pointer which the user can control by minor head movements.
- In addition, presence of the Shift key helps in accommodating more characters.

# Background and Related Work

- Layout optimization has been studied to improve text entry. The frequency of digraphs has been considered and layouts such as OPTI, have been proposed.
- Gesture keyboards such as Cirrin do not require discrete movements but instead, a single continuous movement to type a single word.
- Erdem et. al. designed a laser keyboard which employed a webcam based laser tracking. The letters in the keyboard are arranged in alphabetical order.

# Character Layout

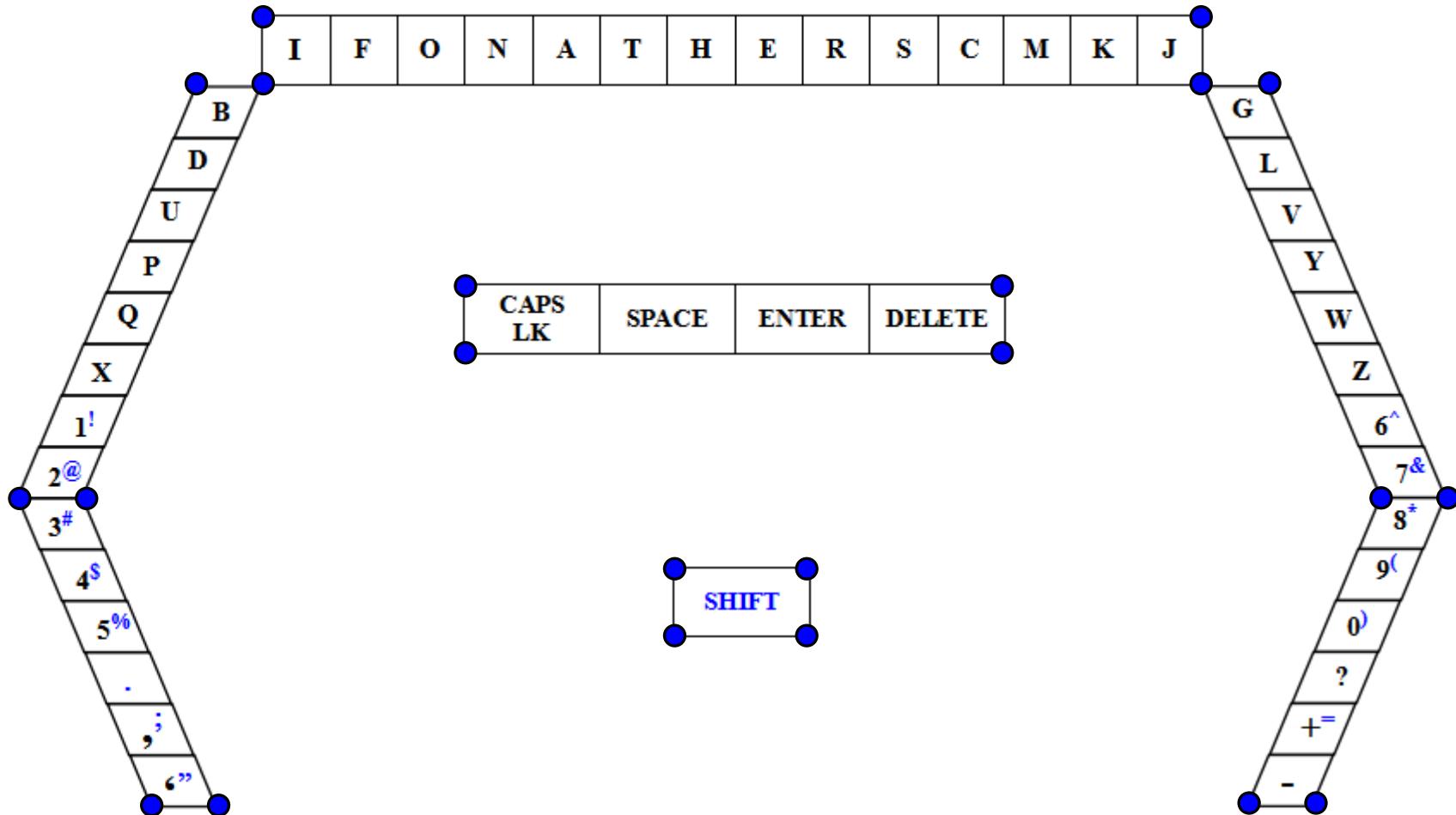
**Table 1. Number of digraphs expected in 2,000 words of English text**

Digra ph	Frequ -ency	Digra ph	Frequ ency	Digra ph	Frequ ency	Digra ph	Frequ ency
th	<b>50</b>	nd	<b>30</b>	ea	<b>22</b>	ou	<b>17</b>
er	<b>40</b>	ha	<b>26</b>	ti	<b>22</b>	ar	<b>16</b>
on	<b>39</b>	at	<b>25</b>	to	<b>22</b>	as	<b>16</b>
an	<b>38</b>	en	<b>25</b>	it	<b>20</b>	de	<b>16</b>
re	<b>36</b>	es	<b>25</b>	st	<b>20</b>	rt	<b>16</b>
he	<b>33</b>	of	<b>25</b>	io	<b>18</b>	ve	<b>16</b>
in	<b>31</b>	or	<b>25</b>	le	<b>18</b>		
ed	<b>30</b>	nt	<b>24</b>	is	<b>17</b>		

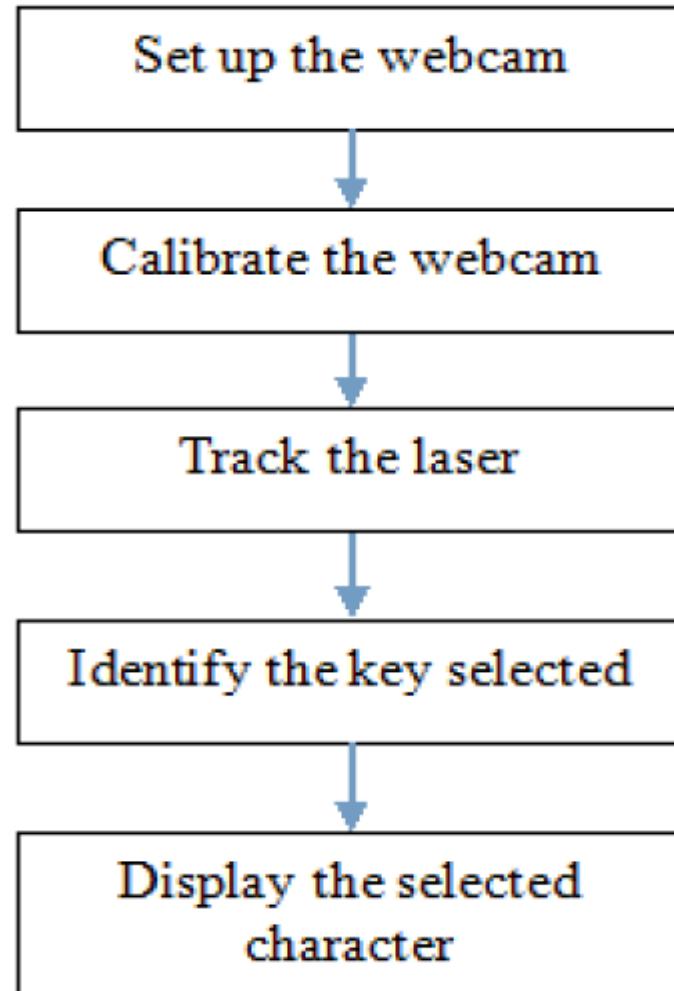
**Table 2. Fifteen most common trigraphs in the English language**

No.	trigraph	No.	trigraph	No.	trigraph	No.	trigraph
1	the	5	ion	9	has	13	oft
2	and	6	tio	10	nce	14	sth
3	tha	7	for	11	edt	15	men
4	ent	8	nde	12	tis		

# Proposed Keyboard Layout



# Design Methodology



# Design Methodology

## 1. Calibrate the Webcam

- i. The method chosen for calibration relies on color detection.
- ii. Each corner point of our keyboard is marked by a blue filled circle.
- iii. These circles are detected and the orientation of the keyboard is determined accordingly.

# Design Methodology

2. Track the Laser

3. Identify and print the key selected

i. Based on the blue points detected during calibration, line equations are used to set the boundary for each key.

ii. The key whose boundaries within which the laser point lies is determined and that key character is displayed.

# Calibration

---

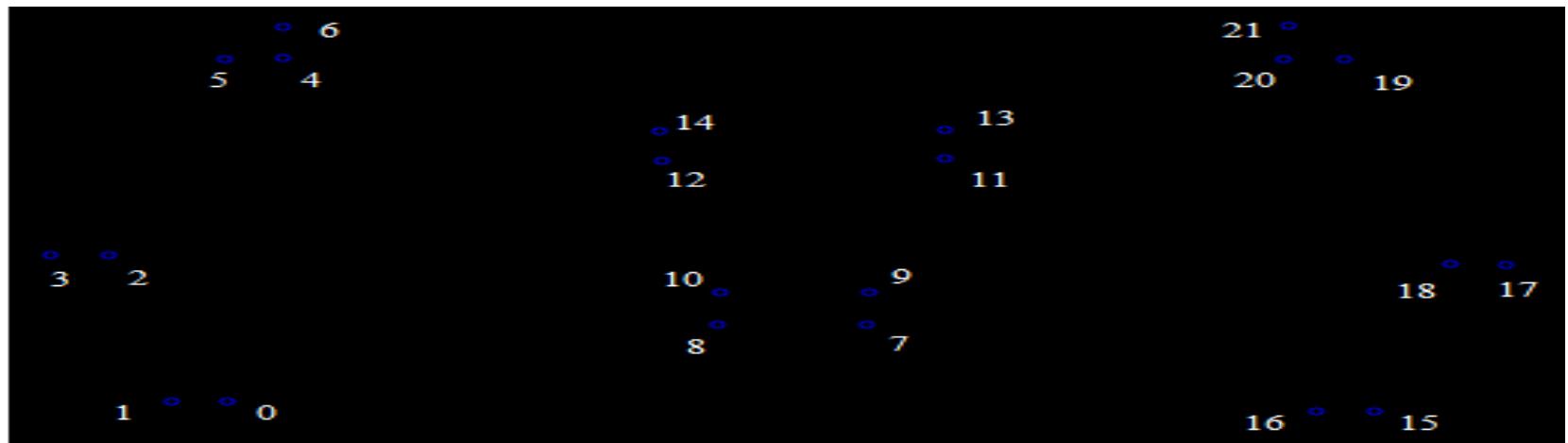
## Steps: To calibrate keyboard position

---

**Input:** Captured frame.

**Output:** Set of points representing the corners of the keyboard.

- 1: Convert captured frame to hsv.
- 2: Define the range for blue color detection (lower and upper bounds)
- 3: Get the contours that lie within this range.
- 4: Filter the contours based on area.
- 5: Store the center of each of these contours.
- 6: Use these to calibrate the keyboard angle.



# Calibration

---

**Steps:** To calibrate keyboard angle

---

**Input:** Set of points representing the corners of the keyboard.

**Output:** The calibrated keyboard.

**1:** Sort all points according to x distance

**2:** Divide sorted points into 3 parts as

Left: Points 0 – 6, Mid: Points 7 – 14,

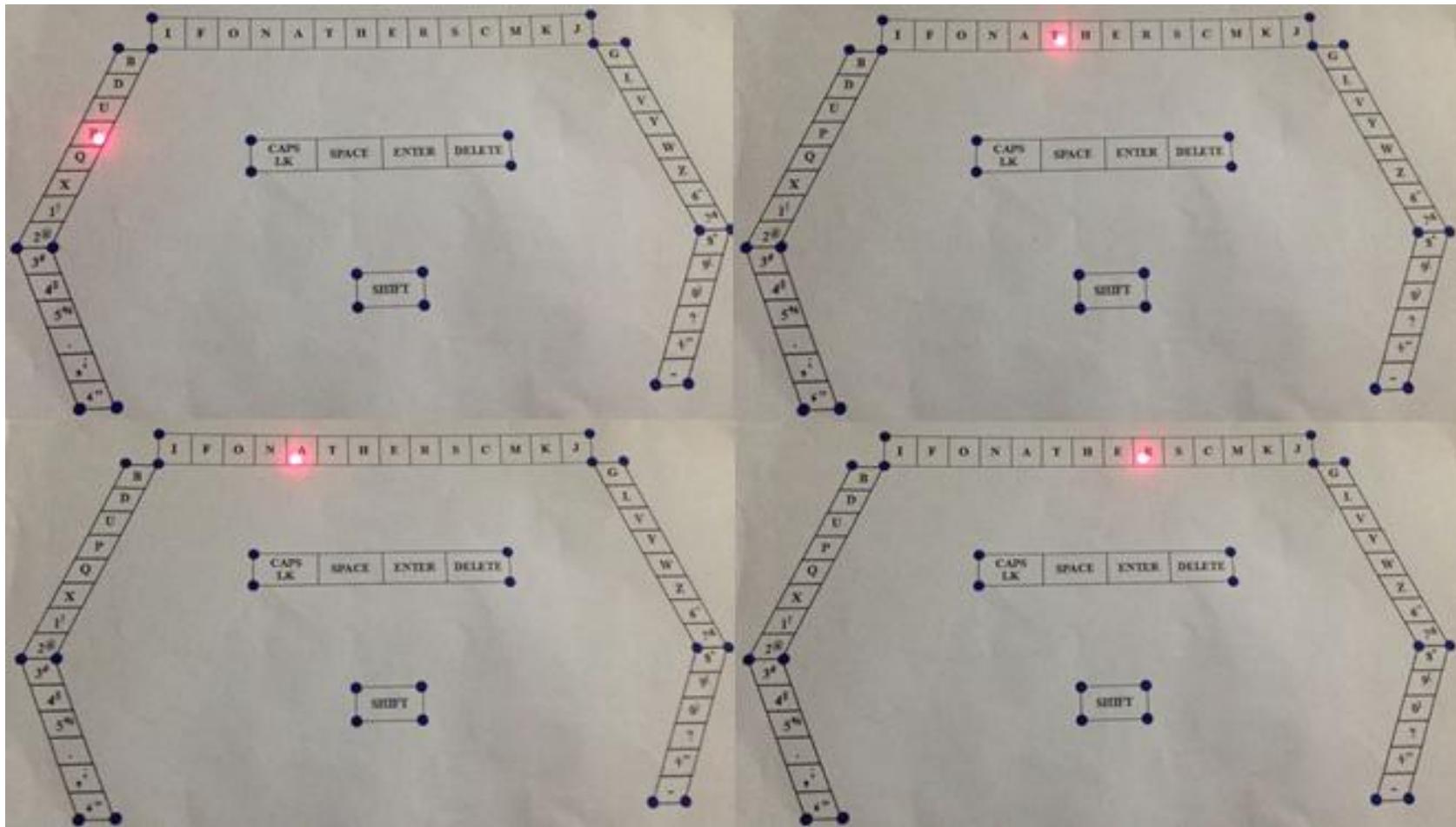
Right: Points 15 - 21

**3:** To each of these parts apply the following

a. Sort based on y distance

b. Sort consecutive pairs of points based on x distance

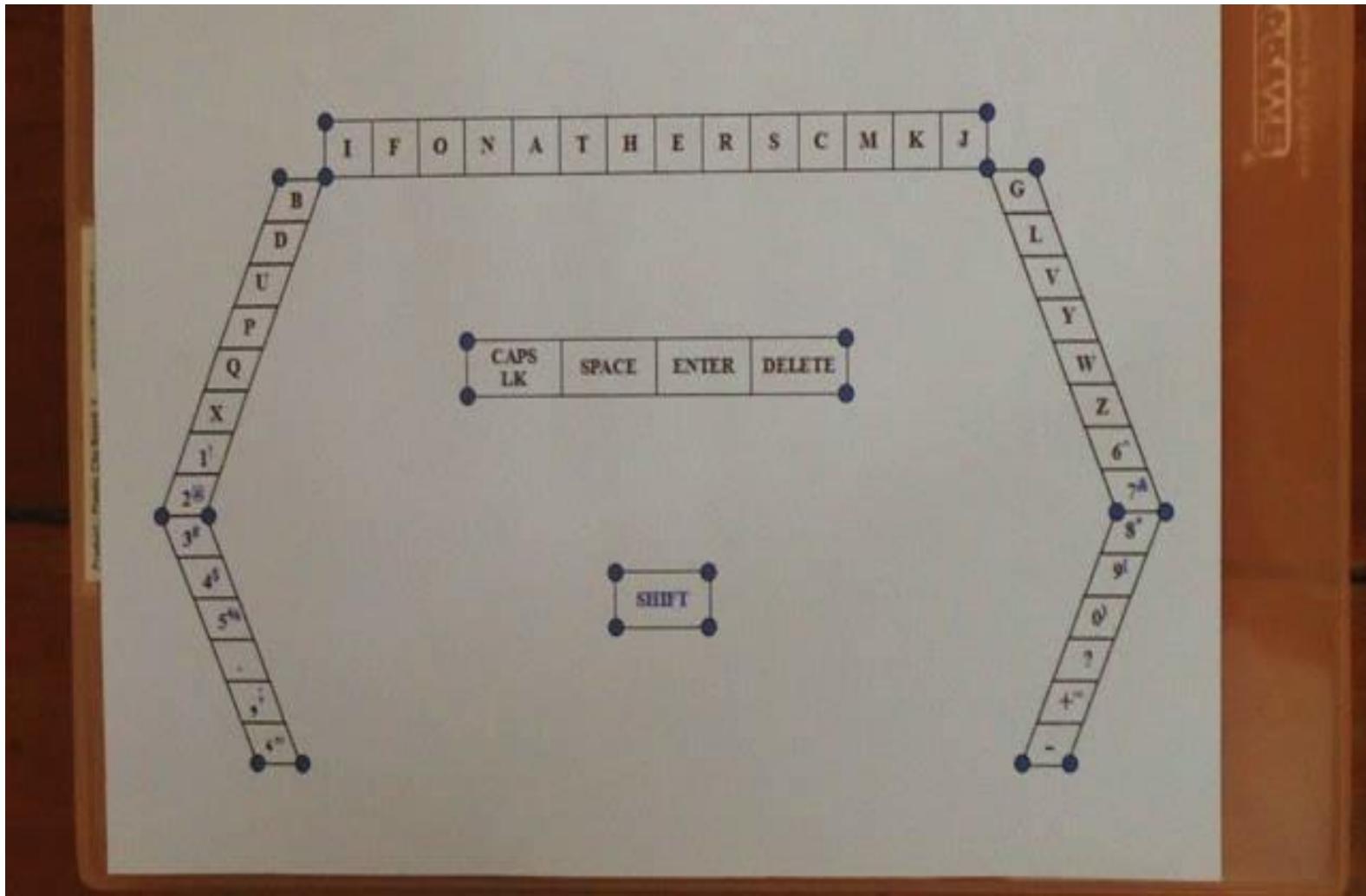
**4:** Recreate the keyboard on the screen using these points.



**Laser tracking of the word 'part'**

# Demo Video

(THIS KEYBOARD FOR THE DISABLED)



# Results & Analysis

Time Taken, Total Error Rate and Not Corrected Error Rate Recorded for Each Subject and Trial

Subject No.	Subject 1			Subject 2			Subject 3			Subject 4		
	Trial No.	wpm	Total Error Rate (%)	Not Corrected Error Rate (%)	wpm	Total Error Rate (%)	Not Corrected Error Rate (%)	wpm	Total Error Rate (%)	Not Corrected Error Rate (%)	wpm	Not Corrected Error Rate (%)
Trial 1		10.97	24.24	9.09	9.25	21.21	6.06	10.06	27.27	9.09	9.81	24.24
Trial 2		11.04	21.21	6.06	9.63	27.27	9.09	10.26	27.27	9.09	10.03	24.24
Trial 3		11.28	21.21	9.09	10.26	18.18	9.09	10.26	24.24	9.09	10.40	21.21
Trial 4		11.96	18.18	3.03	11.11	18.18	0.00	10.62	21.21	6.06	10.88	18.18
Trial 5		11.96	21.21	0.00	12.63	15.15	3.03	11.39	18.18	6.06	11.58	18.18
Average		11.44	21.21	5.45	10.58	20.00	5.45	10.52	23.63	7.88	10.54	21.21

# Conclusions

- The average session speed varies from 10.52 - 11.44 wpm.
- Average Total Error Rate ranges from 20.00 - 21.21% while average Not Corrected Error Rate ranges from 5.45 - 7.88%.
- An increasing learning rate is observed.
- With each trial, testers grew more familiar and improved, and is evident from decreased error rate & increased speed.

# Future Work

- Since calibration depends on the color of the circles at the corners of the keyboard, it is affected if any blue object of approximately the same size as the circles is captured.
- Future work will be focusing on improving the calibration technique and including commonly used keyboard shortcuts for copy, paste etc.

# References

- I. S. MacKenzie et al. “The design and evaluation of a highperformance soft keyboard,” in Proc. Extended Abstr. Human Factors Comput. Syst., 1999, pp. 25–31
- Jennifer Mankoff and Gregory D. Abowd. Cirrin: A word-level unistroke keyboard for pen input. In Proceedings of UIST '98. Technical note. pp.213-214
- Erdem, M. Erkut, et al. "Computer vision based unistroke keyboard system and mouse for the handicapped." ICME'03. Proceedings. 2003

## Ember: A Smartphone Web Browser Interface for the Blind

Ruchika S, Susmitha Pulakhandam, Isha Singh Jassi, and G Ram Mohan Reddy

(Department of Information Technology, National Institute of Technology Karnataka Surathkal,  
Mangalore, India)

**Abstract** Ember is a smartphone web browser interface for the blind. It was designed exclusively for the blind, keeping in mind their needs and requirements. With large screen targets and easy Braille based text entry, Ember aims to make browsing an enjoyable experience for the blind user. Furthermore, Ember has a verbal command option for easier input. A simple way to find targets on the interface has been incorporated with Ember, by using a screen guard with projections at six points that correspond to the six dots used in Braille. After testing, it was found that twenty-five out of twenty-eight volunteers, on an average, could navigate faster using the Ember audio navigation method than the traditional web browser, and the learning curve for the Ember interface is less steeper than the learning curve for the traditional browser. The Ember keypad was also tested against the QWERTY keypad and the SmartBraille, a well-recognized keypad for Braille-based text entry, and was found to be more intuitive and natural to use.

**Key words:** HCI; smartphone interface; Braille; audio haptic feedback; multi-touch screen

**Ruchika S, Pulakhandam S, Jassi IS, Reddy GRM. Ember: a smartphone web browser interface for the blind.** *Int J Software Informatics*, Vol.9, No.1 (2015): 23–36.  
<http://www.ijsi.org/1673-7288/9/i206.htm>

### 1 Introduction

In today's technologically advanced world, touch interfaces have become significant and are a necessary part while designing any new interface for users. We have ATM machines, personal computers, smartphones, etc. all using this vital interface. Smartphones, in fact, have become an indispensable part of our lives. A smartphone not only has the features of the traditional mobile phone, but also those of other devices such as a media player, a camera, GPS, etc. Functionalities of many devices are available within a single device and hence it becomes important that it is available to all types of users. The blind are at a loss here and hence it is required that novel approaches be made to provide these functionalities to these users.

The touch interface can be evolved in a way such that the blind can use it effectively. The traditional touch and locate paradigm that they follow cannot be applied with touch interfaces of a smartphone. To improve the usability quotient for a blind user haptic and audio feedback can be provided for every touch on the

---

Corresponding author: G Ram Mohan Reddy, Email: profgrmreddy@gmail.com  
Received 2014-29-12; Revised 2015-04-30; Accepted 2015-04-30.

interface<sup>[11]</sup>. The interface design had to be intuitive, quick to learn and easy to use, this required analysis and insight.

For using internet on their desktops and personal computers the blind use screen reading software such as JAWS and the like. A disadvantage with the screen readers being that the user becomes dependent on how the website is structured as the screen readers read out every line on the website line by line in a linear fashion. Also, screen readers are not portable and meant only for the desktop personal computers as of now. Another approach is using Braille readers, these readers convert every line of text on a website into Braille for the user to read line by line again. This provides a familiar setup to the blind user, but may not be advisable when the user doesn't want to read line by line. The choice to take selected information from a webpage does not lie with the user. Table 1 shows the problems associated with each approach. The Ember interface hopes to eradicate these issues by providing tactile method of interaction which comes most naturally to the blind.

**Table 1 Comparison of the strengths and weaknesses of Braille readers and screen reading software**

Approach	Strength	Weakness
Braille Readers	Incorporates Braille and tactile; familiar	Can only read one line at a time, bulky, no information about layout of the page
Screen Reading Software	Easy to use, knowledge of Braille is not a prerequisite	Effectiveness depends on the website developer can be slow and inflexible

It was understood that the blind could use the edges of the smartphone as bearings to locate targets on the screen and hence the targets for the interface were designed in such a way that they were easily locatable. The important menus were all located on the four corners of the smartphone screen such as the tools, backward, forward, home, audio input menu, etc. This arrangement left a huge middle space for the text input, the user could touch anywhere within this area and the Ember keypad would pop up for receiving the Braille input. Unlike traditional QWERTY keypad the keys and targets in the Ember interface were large and easier to find and use. The Ember keypad also provided audio input at all stages.

The key contribution of our work is the ability this interface provides to blind users with respect to navigating around the web on their touch screen mobile phones. Web developers and website designers are usually so engrossed in designing a website that is compatible with many different browsers and different platforms that they keep designing websites compatible with screen guards or Braille readers and this is where Ember comes in. Ember would help to the Blind users to use a web browser more easily and input text more easily. Ember would help users navigate websites better on their smartphone.

The rest of this paper has been organized in the following manner. Section 2 holds related work that has been previously carried out in this regard. It also compares and contrasts our proposed work and the various literary works mentioned in this section. Section 3 describes in great detail the methodology followed and the working of the web browser interface. Section 4 discusses the strengths & shortfalls of implementation details with tests carried out and the corresponding results & analysis.

It also holds the tests and results with the analysis. Section 5 deals with concluding remarks and Section 6 highlights the future directions of the proposed work to be extended to overcome the various shortcomings that were observed in the proposed work.

## 2 Related Work

Ember extends previous research on the accessibility of touchscreen interfaces by providing a more suitable input method and navigation techniques which address the concerns expressed by blind users.

### 2.1 Implications for design

The *Slide Rule*<sup>[1]</sup> research proposed ideologies that helped shape the design of Ember's interface. Blind users often carry a multitude of devices even though most devices overlap in function because a particular device may provide a suitable interface for certain functionalities, but prove poor in another respect. The research showed that the difficulties blind users have include difficulty in learning where objects were located on the screen. Some informants also mentioned that they were afraid of making mistakes that could not be undone.

The *Slide Rule* incorporates various gestures, the L-select gesture in particular, to navigate through lists, etc. Based on this feedback Ember was designed for the touchscreen mobile because the touchscreen can incorporate the functions of several devices into a single device. The interface should allow the users to interact with familiar layouts. Ember's skeleton layout stays the same throughout all navigation phases so the learning process is made easier. The *Slide Rule* research also showed that touchscreen interfaces must be easy to explore and minimize the need to find a particular point on the screen through trial and error. Ember's layout uses the edges of the keyboard as bearings, provides large buttons and a very large area to prompt text input. As the user passes over a button or text field with his finger, the label of the widget is read out to him. The need to pinpoint a specific area of the screen is almost negligent when the verbal command option is used. The consistent audio feedback and verbal command options make Ember's design functionally more apt than the *Slide Rule*.

### 2.2 Acknowledging the need for a different method of touch based text entry

In Ref. [2], four different methods of text input were analysed. QWERTY<sup>[3, 9]</sup>, MultiTap<sup>[6]</sup>, NavTouch<sup>[4]</sup> and BrailleType<sup>[5]</sup> were compared by testing their effectiveness across users of different genders, age, spatial ability, verbal IQ and Braille reading and writing in words per minute. The study aims to assess the advantages and limitations of different touch-based text-entry approaches, and to show that different blind users with different individual attributes can benefit from one method over another. The results showed that there was not a consensus on most methods except that BrailleType users collectively agreed that they would use the system.

BrailleType takes advantage of the capabilities of those who know the Braille alphabet. The touchscreen is divided into six large cells, each representing the dot positions. Users can explore painlessly as audio feedback is provided and double tap is

required to select a target. By reducing the number of on screen targets, as compared to the QWERTY pad, for example, less stress is placed on the blind user. Although BrailleType was the method which posed the least number of errors, there were still errors. The errors were mostly caused by the timeout mechanism of the keypad. The keypad did not allow experienced users to type faster and so the wrong letter was registered when the user tried to move on to the next letter. It made the process of text input very slow.

The Ember Keypad does not have this problem there is no need for a timeout. The keypad is designed in the way that all the targets to be pressed at once and are registered simultaneously using multi-touch. The phone is held horizontally with the touchscreen facing away from the user such that they can use two hands simultaneously to type. Letters can be typed continuously in accordance to the user's comfort level. The letter is repeated to the user after it registers<sup>[12]</sup>.

Table 2 describes in detail the strengths and weaknesses of the aforementioned literary works carried out by different researchers with respect to the touchscreen interfaces provided by them and the method of text entry for the Blind.

**Table 2 Table showing the strengths and weaknesses of previously done work with respect to touchscreen interfaces and mobile-based text entry for the Blind**

Approach		Strength	Weakness
Slide Rule <sup>[1]</sup>	Kane, S.K, et al.	Highlights the point that touchscreen interfaces improve usability for blind users; Incorporates a few audio-based multi-touch interaction techniques	QWERTY keyboard used; no support for larger touch screen displays
QWERTY <sup>[3,9]</sup>	Noyes, Jan.	Enables blind users to input text in a similar fashion as a sighted user using a simple screen reader	There are a large number of small targets which are difficult to locate for those who are not used to the layout.
MultiTap <sup>[6]</sup>	Gutowitz, Howard	Layout used is very similar to the ones used in keypad based devices; reduced number of targets on the screen	Number of keystrokes for text input is generally large
NavTouch <sup>[4]</sup>	Guerreiro, Tiago	Gesture approach-users can navigate horizontally or vertically using vowels as shortcuts to the intended letter	Interaction is limited to directional strokes for every text input; comparatively a slower input method
BrailleType <sup>[5]</sup>	Oliveira, João, et al.	The layout provides a representation of the Braille cell; easy to explore targets	All the necessary dots for a particular Braille character are to be selected at once.

### 3 Proposed Work

The aim of developing a smartphone web browser interface for the blind(Fig. 1) is to enable blind users to browse the internet comfortably and to still have all the features of a typical browser, such as bookmarks history, etc., available to them. There are three innovative components of the Ember design that will be discussed: the interface, verbal bookmark search by name and the Ember Braille keypad.

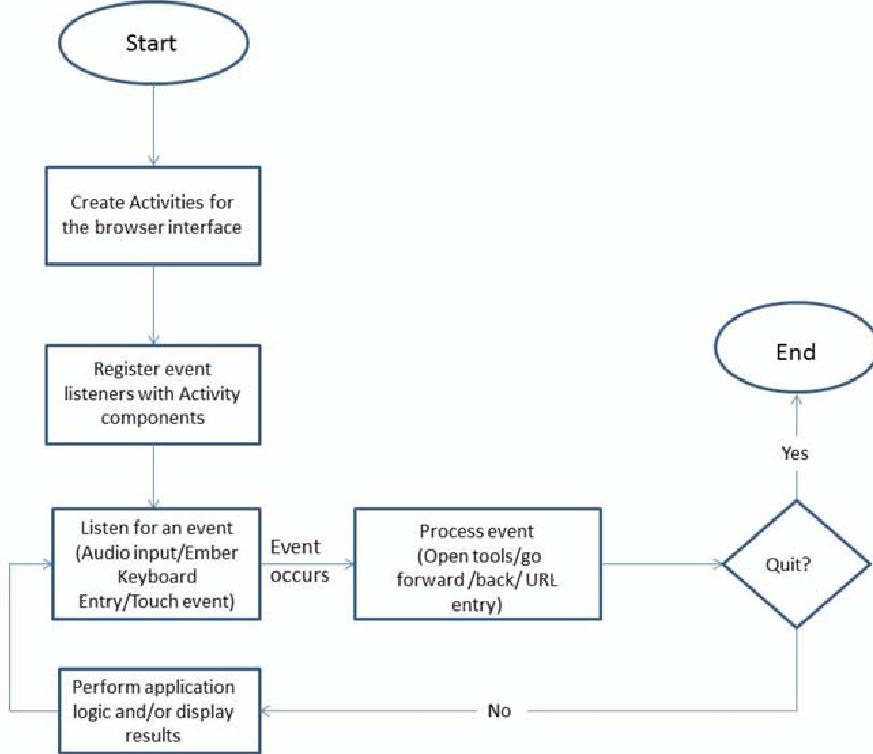


Figure 1. Framework Diagram

### 3.1 The interface

#### 3.1.1 Layout

The layout of Ember as shown in Figs. 2 and 3, aims to achieve larger target size and a less number of targets<sup>[14]</sup>. The targets must be easy to locate. There are five large buttons placed along the edges of the phone: forward, backward, tools, audio input and home. The user can use the edges of the phone as bearings. The large centre space in the middle of the layout is the edit box. There is a large area that the user can touch to initiate text entry. This way the user is never forced to pinpoint an exact location on the screen. Although the layout is not comparable aesthetically to mainstream browsers, Ember is much more compatible to the blind user's needs. The user has the option to go back to the home screen from any point in navigation. The layout of the browser is in this format throughout the application so that the user can familiarize with it easily.

Using an assist app like TalkBack on the Android platform, the user can explore the interface without the fear of making a mistake. The name of the screen element is read out to the user when they touch it. Selection is done using double tap and scrolling is done with two fingers. Even menus are announced as they pop up on the screen. The combination of easy to locate target and audio feedback allows the user to adapt to the interface easily and explore with confidence<sup>[13]</sup>.

The targets are placed at the edges of the screen and the text area is the largest section of the interface.

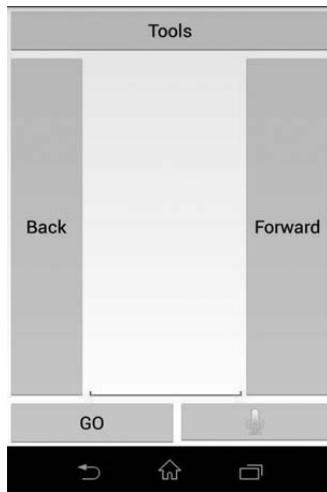


Figure 2. Screenshot of Ember home screen

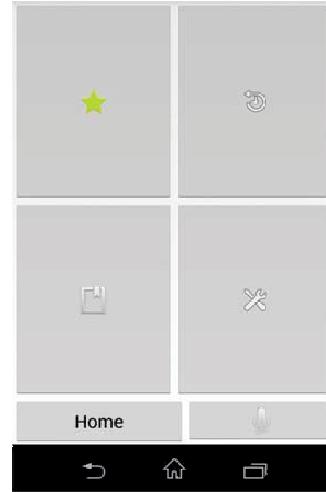


Figure 3. Screenshot of Ember tools screen

### 3.1.2 Navigation

Navigation is done by two means, by touching the appropriate button or by providing voice commands. On the long press of the audio button, the browser captures the voice command and moves to the corresponding activity on successful processing. This dual method of navigation is designed to provide the user with the most convenient method of interaction according to the situation they are in. For example if the user is in a library then they can use the touch method of navigation but if they are alone at home then the easiest method would be to just use verbal commands.

### 3.2 Verbal bookmark search by name

A particularly novel feature of Ember is verbal bookmark search by name. When the user chooses to add a page to bookmarks they can name the bookmark. The user presses the audio input button and says the name of the bookmark. The audio input is converted to text and after text to speech processing the interface plays back the name of the bookmark to the user. The user can either choose to accept the result to cancel and record again. When the user wants to access the bookmark again, they just have to give the bookmark name as a verbal command in the bookmarks menu. They can also scroll through their bookmarks as in a traditional web browser.

### 3.3 The ember braille keypad

#### 3.3.1 Text input

The Ember Braille Keypad integrates the blind user's knowledge of Braille with the requirement of having fewer targets on the touchscreen interface at once<sup>[7,8]</sup>. The

typical QWERTY keypad on a touchscreen phone has an overwhelming number of targets which can be difficult for even the most experienced blind user to use<sup>[9]</sup>. The buttons are small and it is extremely difficult to type without errors using a trial and error method. The Ember Braille keypad, like the BrailleType keypad, requires the user to map the Braille letter on the touchscreen instead of selecting a button for a particular letter<sup>[10]</sup>. The screen is divided into six zones so there are only six large targets the user has to handle. Unlike BrailleType, the Braille Keypad uses multi-touch technology to register multiple points of user interaction with the screen at a certain instance of time. This means that the user forms a letter by touching all required zones at once. For this reason there is no timeout mechanism as in BrailleType, reducing the number of errors caused by differences in user competency. The user is provided with audio feedback every time a letter is registered by the system.

### 3.3.2 Typing on the go

The Ember Braille keypad requires the user to engage at least three fingers on each hand. If the user is using the phone while walking i.e., the phone is not placed on a surface, then the user must hold the phone in a way that he can type comfortably. The phone is held away from the user as shown in Fig. 4. This method ensures that the user is not uncomfortable and holding the phone in an awkward way.



Figure 4. Typing on the go. This method of holding the phone will make typing on the Ember Keypad comfortable even while the phone is not placed on a surface

## 4. Results and Analysis

### 4.1 Testing procedure

The Ember interface navigation and Ember Braille keypad were tested by a group of thirteen blind adults who were Braille literates. Ember was tested against a traditional web browser. The Micromax A116 and the Sony Xperia C2104 were used in the testing procedure. To test the ease of navigation, volunteers were instructed to choose the menu, prompt the add bookmark activity, add a new bookmark and then search for the same in the bookmarks library. The volunteers were asked to perform the same navigation test three times for each of the web browsers. The difference in time between each of the three trials for each browser indicates the learning curve

while the average time taken by each browser shows the overall complexity that the volunteers encountered.

**Table 3 Comparison of navigation time. Navigation time shows the complexity presented by each interface. Ember interface shows less average time**

Sl. No of Volunteer	Audio(seconds)		Tactile(seconds)	
	Ember	Normal	Ember	Normal
1	187.65	195.23	140.23	151.67
2	190.25	191.66	135.72	142.53
3	199.56	201.24	142.36	145.93
4	185.37	190.34	138.24	149.25
5	195.9	197.96	139.91	145.73
6	203	208.31	141.11	153.72
7	192.66	192.23	152.71	155.97
8	200.42	206.73	147.84	152.67
9	207.89	209.28	155.71	167.23
10	204.4	202.45	143.75	159.53
11	189.59	192.52	148.89	150.77
12	192.3	193.96	152.6	155.25
13	210.73	209.24	157	161.38
14	209	211.79	139.68	144.24
15	187.34	189.46	148.45	152.76
16	199.7	204.28	152.54	156.23
17	185	187.51	156.71	163.39
18	207	209.43	149	156.93
19	205.63	208.71	150.83	158.34
20	213.52	215.27	158.32	160.27
21	188.75	190.24	144.6	151.62
22	195.92	198.92	159.12	166.92
23	199.6	199.8	150.77	159.37
24	205.45	209.23	162.62	167.67
25	203.62	205.51	153.46	160.52
26	193.68	194.82	151.78	162.45
27	196.35	198.48	149.23	159.28
28	207.73	210.23	155.57	164.96

The Ember Braille Keypad was tested by asking the volunteers to type the letters A C, H Q and R in sequence three times. The time taken by the volunteer to successfully type all five letters was recorded. The letter “A” represents a letter with only one zone interaction, “C” represents two zone interaction, “H” three zone, “R” four zone and “Q” five zone interaction. The same test was repeated with the traditional QWERTY keypad. The comparison between the total times of each trial between both keypads represents the difference in complexity presented by each one.

The Ember Keypad was compared to SmartBraille, designed by tm-laboratory. This is another Android application keypad with Braille support. The approach to Braille mapping is very different between the two applications. SmartBraille is designed for users to enter letters in Braille using a tap or a flick. Its operation is

split into three stages from the top down, like a  $3 \times 2$  matrix. If the point to the left is to be filled, then the user must flick left. The user flicks right to fill the point to the right and down to fill both. After each phase the screen is tapped. The volunteers were asked to type a phrase with both keypads. The user's improvement from trial to trial indicates the learning curve associated with either keypad.

#### 4.2 Results

Table 3 shows that twenty-five out of twenty-eight volunteers could on average navigate faster using the Ember audio navigation method than by using the traditional web browser. Twenty-eight out of twenty-eight users could navigate faster using the Ember tactile method of navigation compared to the traditional web browser navigation.

Table 4 depicts how the learning curve for the Ember interface is less steep than the learning curve for the traditional browser. The average trail times for both audio and tactile navigation for the Ember and traditional browser interface show that the time required to complete the navigation test consistently decreased in case of the Ember interface, but was random in the case of the traditional web interface.

**Table 4 Comparison of learning curve. The difference in navigation time between each trial indicates the learning curve. The Ember interface shows a decreasing trend whereas normal interface times are random**

No. of Trials	Audio(seconds)		Tactile(seconds)	
	Ember	Normal	Ember	Normal
1	210.16	211.49	157.24	159.78
2	208.12	213.96	154.93	160.03
3	205.24	207.25	152.71	161.23
4	202.39	206.44	149.45	158.52
5	201.6	210.13	145.9	162.6
6	197.23	209.61	144.56	160.96
7	194.34	208.19	142.35	164.45

Table 5 shows the average time taken by each volunteer to successfully complete typing the letters A, C, H, Q and R on the Ember keypad and the QWERTY keypad. It is seen that the time taken to type using the QWERTY keypad is significantly longer in the case of all volunteers.

#### 4.3 Analysis

The consistent downward slope presented by the Ember interface in Figs. 5 and 6 clearly shows that the volunteer's performance improves significantly with every trial. The traditional web browser shows random results with the slope showing an upward trend and downward trend over the three trials. The Ember curve in the tactile navigation map has a consistent downward slope, but the slope drastically increases from trial two to trial three in the audio method of navigation graph. This indicates that audio method of navigation is even easier to get accustomed to than the tactile method.

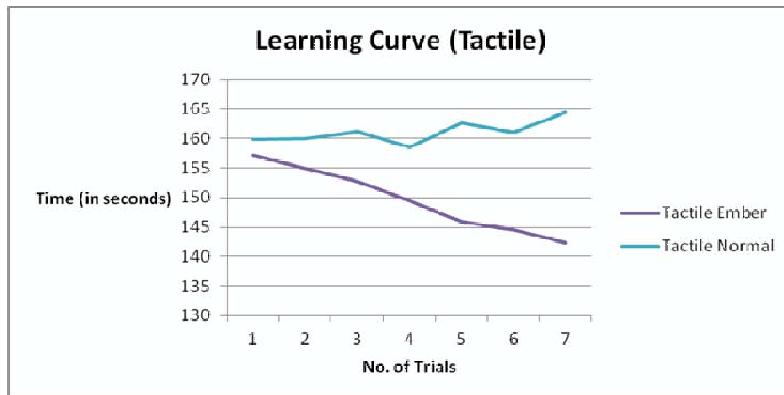


Figure 5. Navigation Learning curve (Ember v/s normal browser – Tactile)

**Table 5** Ember keypad test for complexity

Sl. No of Volunteer	Average Time (seconds)	
	Ember	Normal
1	30	59
2	26.67	49
3	33.33	51
4	39	48.33
5	27	53
6	21.33	45
7	26	53.67
8	25	41
9	32	47
10	37	46.33
11	45	55.67
12	31	52.33
13	28	41
14	26	40
15	33.33	49.33
16	30	48
17	28	45
18	45.33	45
19	32.667	40
20	33	42.33
21	27	39.5
22	29	52
23	31	48.33
24	30	45.67
25	28	41
26	21.33	45
27	32	37.5
28	29	40

Figures 7 and 8 show that the time taken to perform the navigation test was

almost always longer in the case of traditional web browser compared to the Ember interface. It is also seen that the time taken to complete the audio navigation test is more than the time taken to complete the tactile navigation test. Tactile navigation is faster than verbal command navigation.

Figure 9 shows that the time taken to complete typing the sequence of letters is significantly less in the case of Ember Keypad compared to the QWERTY keypad.

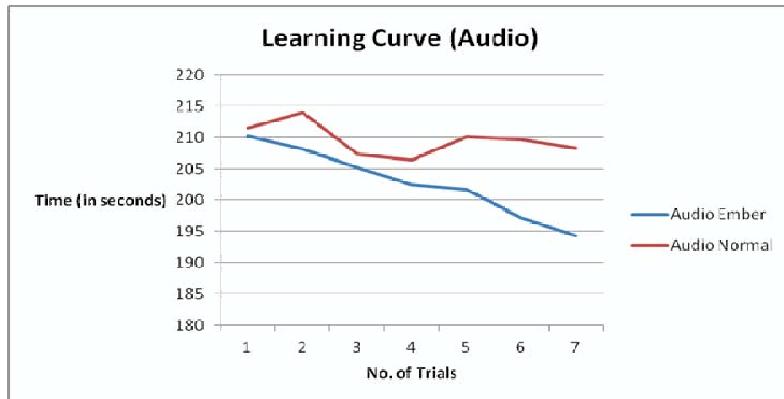


Figure 6. Navigation Learning curve (Ember v/s normal browser – Audio)

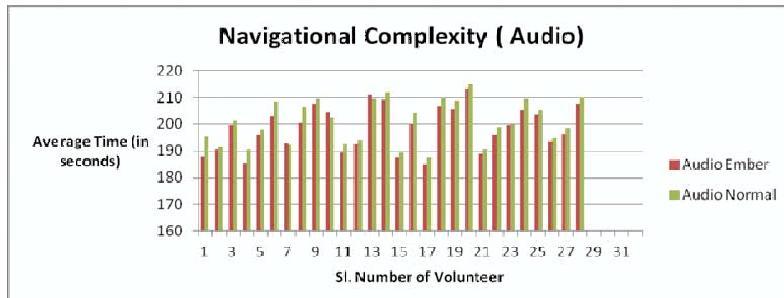


Figure 7. Navigation Complexity evaluation (Ember v/s normal browser – Audio)



Figure 8. Navigation Complexity evaluation (Ember v/s normal browser – Tactile)

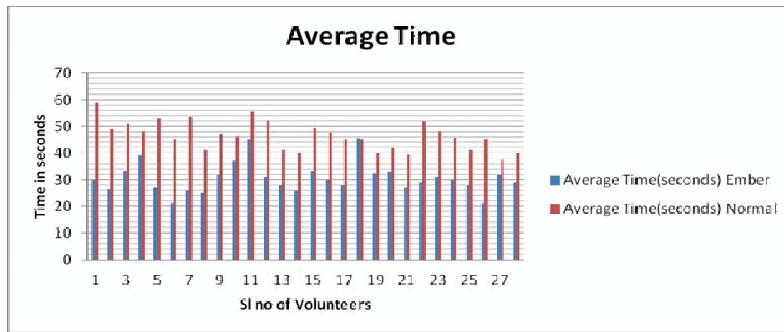


Figure 9. Complexity evaluation (Ember v/s normal browser – Ember keypad)

#### 4.4 Discussion

After testing and analyzing it was found that the volunteers found it easier to use the Ember interface compared to a traditional web browser. The analysis showed that with the use of the Ember keyboard it took them lesser time to input text. The Ember interface was found to be more accessible than the traditional smartphone web browser in terms of navigation and text input. The navigation through Ember was made easier with the use of speech input and audio feedback at every stage. The complexity of navigation (for both audio and tactile navigation) as well as the learning curve was smoother in the Ember interface compared to the traditional web browser.

The volunteers required a period of time to get accustomed to the Ember interface. It was seen that the exploration phase was relatively simple and quick when compared to the traditional web browser. A few volunteers commented that the traditional web browser was intimidating and would require a lot of practice to master. When asked which feature they appreciated most about the Ember interface, the volunteers concentrated on the ease they felt with locating large and the constant audio feedback.

It was seen that the volunteers faced difficulty with the QWERTY keypad. They found the targets too small and locating individual letters cumbersome. On average the volunteers took about one to two minutes to get accustomed to the Ember Keypad. They were comfortable with the Braille based entry. The Ember Keypad still faced errors mainly due to asynchronous selection of targets. If the volunteer intends to type a certain letter, then he must select all the required targets at the same time. The volunteers faced trouble typing the correct letter if they show hesitation while in the process of typing the letter. This delay caused the keypad to register the wrong letter.

Testing after phase one of development provided constructive user feedback. One of the points raised was that it was difficult for the volunteer to locate the six regions of the screen and place their fingers accordingly. To remediate this problem a customized screen guard was introduced. A screen guard is a plastic covering having one adhesive side that is applied on to the phone screen to protect the screen from scratches. The screen guard had six small protrusions each indicating the center of an area. After the screen guard was used it was much easier for the volunteers to find their bearings. The question of deployment was raised. Though the customized screen guard was

an elegant and simple solution to the problem, the volunteers asked questions about where such a screen guard could be obtained. Since the screen guard was made by the Ember team exclusively for testing the solution, there are no other screen guards of this type available. If the Ember keypad is to be used in the future, the customized screen guards could be made on a large scale. This solution was concluded to be a feasible and effective solution.

After testing the Ember keypad against SmartBraille, volunteers found that the Ember keypad was easier to learn and faster to type sentences because each letter in Ember was not a three phase process. Volunteers also found that Ember has a direct mapping to Braille whereas SmartBraille's is more abstract. A few volunteers found that after the initial learning curve, SmartBraille was easier to use because they did not have to map their fingers to the various regions of the Ember Keypad screen. The user only has to flick and tap the screen in the case of SmartBraille. This study signified that the Ember Keypad is on par with an established application and presents a few advantages compared to it as well.

The feedback given by the volunteers was overwhelmingly positive one of the volunteers stated that it was very easy to understand and would be extremely useful with a little practice. Furthermore, one volunteer stated that the Ember Keypad would be an enjoyable and easy way to learn braille. The ophthalmologists consult during the testing process understood Ember's design easily. They found the principles with which the interface was designed to be clear.

## 5 Conclusion

The goal of this research endeavor was to make technology and services practical and effective for blind users. With an average of twenty-five out of twenty-eight volunteers finding audio navigation with the Ember interface significantly better the traditional web browser, the Ember interface confidently presents a new platform for the blind. It also assures an optimistic experience because the Ember interface does not compromise on any features when compared to the traditional web browser. The Ember interface consists of easy-to-locate, large targets consistently placed throughout the application enhancing, usability greatly. The Ember keypad allows the blind to type using their knowledge of Braille and thus facilitates faster and accurate text input. The user experience with the Ember Keypad was further improved by using a screen guard with six small protrusions in order to locate the six regions on the screen for Braille based text entry.

## 6 Future Work

Ember is still in its nascent stages and can be improved upon in a number of ways. More features such as gestures can be added to the interface to make it more accessible, gestures to begin searching, open ember keypad, scroll lists easily. A few smartphones could also incorporate air gestures for better navigation between pages. The comparison of the Ember Keypad to other braille input methods would be a more appropriate comparison to test its usability further. As one of the volunteers had suggested, creating an Ember Keypad exclusively for teaching braille would be a rewarding extension of the project.

Ember as of now does not include a web reader, a read aloud application that would read the results of a search or listings aloud, including such a feature would make Ember more handy and easy to use.

The Ember keypad has been designed for English braille, other languages can also be incorporated effortlessly in future with features that could use gestures to input text.

### Acknowledgement

The completion of this work would not have been possible without the support and cooperation of the Roman and Catherine Lobo School for the Blind, Mangalore, Mobility Training Centre for the blind, Attavar, Mangalore, Dr. Govindraj at Prathibha Eye Hospital, Mitra Jyoti, Bangalore, a charitable trust that aims to support the blind.

### References

- [1] Kane SK, Bigham JP, Wobbrock JO. Slide Rule: Making mobile touch screens accessible to blind people using multi-touch interaction techniques. Proc. of ASSETS '08. New York: ACM Press. 2008. 73–80.
- [2] Oliveira J, et al. Blind people and mobile touch-based text-entry: acknowledging the need for different flavors. Proc. of the 13th International ACM SIGACCESS Conference on Computers and Accessibility. ACM. 2011.
- [3] Noyes J. The QWERTY keyboard: A review. International Journal of Man-Machine Studies, 1983, 18.3: 265–281.
- [4] Guerreiro T, et al. Mobile text-entry models for people with disabilities. Proc. of the 15th European Conference on Cognitive Ergonomics: The Ergonomics of Cool Interaction. ACM, 2008.
- [5] Oliveira J, et al. BrailleType: unleashing braille over touch screen mobile phones. Human-Computer Interaction – INTERACT 2011. Springer Berlin Heidelberg, 2011. 100–107.
- [6] Gutowitz H. Method and apparatus for improved multi-tap text input. U.S. Patent. 6,219,731. 17 Apr. 2001.
- [7] Sergio M, Bernareggi C, Belotti M. TypeInBraille: a braille-based typing application for touchscreen devices. The Proc. of the 13<sup>th</sup> International ACM SIGACCESS Conference on Computers and Accessibility. ACM. 2011.
- [8] Sergio M, Bernareggi C, Belotti M. Typeinbraille: quick eyes-free typing on smartphones. Computers Helping People with Special Needs. Springer Berlin Heidelberg, 2012. 615–622.
- [9] Clawson J, Lyons K, Starner T, Clarkson E. The impacts of limited visual feedback on mobile text entry for the twiddler and mini-QWERTY keyboards. Proc. Ninth IEEE International Symposium on Wearable Computers, IEEE Computer Society. 170–177. 2005.
- [10] Jussi R, et al. Methods for presenting braille characters on a mobile device with a touchscreen and tactile feedback. IEEE Trans.s on Haptics, 2009, 2.1: 28–39.
- [11] McGookin D, Brewster S, Jiang WW. Investigating touchscreen accessibility for people with visual impairments. Proc. of the 5th Nordic Conference on Human-computer Interaction: Building Bridges. ACM. 2008.
- [12] Vanderheiden GC. Use of audio-haptic interface techniques to allow nonvisual access to touchscreen appliances. Human Factors and Ergonomics Society Annual Meeting Proceedings, 1996, 40: 1266
- [13] Yfantidis G, Evreinov G. Adaptive blind interaction technique for touchscreens. Universal Access in the Information Society, 2006, 4(4): 328–337.

# **HCI: Guidelines for Design and Evaluation of Interfaces**

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# Learning Objective

- In the previous lectures, we discussed different GOMS family of models, and underlying laws. Through some research case-studies we have observed how these are used in the evaluation.
- In this lecture, we discuss HCI Design Guidelines so as enable the system designers to evaluate the existing interfaces or to conceptualise new ones.

The following are the standard guidelines

Shneiderman's eight golden rules

Norman's seven principles

Norman's model of interaction

Heuristic evaluation

Nielsen's ten heuristics

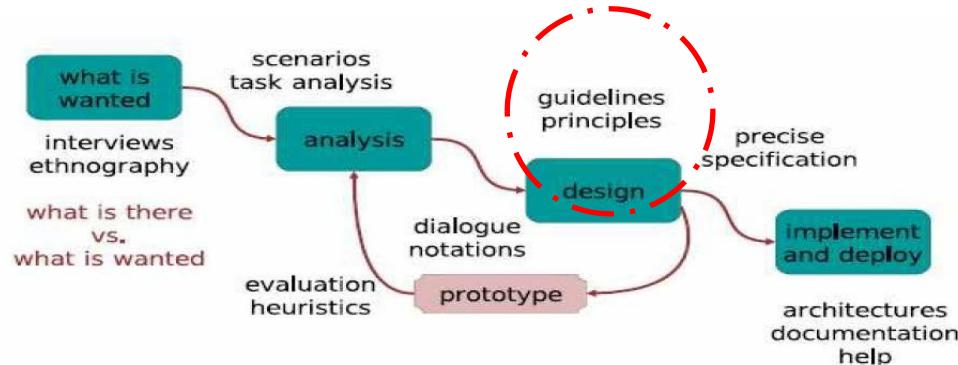
# Ben Shneiderman's Eight Golden Rules for User Interface Design

# Background

We use Alan Dix et al.'s version of the Waterfall Model to illustrate where exactly in the design cycle, the Rules - Guidelines & Principles being discussed in this module, become important.

## HCI in the design process

- Waterfall model



[Dix et al, p.195]

**Design is both qualitative as well as quantitative.**

**'Guidelines' & the 'Principles'** rather than the **'precise' laws & rules'** are used in Design.

**Shneiderman formulated eight such guidelines that can be used in Interface designing.**

# Introduction

- Ben Shneiderman\* consolidated his known tacit knowledge and practice guidelines which are used intuitively by the graphic Interface designers - into a set of **eight general guidelines** for the use of CSIT specialists who were being introduced to Visual Graphic Designers' work of designing the interactive GUI\*\* interfaces. Along with 'looks' the usability of a software depended on the functionality.
- There is ample empirical evidence published in the HCI literature which collaborates and consolidates the **applicability of the eight guide lines**.



\* Ben Shneiderman founded the HCI Lab at University of Maryland, USA; known for Nassi-Shneiderman diagrams used in the Software Engineering.

\*\*GUI:Graphic User Interface

- These are intended more as guidelines rather than 'rules' to be strictly adhered to at every step.
- They are useful for designers as well as software engineers involved in the design of interfaces.
- Using eight guidelines it is possible to distinguish a good interface design from a bad design especially from the Human-User interaction point of view.
- These have been put forth in a very concise and understandable manner by Ben Shneiderman.
- It needs to be noted that apart from these eight there are many more similar useful pointers available in the HCI and the Usability literature
- While merely or blindly applying these eight guidelines (rules) is not necessarily to guarantee a good interface 'design', they are useful in heuristic evaluation to identify the GUIs that fall out of normal 'pattern'. The guidelines can be used to rate GUI's as good or bad.

**The Eight ‘rules’ reproduced from published HCI literature are as follows.**

- 1. Strive for Consistency**
- 2. Cater to Universal Usability**
- 3. Offer Informative feedback**
- 4. Design Dialogs to yield closure**
- 5. Prevent Errors**
- 6. Permit easy reversal of actions**
- 7. Support internal locus of control**
- 8. Reduce short term memory load**

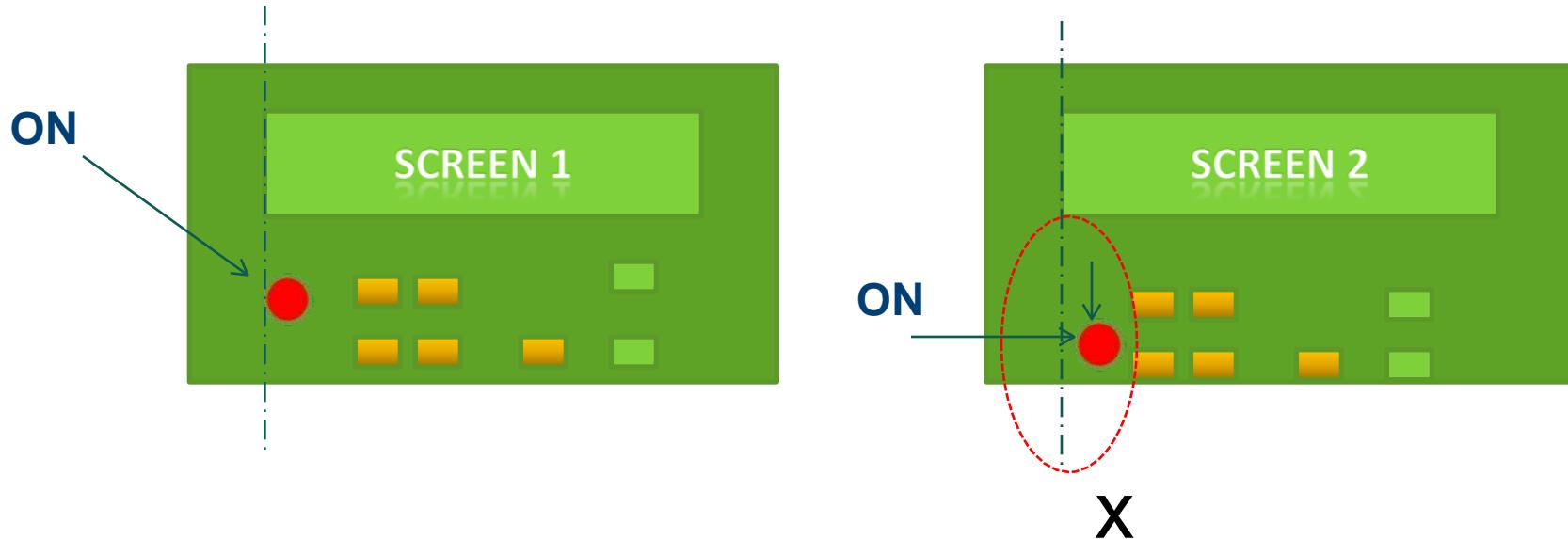
# Explanations & Examples

## 1. Strive for Consistency

- Users need to be able to do the same thing the same way that they have been doing.- every time.
- Interfaces need to exhibit ‘consistent’ quality across screens/applications both visually as well as behaviorally.
- Consistency leads to a pattern which is easier to handle cognitively.
- Consistency such as ‘similar sequence of actions in similar situations’ makes it easy to learn.

**Consistency can be achieved through graphical elements such as fonts, colour, shape, position being consistently same in all menus & screens, across, categories for a particular software.**

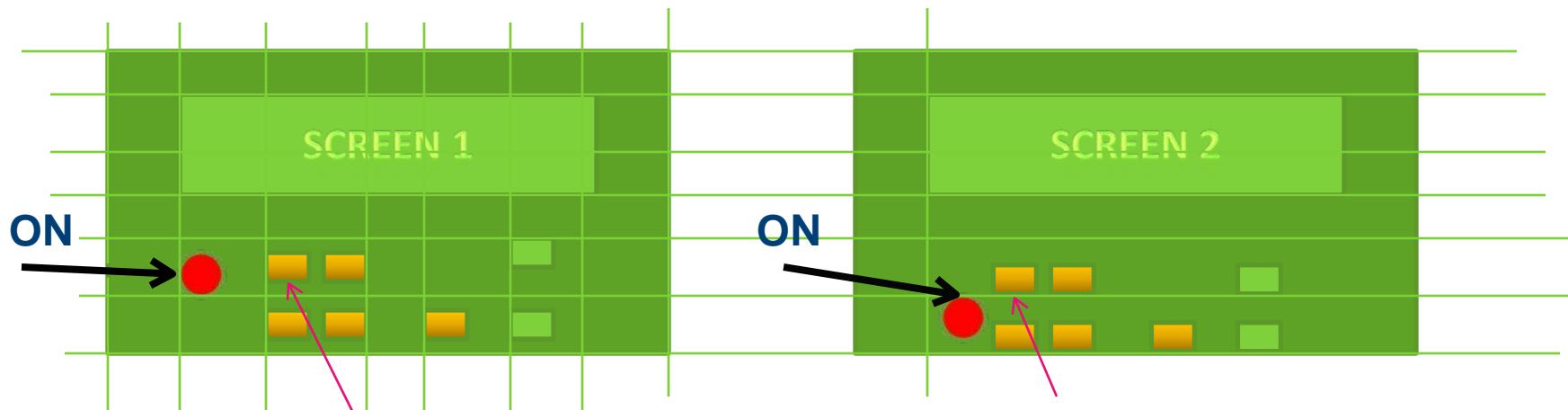
For example: If the **ON** button is on the right in the first screen and moves towards middle in the second screen then positional inconsistency is said to have occurred - however small the displacement is.



GUI designers use a simple technique to maintain the consistency of control elements in successive screen.

## Consistency (Contd...)

GUI designers use a background grid to place interactive elements in a consistent and orderly way so as to make them appear both physically as well as visually at the same place across the entire software package.



Inconsistent positioning of the GUI elements is evident when observed against a grid. Grids are used as background reference to place the elements consistently

In case of certain exception in maintaining consistency, control elements are to be made in a subsequent screen, they should be such that they are comprehensible, distinct and limited in number.

## 2. Cater to Wide Range & Type of Users

1. Strive for Consistency
- 2. Cater to Universal Usability**
3. Offer Informative feedback
4. Design Dialogs to yield closure
5. Prevent Errors
6. Permit easy reversal of actions
7. Support internal locus of control
8. Reduce short term memory load



Universal design strives to cater to as wide a range of human users of different characteristics (age, culture, educational level, disability) with a single design.

While this may not be feasible/possible in all contexts, Shneiderman's rule none the less needs to be followed so as not to leave out taking into consideration a section of users, otherwise competent, who cannot use the interface due to no fault of theirs.

Users: **Novice**, **Intermediate** and **Experts**. Experts tend to use lesser actions at a faster pace. Abbreviations, short cuts, keys etc are some of the techniques used.

**Interfaces need to cater to all levels and the classification of users: novice to experts.**

### 3. Offer Informative Feedback

1. Strive for Consistency
2. Cater to Universal Usability
- 3. Offer Informative feedback**
4. Design Dialogs to yield closure
5. Prevent Errors
6. Permit easy reversal of actions
7. Support internal locus of control
8. Reduce short term memory load

- Interfaces need to not just to be communicative but also need to inform the ‘user’ in terms of learning & feed back which tells them that they are proceeding in the right direction.
- For every action of the user there needs to be a feedback – only then ‘interaction’ (in HCI) is said to take place. Specific error messages composed in a appositive tone give affirmative feedback without having to feel punitive.
- Unless the user gets a feed back we cannot proceed or becomes unsure of the correctness of the action.

## 4. Design Dialogs to Yield Closure

- In an interaction - dialogue needs to have a closure which is recognized by the user as end of an action.
- Sequence of actions need to proceed in a dialogue by engaging the user in a step by step manner.
- Like in a mathematical expression, every enclosing bracket needs a corresponding closing bracket. So also subsequence of actions needs to be grouped with intermittent closing of each sub group followed finally by a closer action of the group.

**Ex:** A message at the end of a sequence of events gives a necessary feed back & closure of sending a SMS.

Your message has been sent. [Undo](#)

## Example 2: Un-closed dialogue



Press ON button



*Look at the green lamp.*

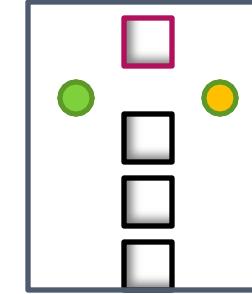


*If green glows press next push button - yellow lamp will glow*



*Push 3<sup>rd</sup> button and continue till green lamp stops glowing.*

End of task.....



## An Example of a closed dialogue:

Press ON button

*Look at the green lamp.*



*If green glows press 2<sup>nd</sup> push button and yellow lamp will glow.*



*Press 3<sup>rd</sup> button and continue with other 3 buttons till green lamp stops glowing.*



*When Yellow lamp stops glowing it indicates sequence over.*

End of task.

Notice that the yellow lamp feedback dialogue above being not closed ?

What happens to the yellow lamp? Did it stop glowing? or why it continues glowing when the task is over ?

..... Are some of the questions that may arise due to non closure of dialogues which can lead to confusion for a user

## 5. Prevent Errors

1. Strive for Consistency
2. Cater to Universal Usability
3. Offer Informative feedback
4. Design Dialogs to yield closure
- 5. Prevent Errors**
6. Permit easy reversal of actions
7. Support internal locus of control
8. Reduce short term memory load

Interfaces need to minimize the errors. Human Computer dialogue can be designed to minimize and prevent errors made by users.

There could be reasons for user errors but the user himself or herself is not one of them! Users can make errors while interacting with computers as well as while inputting/interpreting information.

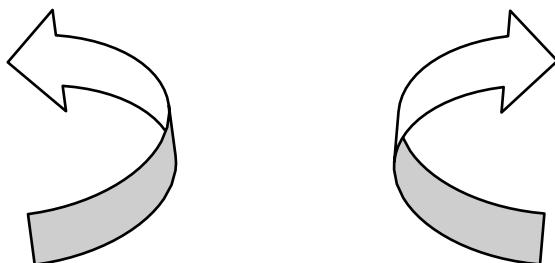
Even if the user makes an error the system needs to be designed to detect it, take corrective or precautionary steps to arrest it. It also needs to offer a way out for recovery from the error.

A default system unchanged message needs to be communicated to the user if an error has happened.

## 6. Permit Easy Reversal of Actions

Interactions need to build in retracing backwards /reverse actions if need be so as give relief from anxiety to the user. The system should encourage exploration without techno fear. One way to do this is to provide a re traceable path backwards of all actions and permit their nullification.

Ex: This PPT application has reversal in both the direction – backwards (last action) and forward (post action)



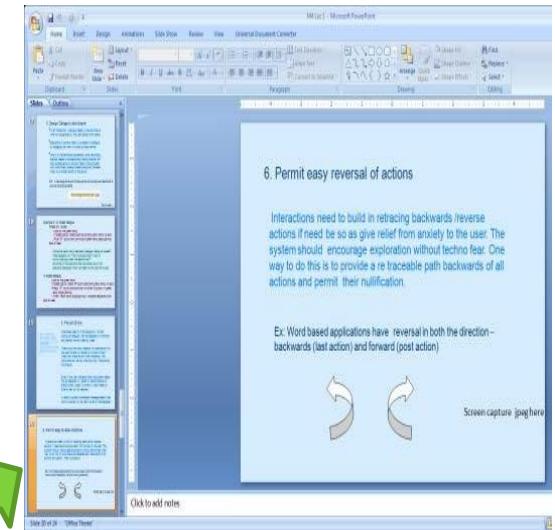
A screenshot of the Microsoft PowerPoint interface. The ribbon menu is visible at the top. In the center, there is a large green arrow pointing downwards, indicating the 'Undo' button. A red circle highlights this button. Below the ribbon, several slides are visible in the slide pane, each containing text related to the concept of permitting easy reversal of actions. The current slide is titled '6. Permit easy reversal of actions'. The text on the slide reads: 'Interactions need to build in retracing backwards /reverse actions if need be so as give relief from anxiety to the user. The system should encourage exploration without techno fear. One way to do this is to provide a re traceable path backwards of all actions and permit their nullification.' At the bottom of the slide, there is a note: 'Ex: Word based applications have reversal in both the direction – backwards (last action) and forward (post action)'. Two small curved arrows are shown at the bottom of the slide, pointing in opposite directions. At the very bottom of the slide, there is a link: 'Screen capture jpg here'.

## 7. Support Internal Locus of Control

Allow user to always feel ‘in control’ of the system and of the situation.

Make the user aware that he/she is in control. User should believe that they are controlling the system and not the other way around. This is achieved by more opportunities for ‘interactions’.

The bearing of where the user presently is helps the user to orient or reorient the interaction. The user should never be allowed to feel lost.



## 8. Reduce Short Term Memory Load

1. Strive for Consistency
2. Cater to Universal Usability
3. Offer Informative feedback
4. Design Dialogs to yield closure
5. Prevent Errors
6. Permit easy reversal of actions
7. Support internal locus of control
- 8. Reduce short term memory load**

94 56 781029

Easier to remember if chunked into smaller sets

94 56 7 810 29

Care not load the cognitive short term memory of the user by expecting user to remember several sequences, actions and its consequences at a time. Means loading its short term memory while interacting.

Miller's\* 7 chunks of information is often prescribed as a solution to limit the short term memory. In psychological experiments it has been found that the short term memory can hold  $7 \pm 2$  bits called chunks of information. Long sequential actions requiring more than 7 chunks need to be broken down into smaller chunks.

\*G. A. Miller; The Magical number seven, plus or minus two: some limits on our capacity to process information. Psychplogical review, 63(2):81–97, 1956.

Each of these Shneiderman's rules were examined with the examples

- 1. Strive for Consistency**
- 2. Cater to Universal Usability**
- 3. Offer Informative feedback**
- 4. Design Dialogs to yield closure**
- 5. Prevent Errors**
- 6. Permit easy reversal of actions**
- 7. Support internal locus of control**
- 8. Reduce short term memory load**

# Homework

Choose any common software interface. Analyze its interfaces by navigating to find out whether it adheres to the eight Shneiderman's Rules or not. Use a Novice User as your reference.



Example: Excel Sheet.

User: 10<sup>th</sup> standard student.

Present your findings in terms of number of violations per rule for the chosen software.

## References:

1. Shneiderman. B.; Designing the user interface: Strategies for effective Human Computer Interaction; Addison-Wesley Publishers Treading MA. 2004)
2. Designing the user interface: Strategies for effective Human Computer Interaction; Ben Shneiderman and Catherine Plaisant, Addison-Wesley Publishers Treading MA. 2010)

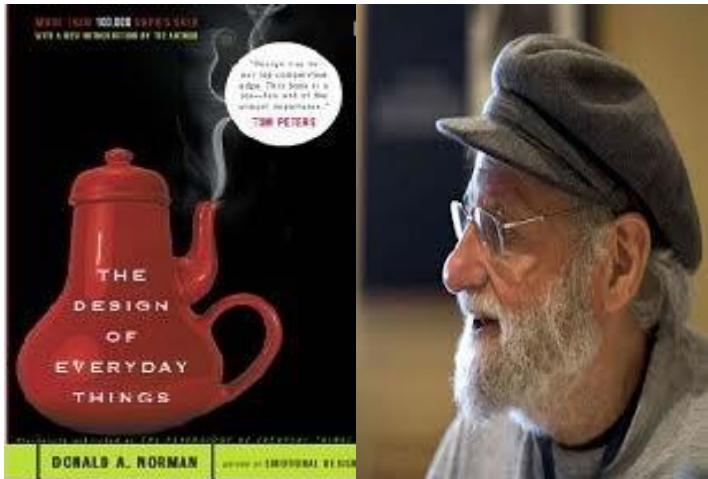
# **HCI Guidelines: Norman's Seven Principles and Norman's Model of Interaction**

Professor Ram Mohana Reddy Gudetti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# HCI Guidelines: Norman's Seven Principles

# Introduction:

Donald Norman, a Researcher, Psychologists and the Designer is well known for his book titled: "The Design of Everyday Things".



In 1988 Donald Norman proposed **Seven Principles** for the evaluation of the interaction between Humans and the Computing Systems (Computers).

Later he formulated a **model** to understand and integrate a user into the Interface design cycle.

He intended seven stages to be used to transform difficult tasks for which HCI and the interface was under development into simple ones.

Norman outlined the underlying principles for his 7 stage models as shown below. The seven stage Interaction model is shown in subsequent slides.

### Principles underlying the seven stage model

1. Use both knowledge in world & knowledge in the head
2. Simplify task structures.
3. Make things visible
4. Get the mapping right  
*(User mental model = Conceptual model = Designed model)*
5. Convert constraints into advantages  
*(Physical constraints, Cultural constraints, Technological constraints)*
6. Design for Error
7. When all else fails – Standardize.

Each of these seven principles will be discussed in the following slides.

Discussions on Norman's seven principles.

## 1. Use both Knowledge in the Real World & Knowledge in the Head

- As a basis for his Interaction Model Norman proposed the following levels of abstraction of knowledge of the user:

- Task Level
- Goal Level
- Semantic level
- Syntax level
- Lexical level
- Physical Level

The User models his/her knowledge in/of the world into his / mental realm by the process of cognition.

The user's knowledge model need not necessarily be the same as the knowledge model of the world.

The question is which one should the designer take as reference –Knowledge in the Real World? Or Model of world knowledge in the User's Mental realm ?

Continued....

Relying on either of them alone would lead to an incomplete abstraction of knowledge by the Designer.

Norman's principle mandates that both types of knowledge be considered.

• **Semantic Level** describes a set of objects, attributes and operations, which the 'system' and the 'user' can communicate. Semantics is about how the user interprets and makes meanings out of the system.

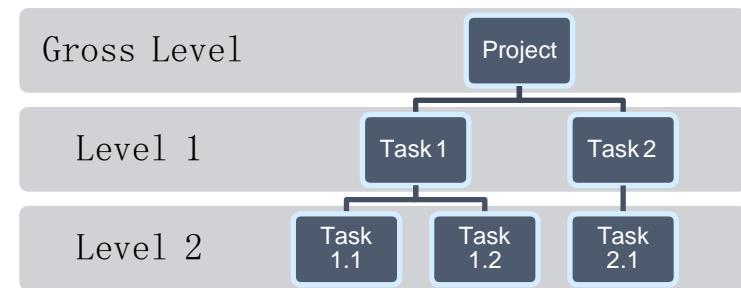
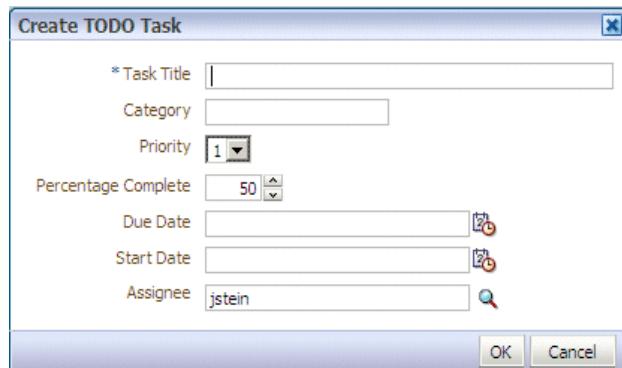
• **Syntactic Level** describes which conceptual entities and operations may be referred to in a particular command context or system state.

• **Interaction Level** describes the translation of commands and objects into the associated physical actions and the structure of the interaction, including typing / mouse / gesture / voice / tactile rules.

## 2. Simplify Task Structures.

- **Task Level:** Task level is to analyze the user's needs and to structure the task domain in such a way, that a computer system can play its part. The task level describes the structure of the tasks which can be delegated to the computer system.
- This principle states that a big 'Task' is to be broken down into (by analysis) their simplest action level (smaller tasks) such that at each level there is as far as possible only one action involved.
- This makes easy mapping with Computer's programming language for the HCI designer to build Interactive systems and Interfaces & Hierarchies.

Ex: Gross level  
and Broken Down  
level of a GUI



### 3. Make Things Visible

Objects at the Semantic level need to be **mapped** to the objects at Syntactic level, for the user. This can be achieved with making the connection as 'Visual' as possible.

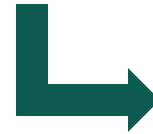
GUI Interface designers use 'Metaphors' to make the connection.

Example:

**Delete** action (at the Semantic Level) (Command Line)



= Waste paper basket to dump (Syntactic level object) (natural language)



= Visual :



Mapping: The link between what you want to do and what is perceived possible.

Continued...

HCI Designers use this principle of 'Making it Visual' to the maximum while designing Interfaces.

Interaction styles such as  
WIMP (Windows – Icons – Menus – Pointer)

Three Dimensional Interfaces as in  
Virtual Reality can be found in current software interfaces.

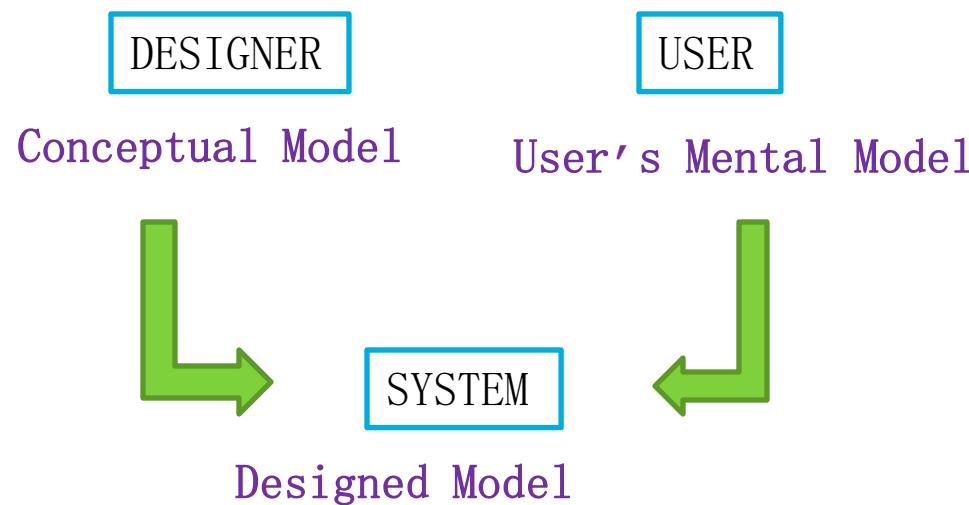


Example of 'Visual':  
Windows 8 Interface.

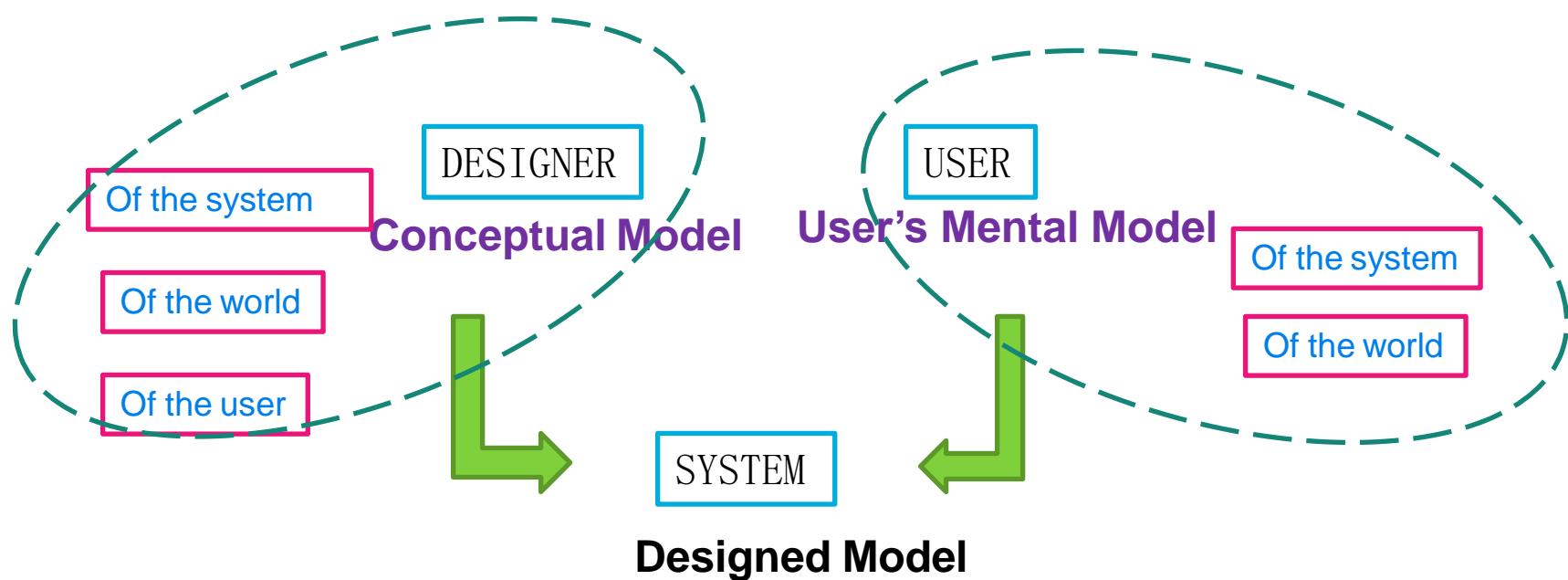
## 4. Get the Mapping Right

Mapping: The link between what the *user wants to do* and what *is perceived as possible - by the user based on the user's own logic*.

**User Mental Model = Conceptual Model = Designed Model**



The three models are elaborated upon in the following slides.

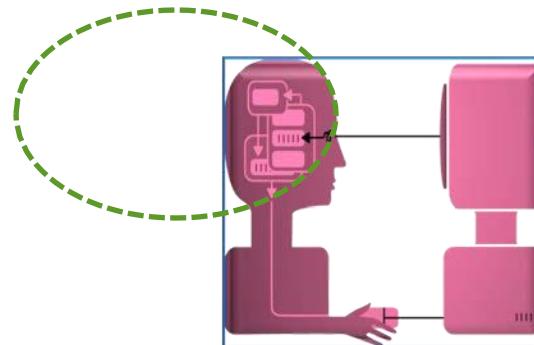


The **User's Mental Model** is the model of a system's working that a user creates when learning and using a computer. It is not technically accurate. It may also be not stable over time. User's mental models keep changing and evolving as learning continues.

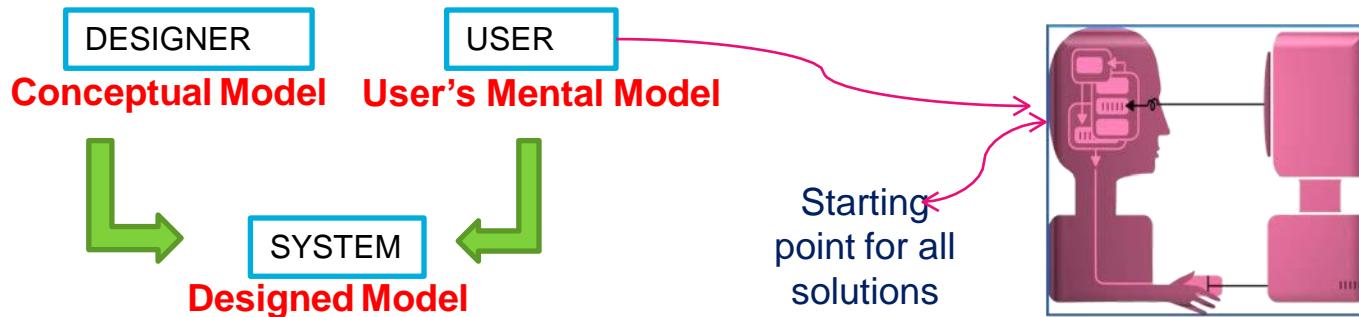
In a way Mental Models are models people have of themselves, others and environment. It is their inner understanding.

The mental model of a device is formed by interpreting its perceived actions and is a visible physical structure. Some times the word 'System image model' is also used to imply the real world physical model.

## The Conceptual Model.



- This is a technically accurate model of the computer / device / system created by designers / teachers / researchers for their specific internal technical use.
- Users too have a Conceptual model but it is their mental model unless the user is as technically qualified as the evaluator. In a way as far as the user is concerned the mental models and the conceptual models are inherent to each other.
- Designer's too have Mental models of the system. So a Conceptual model of the system needs to be as close as possible to the System's Image Model.



A good device / system will emerge when the starting point of the design process is the user- his/her mental model' (i.e. derived through user research - task analysis, Cognitive walk-through, Contextual inquiry etc.) being the basis of the system image and its conceptual model.

Conceptualized solution which Designer had in his/her mind is the "*Design Model*".

The User model (what the user develops in the self to explain the system operation) and the system image (the system's appearance, operation way it responds is usually a blend of the users mental model and conceptual model all rolled into one. (unless the user happens to be an expert (designer))

Ideally, the design model and user model have to be as close as possible for the systems acceptance. The designer must ensure that the system image is consistent with and operates according to the proper conceptual model.

## 5. Convert Constraints into Advantages

This is about how to ensure that the user knows what to do next when there are more than one possibility or more than one given option.

In other words how a designer needs to embed constraints in the sequence of operations in an interface such that the user is guided to the right sequence choice by reducing the chance of error in choosing the wrong option

As a principle Interfaces need to have any of the three type of constraints  
Physical; Technological; Cultural

### **Physical Constraints:**

Based on shape, size, area (example mouse over area demarcation)

## Cultural Constraints:

Culturally and semantically practiced rituals, symbols, color codes.

Ex: Always start from Right & stop on Left lower end in a word document.

**Technological Constraints:** Example: Closing a file without saving, hence user needs to be warned every time this is likely to be operated by the user.

The User need to Program it in such a manner that it is mandatory for the user to press the save button before close button.

## Visibility and Feedback:

Visual design also can suggest constraints.

Ex: If a number of identical buttons are required for diverse functions, Visually building differences such as colour or grouping , it is possible to 'constrain' a user in not pressing randomly the identical looking buttons placed in close proximity.

## 6. Design for Errors

Errors are not taken as human faults in users in HCI.

This means Errors by users cannot be blamed on Users.

Users are not the cause for errors. Often errors are 'slips' - intend to do one thing but end up doing another accidentally.

Errors happen when there is a mismatch between User's mental model, designers' understanding of User's mental model; system limitations.

Research literature reveals that Errors can be classified as:

**Description Errors:** Two objects physically alike are described / taken mistakenly for each other. One solution employed is 'highlighting' the object which is in line of next action so that 'attention' is drawn to that right object from amongst similar looking group of objects.

**Data Errors:** Could be perception errors or selection errors. A solution could be reversal of action without penalty and 'affordance' by the user to correct the error by retracing action steps.

## Associative Action Errors:

Associative Errors are those that involve activating one sequence in place of another and realizing it when wrong/unexpected response results.

Associative Errors happen when short term memory is overloaded or long term memory fails. Forgetting to do something as prescribed or reversing the sequence - Pressing the second button first instead of the first button etc.

- ‘Slips’ & Errors - need to be taken care of in Design by providing feedback (either pre or post action ).

Example : Prompting.

- The cause of the error needs to be understood more than the error.
- Retracing actions must be provided for.
- Assume Task to be imperfect and assume that users will always ‘approximate’ their actions.

## 7. When all else is Unsuitable (Fails) – Standardize.

In certain situations / contexts wherein the nature of the task is critical (mission critical operations), the user needs to be 'forced' to follow the only choice as given (afforded) by the design.

Example of such situations: Medical Devices; Warfare Equipment (Missiles); Nuclear Equipment; Power Plant Controls, Energy Grids; Air Traffic Controls.

In such critical application contexts 'STANDARDISATION' practice is followed.

However very stringent usability testing & evaluation practice is followed before '**standardising**' the format for both INPUT as well as the OUTPUT.

Standardisation comes under the ‘ Best Practice’ adaptation wherein specific rules are the basis;

Where as PRINCIPLES are abstract design rules with navigation and the UCD focus , ‘ GUIDELINES’ allow more freedom to the designer.

Between ‘ Standardisation, Principles and Guidelines, the context and the level of the user (expertise) are the determining factors.

## Conclusions:

Here, we discussed some of the more popular HCI norms such as:

- Norman's Seven Principles of Design were explored in detail in terms of their background, theory and applications in design.

Since the 'design' involves both qualitative as well as quantitative aspects - guide lines, rules & principles are more suited than absolute laws as in the case of science.

## Homework:

1. Chose a Software interface and conduct an evaluation using the Norman's Seven Principles.
2. Draw an 'interaction model' based on Norman's model for the following Interface: Assume all data.

An interface for checking number of Leaves (absence with permission) availed by a student and type (Medical, Vacation; Conference visits;). Refer to the student leave rules of your institution for necessary constraints and other relevant data.

# HCI Guidelines: Norman's Model of Interaction

## Introduction

Let us first understand the word “ INTERACTION”

All man-made objects offer the possibility for interaction.

When an object is designed for a purpose (function) it affords interaction. Interaction is a way of framing the relationship between people and objects designed for them.

Interaction is thus a way of framing the relationship between the object and the User.

All Design activities can be viewed as design for interaction. In fact not only objects but space & messages (communication) too involve interaction. Interaction is a key aspect of function, and function is a key aspect of design.

However often one can easily notice that the designers often use the word ‘INTERACTION’ rather carelessly .

Untrained Designers often tend to confuse ‘Interaction’ with ‘Reaction’.

For example: Designers claim to be designing “Interactive web pages”.

The fact is clicking on links to navigate to a new webpage is NOT an “INTERACTION”. It is the ‘reaction’ of input by the hyperlinked pages. The computer is automatically reacting to input because it has been programmed to do so. This programmed action couples the ‘input’ to ‘output’ in a fixed way.

Interaction is however a dynamic action that through a dialogue (involving the feedback) adjusts to input and gives appropriate output.

In HCI – the feedback loop model of interaction treats a person as closely coupled with a dynamic system.

In HCI – Interaction is simply stated as two way communication between the “User and System”



It should however be noted that due to the human complex cognitive system, representing interaction between a person and a dynamic system as a simple feedback loop can only be a good first approximation.

## Definitions of some Terms of Interaction

**Domain:** It deals with the expertise, knowledge of a some real world Activity (Environment). In GUI domain concepts such as geometric shape, colour, Symbols etc. are involved.

**Task:** It is the operation to manipulate concepts in a domain.

**Goal:** It is the desired output from a performed task.

Example in GUI: A button

**Intention:** It is the specific action required to meet the goal

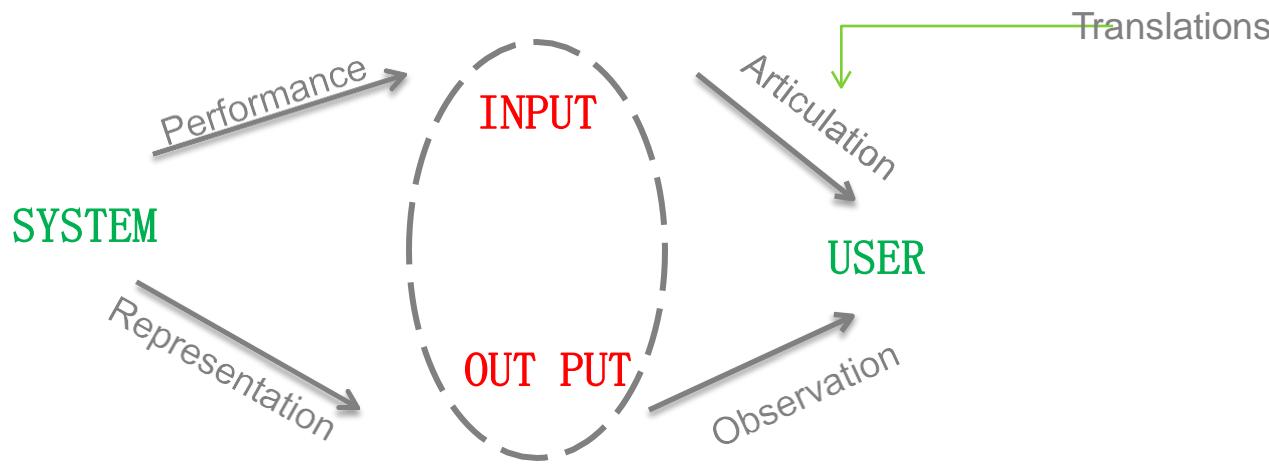
**Task Analysis:** It is the study of the problem space

In HCI interaction models are translations between user and system  
**There are different Interaction Models mentioned in HCI**

- Donald Norman's Interaction Model
- Abowd & Beale's model

A generalised Interaction Model (from Dix et al) has four components:

- (i) System; (ii) User; (iii) Input & (iv) Output.



There are different Interaction Styles (nature of the dialogue)

And there are different Interaction Contexts  
**(Social, Organizational, Educational, Commercial etc.)**

We will discuss Donald Norman's Interaction Model in the following slides

# Norman's Model of Interaction

Donald Norman's Interaction model concentrates on the Users Thought processes and accompanying actions.

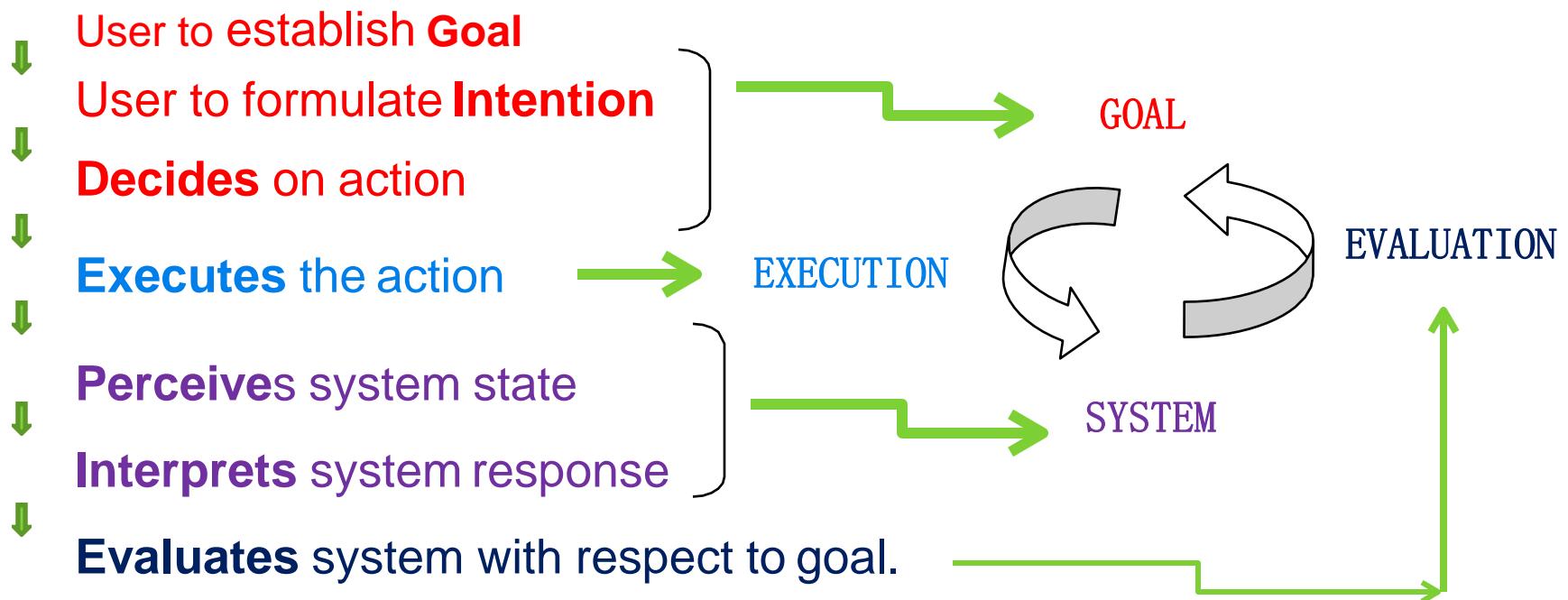
Norman proposed that actions are performed by the users in cycle such as

- (i) Establishing a Goal
- (ii) Executing the Action
- (iii) Evaluating the Results

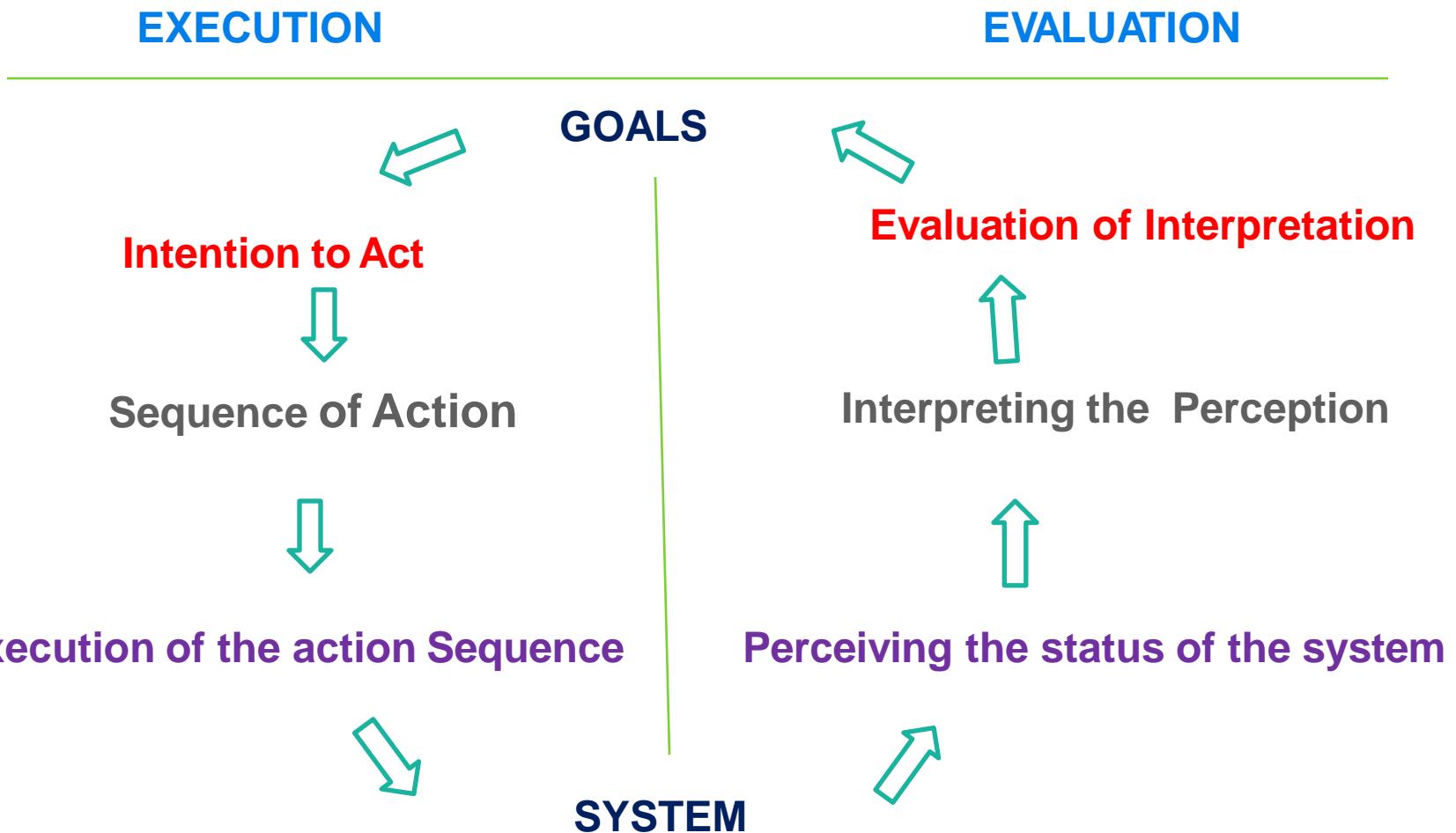
Given a need a user sets about achieving the goal of fulfilling the needs.

A series (sequence) of actions are performed – one leading to another – till the expected results are obtained.

## Norman's Model of Interaction consists of SEVEN STAGES as follows:



# Another way of depicting Norman's 7 stage Action model

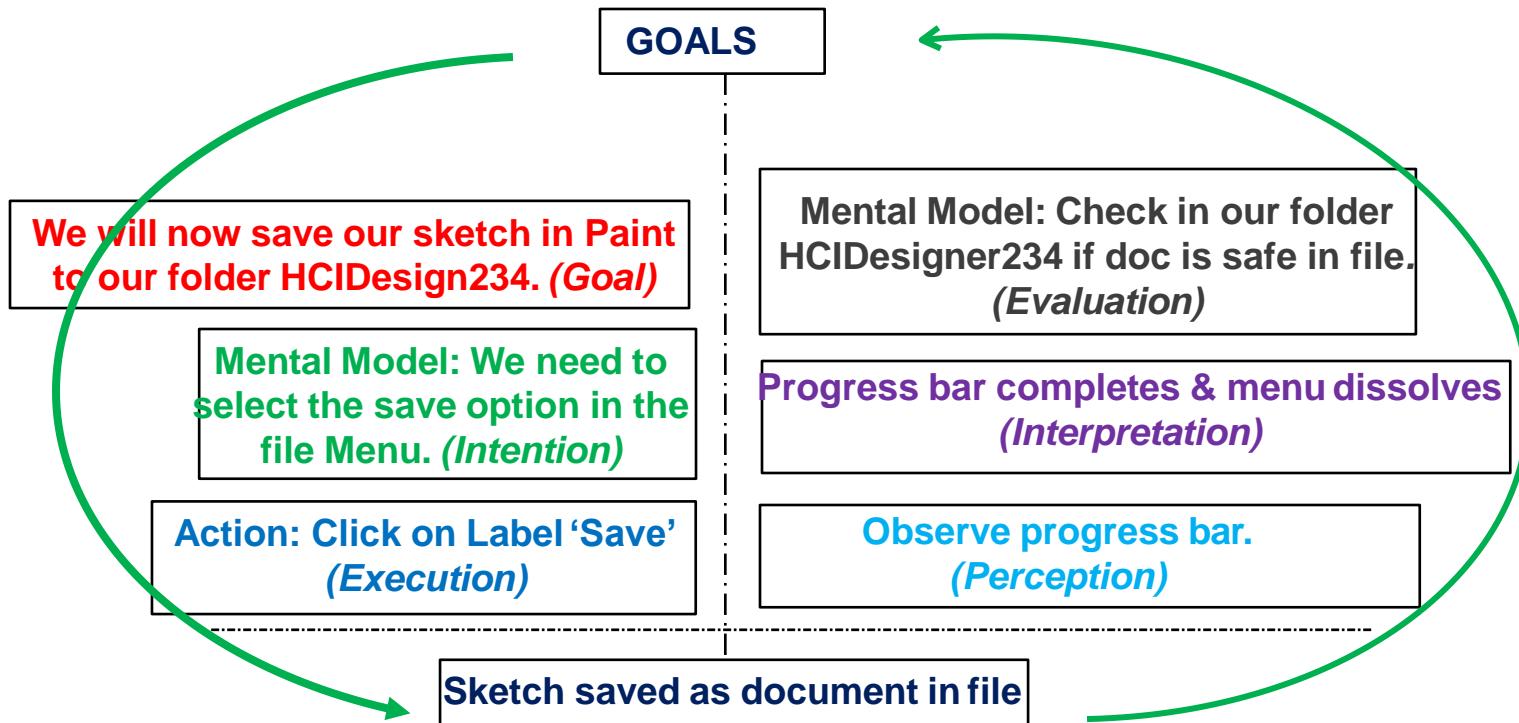


# Understanding Norman's Model with Example:

Need: Documenting work done

Task: Save Our Sketch

Goal: Safely store the sketch in a place which we can fetch it from



As a basis for the Interaction Model, Norman proposed the following **levels of abstraction of knowledge of the user**

- **Task Level**
- **Goal Level**
- **Semantic level**
- **Syntax level**
- **Lexical level**
- **Physical Level**

**Task Level:** task level is to analyze the user's needs and to structure the task domain in such a way, that a computer system can play a part in it. The task level describes the structure of the tasks which can be delegated to the computer system.

**Semantic level** describes the set of objects, attributes, and operations, the system and the user can communicate. Semantics is about how the user interprets it and makes meanings out of the system.

**Syntax level** describes which conceptual entities and operations may be referred to in a particular command context or system state.

**Lexical level:** language, wording.

Norman's HCI model consists of three types:

### User's Mental Model; System-Image Model; Conceptual Model.

The User's Mental Model is the model of a machine's working that a user creates when learning and using a computer. It is not technically accurate. It may also be not stable over time.

User's mental models are dynamic as they keep changing, evolving as the learning continues.

In a way Mental Models are models people have of themselves, others and environment.

The mental model of a device is formed by interpreting its perceived actions and its visible structure.

The System-Image Model is the visible physical part of the computing system / device.

## The Conceptual Model.

This is the technically accurate model of the computer / device / system created by designers/teachers/researchers for their specific internal technical use.

Users too have a Conceptual model but it is their mental model unless the user is a technically qualified as the evaluator. In a way as far as the user is concerned mental models and conceptual models are inherent to each other. Designer's too have Mental models of the system. So a Conceptual model of the system needs to be as close as possible to the System's Image Model.

The User model (*what the user develops in the self to explain the operation of the system*) and the system image (*the system's appearance, system's operation way it responds*) is usually a blend of the **users mental model and the conceptual model** all rolled into one.

## Interaction Model and Device / System Design

A good device / system will emerge when the starting point of the design process is the user's mental model' (in turn derived through user research-task analysis, Cognitive walk-through, Contextual inquiry etc.) being the basis of the system image and its conceptual model.

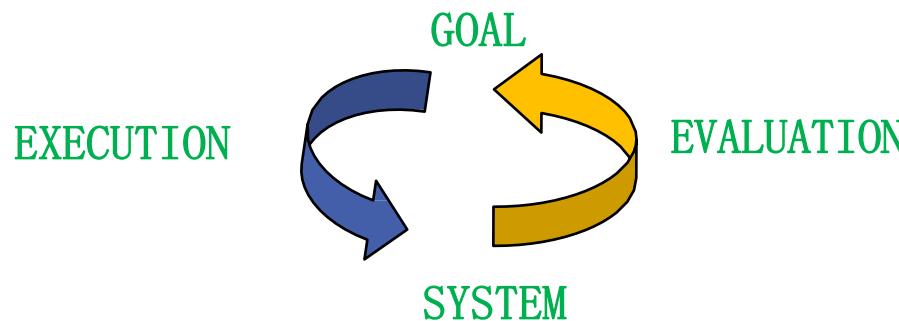
The Conceptualisation of the Designer had in his/her mind is called the "*design model*".

Ideally, the design model and user model have to be as close as possible for the systems acceptance.

The designer must ensure that the system image is consistent with and operates according to the proper conceptual model.

**Norman applies the "Gulf Model" to explain why some interfaces cause problems to the users.**

Norman uses the terms "**Gulf of Execution**" and "**Gulf of Evaluation**". Norman's model (also sometimes referred to as the Gulf Model) is useful in understanding the reasons of interface failures from the user's point of view. The Seven stages of action model is an elaboration of the Gulf model.

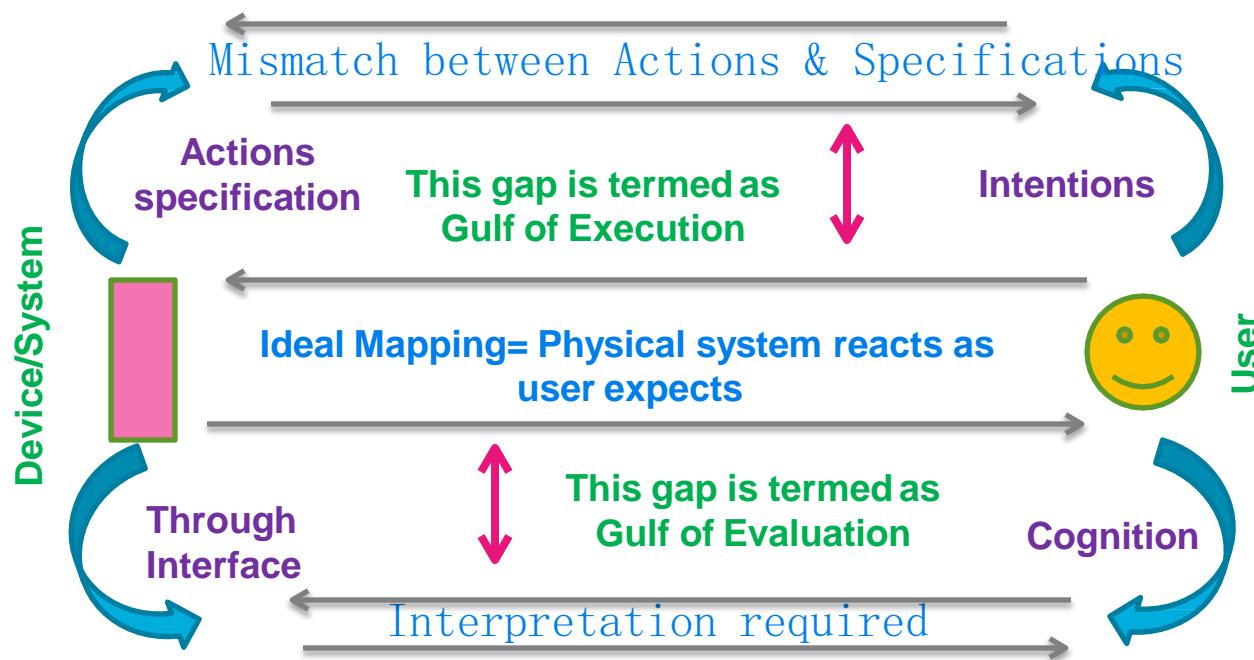


**Gulf of Execution:** It is the difference between user's formulation of the action to reach their goals and the actions allowed by the system.

User's formulation of action     $\neq$    Actions allowed by the system.

**The Gulf of Evaluation** is the difference between the physical presentation of system state and the expectations of the user.

User's Expectation     $\neq$    System's presentation.



# Interaction Styles

Having understood Interaction Frame work as a model let us  
Look at different Interaction Styles

Some common interaction styles

- Command line interface
- Menus
- Natural language
- Question/answer and Query dialogue (Ex: SQL)
- Form-fills and spreadsheets
- WIMP – [Windows; Icons; Menus; pointers]
  
- Three-dimensional interfaces
- Gestural Interfaces
- Voice operated commands
- Thought (mind) operated commands (Evolving rapidly)

## Conclusions

Interaction models are conceptualisations of the process of Interaction between the user and the system.

Norman's seven stage Interaction model explains interactivity from the user's point of view.

There is a gulf (gap) between (i) EXECUTION and (ii) EVALUATION

There could be a number of reasons why an interaction can fail at a number of points in the dialogue.

The interaction model can be a useful tool for analysing as well as conceptualising dialogue between a system and user.

## Homework:

Draw the Users Mental Model for a Transfer of Money from one account to another on an ATM

Using Norman's seven principles draw a Norman's Interaction Diagram for 2 Tasks in any application software of your choice.

## References:

Shneiderman; Designing the user Interface: strategies for effective human computer interaction. Addison –Wesley, Reading MA 1987.

Nielsen, Enhancing the exploratory power of usability heuristics.  
Proceedings of the ACM CHI'94 Conference.

D. A. Norman; The Design of Every Day Things. Doubleday, New York1988.

Dix.A, Finlay J; Abowd G.D & Beale R; Human Computer Interaction, 3<sup>rd</sup> Edition,  
Pearson Education 2005.

# **HCI Guidelines:**

## **Overview of Nielsen's Ten Heuristics and**

## **How to Conduct a Heuristic Evaluation**

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# **HCI Guidelines:**

## **Nielsen's Ten Heuristics**

## Learning Objectives:

- In this lecture module, we will introduce another set of well known interface design guidelines proposed by Jacob Nielsen.
- The application of these design guidelines to specific situations like a website will be discussed in the background of UCD framework.

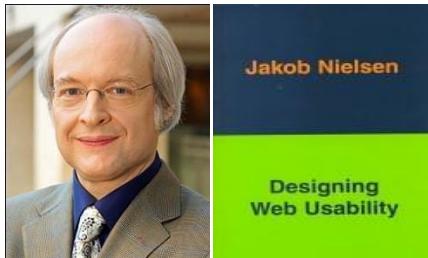
# Introduction

Jakob Nielsen\* (working along with Molich in 1990) proposed a set of ten guidelines that can be used as Principles of Designing a new Interface. These guidelines can also be used as Heuristics for evaluating an Interface.

Since these ten guidelines were more in the spirit of “Rules of Thumbs” than the specific rules, they are referred to as ‘Heuristics’ rather than rules or laws that hold true in every case.

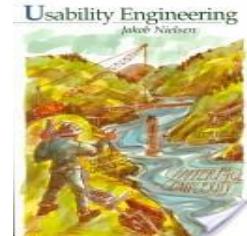
**A heuristic is an approach to the problem solving or self discovery that employs a practical method that does not guarantee an optimal, perfect or rational solution, but produces a near optimal or an approximate or a short-term solution. Thus, heuristics can be mental shortcuts that ease the cognitive load of making a decision. Examples that employ heuristics incl. using a trial and error method, a rule of thumb or an educated guess.**

**Heuristics in AI:** A **heuristic** function, also called simply a **heuristic**, is a function that ranks alternatives in search algorithms at each branching step based on the available information to decide which branch to follow. For example, it may approximate the exact solution.



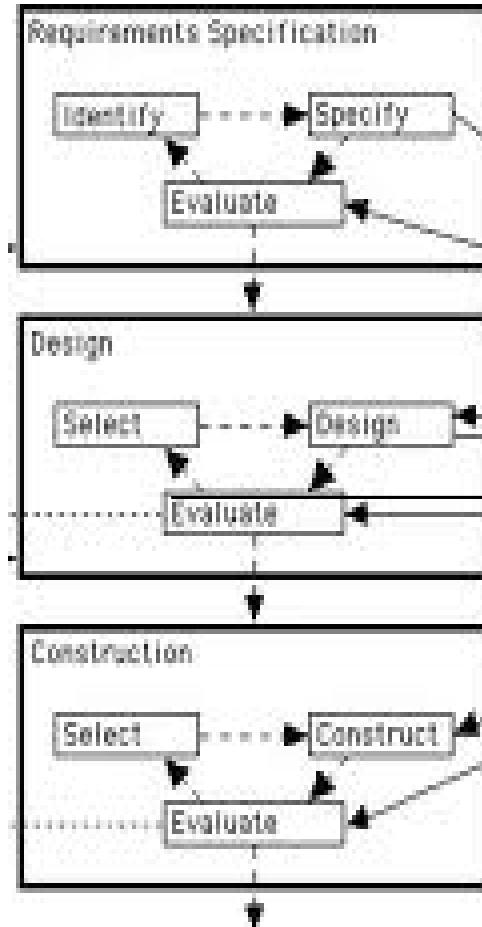
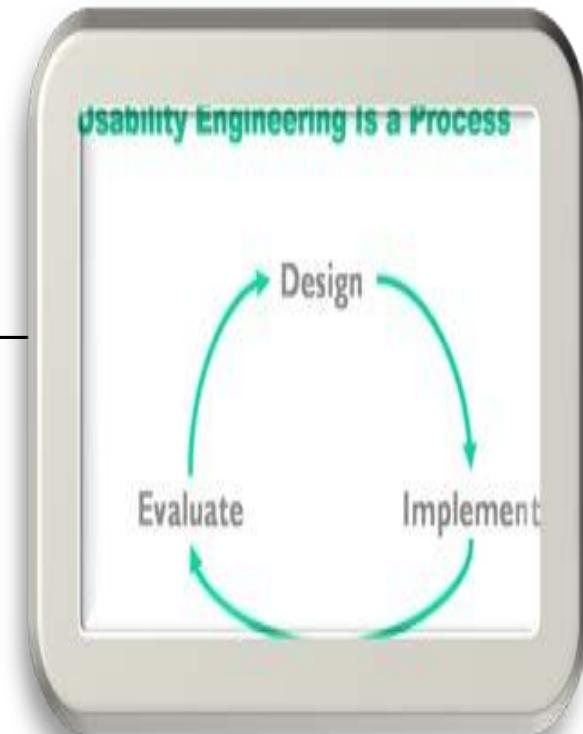
Jakob Nielsen is a leading Web Usability Consultant. He holds a Ph.D. in Human-Computer Interaction from the Technical University of Denmark in Copenhagen.

He has authored many books in the areas of Usability, HCI, and Experience Design. His book titled “Usability Engineering” 1993 is a textbook on methods to make interfaces easier to use.



Usability Engineering involves User Research; Design Research and Validation of Design through Construction & User Testing.

In some institutions it is taught as an independent discipline while in others it is part of HCI discipline.



# Introduction

Heuristics means “Rules of the Thumb”.

These Ten ‘Rules of the Thumb’ were derived after careful research by Nielsen who after conducting a factor analysis of 249 usability problems, came up with ten simply stated guidelines in 1994.

Nielsen’s heuristics are empirically based derivations. Widely used by the Usability professionals (including the Interface designers), and they quickly identify likely interface design problems in an application.

Method suggested by Nielsen is popular because of its simplicity and low cost. It is preferred evaluation technique at the preliminary design stages by the HCI professional.

# Nielsen's Ten Heuristics

- 1. Visibility of System Status**
- 2. Match between System and the Real World**
- 3. User Control and Freedom**
- 4. Consistency and Standards**
- 5. Error Prevention**
- 6. Recognition rather than Recall**
- 7. Flexibility and Efficiency of Use**
- 8. Aesthetic and Minimalist Design**
- 9. Help Users Recognise, Diagnose, and Recover from Errors**
- 10. Provision of Help and Documentation**

Each principle will be explained in the following slides.

# Visibility of System Status

*Users need to be kept informed by the system about what is going on, through appropriate feedback within reasonable time.*

**Elaboration:** This means the user needs to be constantly made aware of his/her interaction with the interface while interacting. The control response ratio (input to output time) need to be as small as possible. Any interface needs to communicate such that it is in a ready state to be operated upon – at the start of an interaction cycle.

For Example:

A glowing LED or flashing element indicating that the interface is live.

An animated symbol that states that 'saving' act is going on.....



**Most important to users is to know "Where am I?" and "Where can I go next?" Internal reference is a must to feel in control.**

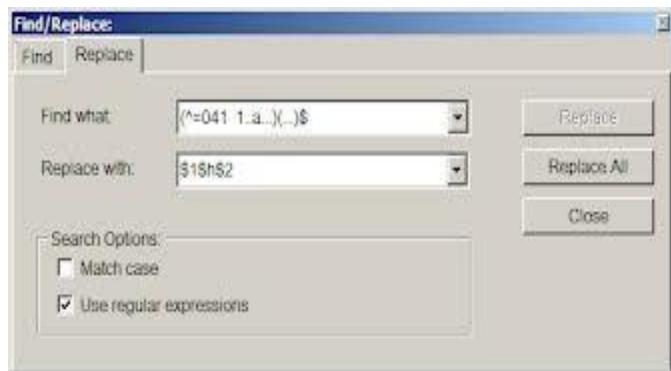
# Match between System and the Real World

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

**Elaboration:** Technical jargon or using terms like 'Initiate' or 'Load' in place of 'Start' contributes to initial mismatch between the users cognitive process and machines feedback dialogue.

An interface need to allow smooth transition from contextual 'reality' world to artificial machine world. ....in other words from ' reality' to 'digitality'.

Tendency to use the programming language and syntax on the display, while understandable to the software programmer, will certainly be a mismatch to a user.



Users are from different backgrounds, skills levels, specializations & culture.

The context on the screen needs to match with the context of the user's mental model

# User Control and Freedom

*Users often choose system functions which they did not want. (Mouse click due to haste). This calls for Support undo and redo.  
A user need to go through tracing too many steps back to regain control.*

**Elaboration:** Sequential thought process in a user that follows a simple everyday human habit need to be reflected in the dialogue between the user and the device. A good interface facilitates this.

Being in control implies that one can choose to stop interacting on time rather than be forced or trapped by the interface into inaction. Feeling in the user that he/she is in full control of the system at all times must be created. If the user attempts to gain full control of the system and if a message like 404 error occurs then the system is unfriendly & unhelpful!



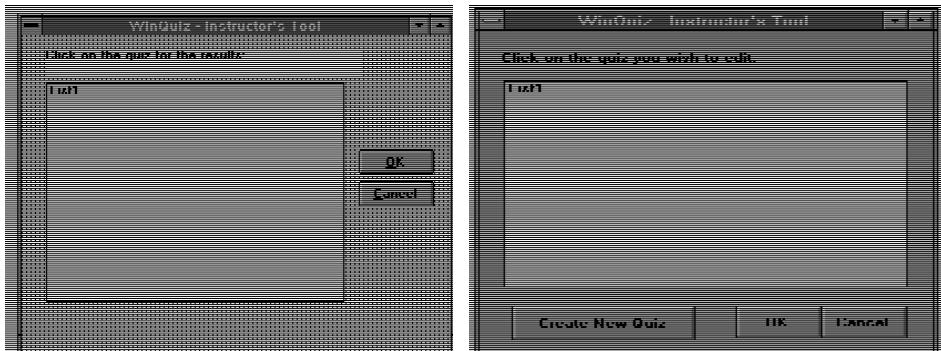
Can users select and sequence tasks? Can they easily return to where they were if they choose an inappropriate / action path? The first example “accuses” them of committing an error. The second example is much better but does not tell the user what to do next! The third example is inappropriate!

# Consistency and Standards

*Using different words to mean the same action or using different symbols on different pages can be confusing to the user. Users should not have to wonder whether different words, situations, or actions mean the same thing. They should not be in doubt as to what to do next.*

**Elaboration:** Within an interface if multiple words or actions are used to mean the same thing, it can only lead to the confusion in the user due to perceived lack of consistency. Interaction pattern gets disrupted. When pattern becomes complex, user's cognitive load increases.

**Consistency in dialogue as well as in visual elements is achieved by specifying and adhering to a dictionary of words / labels / symbols/ colors which together form a 'standard' – a prescribed set – compulsorily to be followed.**

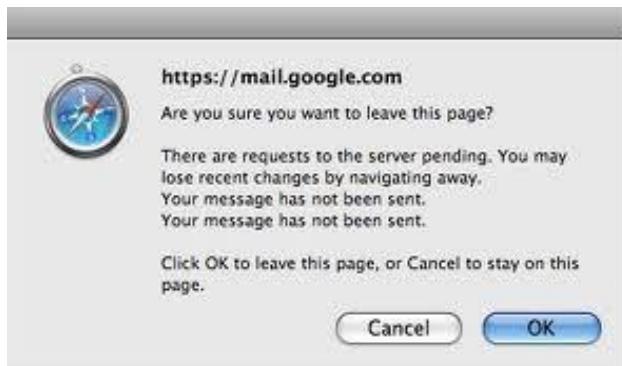


Inconsistent wording & windows / buttons can confuse users when the destination page has a different title from the link. The two screens belong to the same software but appear differently at different places within the website.

# Error Prevention

*By research it is possible to pinpoint the typical errors that users normally tend to commit. Prevention of error is the best approach. However recovery from error prone actions through a well designed error message should be adopted.*

**Elaboration:** To err is human. Errors can happen regardless the level of expertise of the user or familiarity of the interface. A good principle of design is to seek out error prone interactions, build in error prevention within the dialogue. Forewarning, restricting, prompting, retracing or recovery routes, etc. are means of addressing errors. Errors will lead to a situation wherein the users feel subdued by a machine. Anticipating for errors and incorporating preventive measures ensures fear free and ego free user thereby giving importance to 'H' in HCI through 'I'



GUI-style widgets cut down on the errors but may still have to be double checked before confirmation

# Recognition rather than Recall

*Loading the short term memory of the user beyond a limit has negative consequences. Given a navigation path, a user need not to remember or recall all the instructions. Users are better at **recognizing** things they have previously experienced. Prompts, visibility, sequential direction, pop-ups etc. should come to the aid of the user. Help needs to be easily retrievable.*

**Elaboration:** Reduction on cognitive load during the interaction ensures that the user is not asked to rely on means and methods that extract human cost. If an interface requires specialized training and use of memory to operate, then it will be quickly abandoned by the user.

Analogy, metaphor, symbols, sounds, etc. are used as design elements in an interface to ease recall thereby eliminating the need for ‘thinking while interacting’ and memory loads for the user.

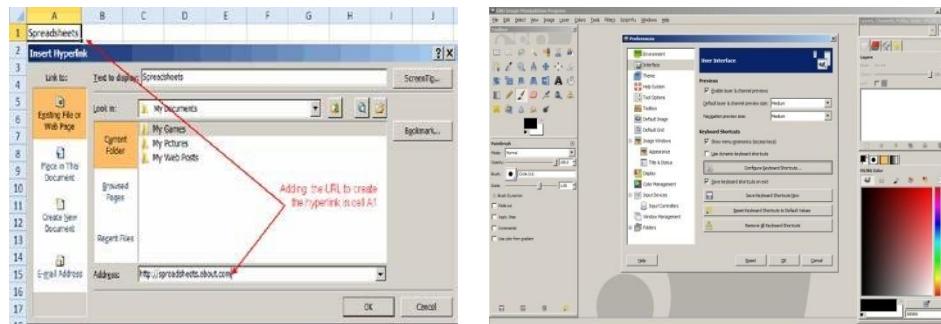


Good labels & the descriptive links are crucial for recognition. The first two icons as shown in the Fig. are difficult for the user to recognize or to recall. The third one helps the user to recognize where they are and recall which file is currently open.

# Flexibility and Efficiency of Use

The system can cater to both inexperienced and experienced users. As the user becomes proficient - shortcuts can be encouraged. Thereby increasing the efficiency. Allowing the rearranging of the screen elements by the user can also be adopted.

**Elaboration:** Once a user becomes adept at using an interface, he/she upgrades into a higher level user from a novice. Such users will always seek to complete the task faster. Such users seek out shortcuts. An interface need to allow this. It needs to be flexible and make it possible for the user to adopt quicker dialogues through shortcuts. The user feels efficient as well as proficient. The feeling of mastering the software is a flexible sign of being in control thereby.



Advanced users (experts) can opt for shortcuts in the spreadsheet example.

Flexibility of keeping the required no. of buttons / sections in view or hiding them gives the option to the user to rearrange GUI as needed as shown in Fig. 2.

# Aesthetic and Minimalist Design

*Relevancy, simplicity, minimum amount of labels, uncluttered graphics result in efficient communication dialogue between the user and the interface. All unnecessary superfluous elements need to be dropped.*

**Elaboration:** Visual clutter in the interface only adds to inefficiency however impressive it is visually.

Simplicity is equal to the efficiency which is equal to the elegance which is equal to beauty that is the aesthetic algorithm in minimalism. Use of least number of elements (minimalism) is more 'scientific' rather than 'artistic'.

**Visual noise needs to be completely eliminated.**



# Help Users Recognize, Diagnose & Recover from Errors

*Preventing a user who is about to make an error would be a good approach. Gentle wording of error messages, constructive suggestions, re-educating the user- can contribute towards a happy self-confident user who is not afraid of being caught unawares or penalized.*

**Elaboration:** No body likes to be loudly informed that he/she has erred. Error messages need to be disused as suggestions / prompts and precise instructions so as to be able to correct the error and recover. The learning component in errors so that the user recognizes the error as it is being made, or recognizes the reason why the error happened in the first place – helps the user learn.



There is no way to understand the consequences of canceling. The onus seems to be on the user who will be held responsible for whatever is opted for. The proper diagnosis & how to possibly recover is not clear. Very unfriendly interface.

# Help and Documentation

*Even though it is better if the system can be used without documentation, it may be necessary for the user to provide the help and documentation. Help queries need to be answered promptly without the user to go through an elaborate eliminating list.*

**Elaboration:** This again is to assist the user *learn* and understand the dialogue between the user and the machine or understand - where what went wrong - or aid recall during memory-lapses due to long usage time gaps. Adequate 'Help' support system when the user wants and at the point where the user wants it – hence it is a good principle of Interface design.



The screen shots attempt to Train the user by offering information on the consequences of decision

# Conclusions:

- These ten heuristics of usability help in refining a potential design into a good design. These ten principles will ensure that interfaces evolve in the right direction.
- These rules of the thumb act a check list to evaluate a design.
- They also can be used as check list while evaluating any GUI.

## Homework:

You are asked to choose any Interface of a device or a website and conduct an audit to identify where the Nielsen's ten rules have been (i) adhered to (ii) not adhered to.

Further, you are asked to suggest relevant corrections.

# **HCI Guidelines: How to Conduct a Heuristic Evaluation**

## Learning Objectives:

To understand the process of evaluation using the Nielsen's ten principles of Heuristics.

To employ the Nielsen's ten principles for evaluating an interface.

# Introduction

Heuristics evaluation is a systematic process of inspection of a user interface for the usability problems. It is both - “**before design finalization’ predictive method** - as well as an ‘**after design’ evaluation and rating method.**

The goal of heuristic evaluation is to find the usability problems in design so that they can be attended to, as an integral part of iterative design processes.

Heuristic evaluation method involves having a small set of evaluators (5 to 7) examine the interface and judge its compliance with recognized usability principles such as Nielsen’s ten Usability Principles.

# Nielsen's Ten Heuristic Principles

- Visibility of System Status
- Match between System & Real World
- User Control & Freedom
- Consistency & Standards
- Error Prevention
- Recognition rather than Recall
- Flexibility & Efficiency of Use
- Aesthetic & Minimalist Design
- Help, Diagnosis & Recovery from Errors
- Documentation & Help

**Nielsen's ten points aid as a check list for the heuristic evaluator to audit a interface/application/product.**

According to Nielsen, the ten heuristic principles help in identifying and explaining problems. Other researchers have added to the above list of principles.

**A framework of Usability principles is also used for conducting the heuristic evaluation.**

Heuristic evaluation is performed by having each individual evaluator inspect the interface alone.

Only after all evaluations have been completed are the evaluators allowed to exchange & discuss and have their findings aggregated. This procedure is important in order to ensure independent and unbiased evaluations from each evaluator.

The results of the evaluation can be recorded either as written reports from each evaluator or by having the evaluators verbalize their comments to an observer as they go through the interface.

Heuristic reviews are less expensive and less time consuming to conduct the evaluation.

Heuristic evaluation can be accomplished using only a simulation prototype or mock-up as a complete finished product is not necessary. Even wireframes suffice.

Examples of how Heuristics analysis is conducted are presented in the following slides

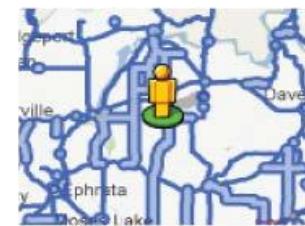
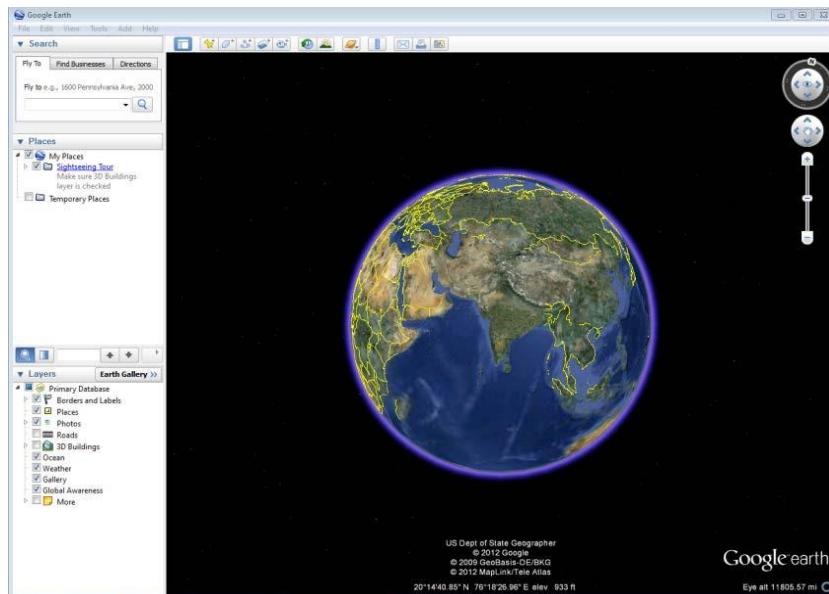
# Case Study 1 GOOGLE MAPS in Goggle Earth

Evaluator: Expert User

Heuristics Used: Nielsen's Ten Heuristics

Google Maps is a well known free service provided by Google world wide.

It's not just a bunch of maps, it includes multiple layers: roads, terrain, satellite, the street view, traffic etc. It also integrates the user ratings and pictures with locations and businesses in the area.

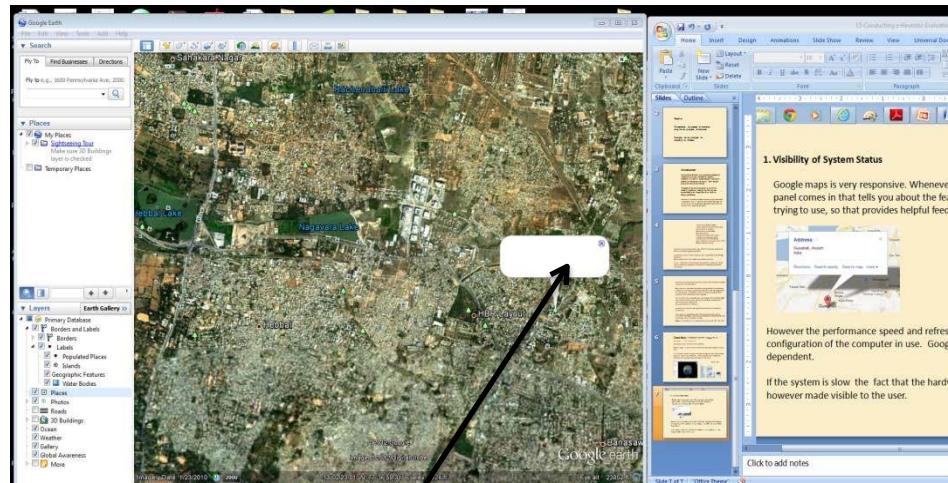


Controls & Views

# 1. Visibility of System Status

*Findings of the expert on visibility & status capability of Google earth is explained below*

“Google maps is very responsive. Whenever you click a button, a panel comes in and tells you about the feature that you're trying to use, so that it provides the helpful feedback.”

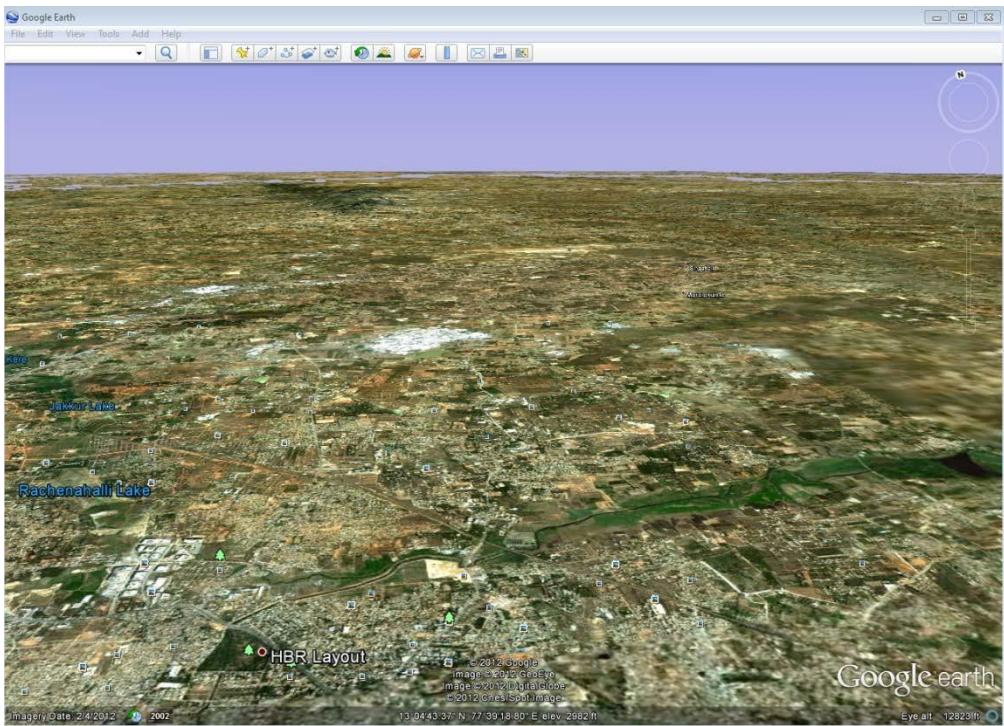


“System status if the Network connection is absent or suddenly is lost is not made visible to the user. The user sees a blank label” (second screen shot)

“However the performance speed and the refresh rate depends on the hardware configuration of computer in use. Google is very CPU intensive and RAM dependent.

If the system is slow due to the hardware mismatch this fact that the hardware is not optimum - is not however made visible to the user”.

## 2. Match between System and Real-World



Closeness to Real-world is very good.

The window can be panned and horizon can be lowered giving a very Real-World view.

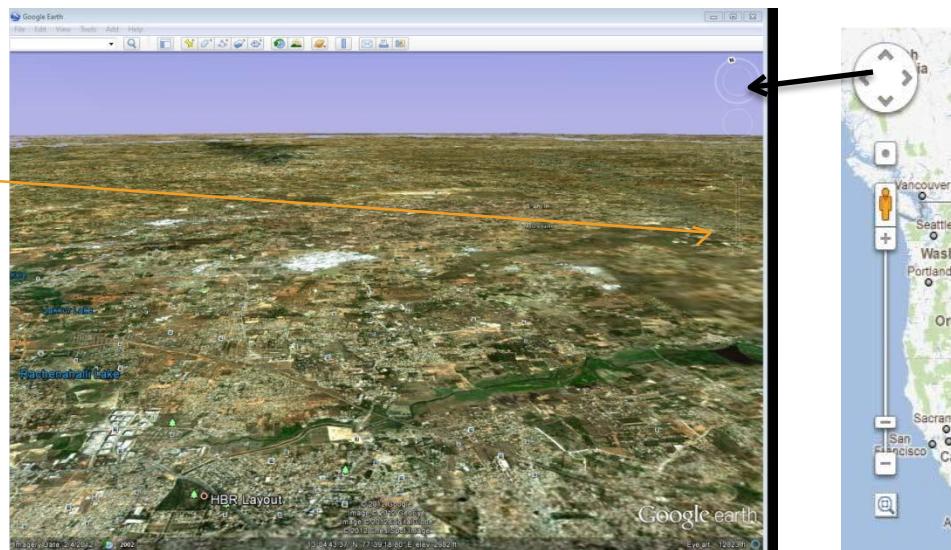
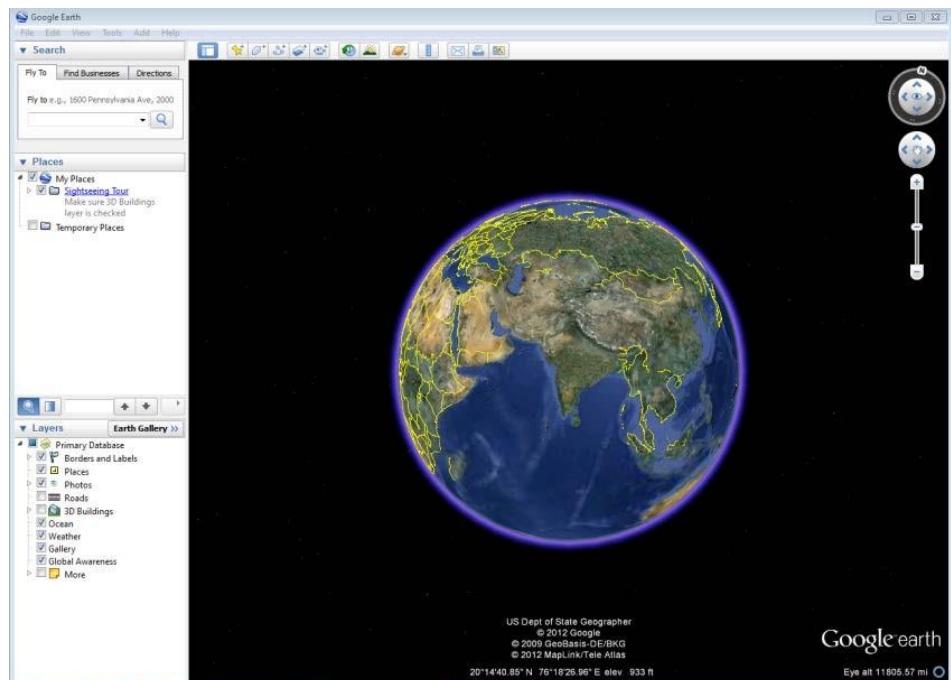
### 3. User Control and Freedom

Almost all the features are available as checkboxes. When they are checked, those items are added to the map and when they're not checked they go right back off the map.



The zoom in controls are fairly intuitive. They recede into the background and come alive when mouse hovers on it. The direction ring gives full control to the user.

Freedom for the experienced user but for a novice a disappearing zoom slider bar can be confusing!



## 4. Consistency and Standards

Google maps is pretty consistent with the words and phrases that they use.

Symbols are pretty clear and a user could figure it without even needing the labels.

Successive screens maintain the consistency in continuity in terms of how & where tools and the buttons appear.

The response of various interactions is standardized across the entire application.



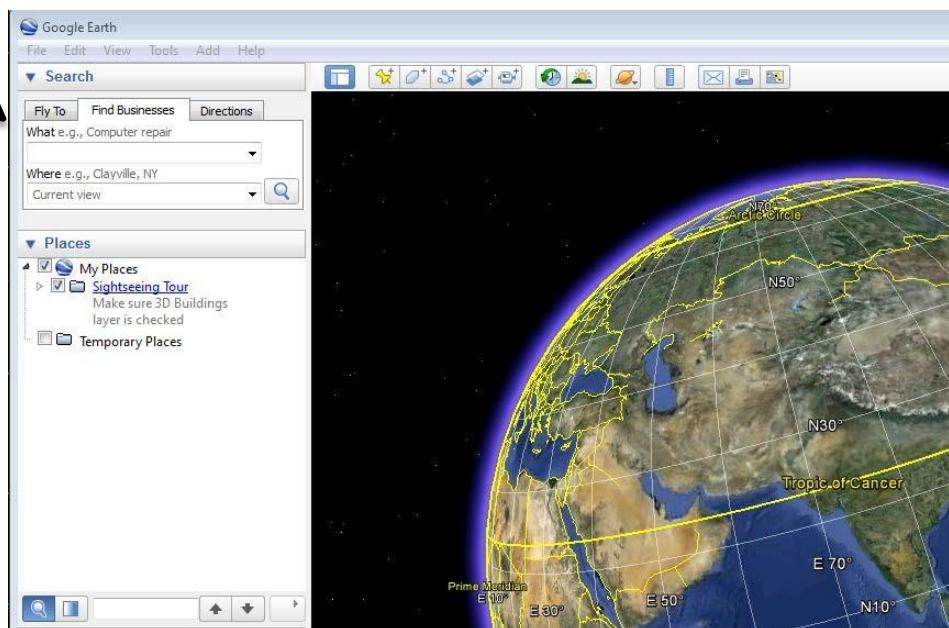
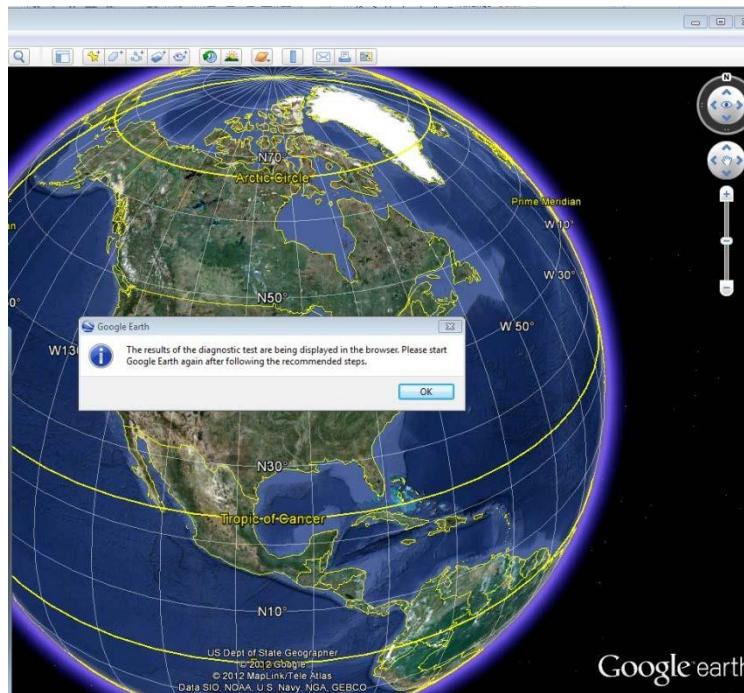
Controls & Views



## 5. Error Prevention

There really isn't much error involved in a mapping program.

There is a possibility that a user may enter the wrong address, but Google maps is well configured to automatically decipher what the user may have been looking for and presents them with a list of options that could be correct.



This not only prevents the users in making accidental errors but also suggest corrections for the user.

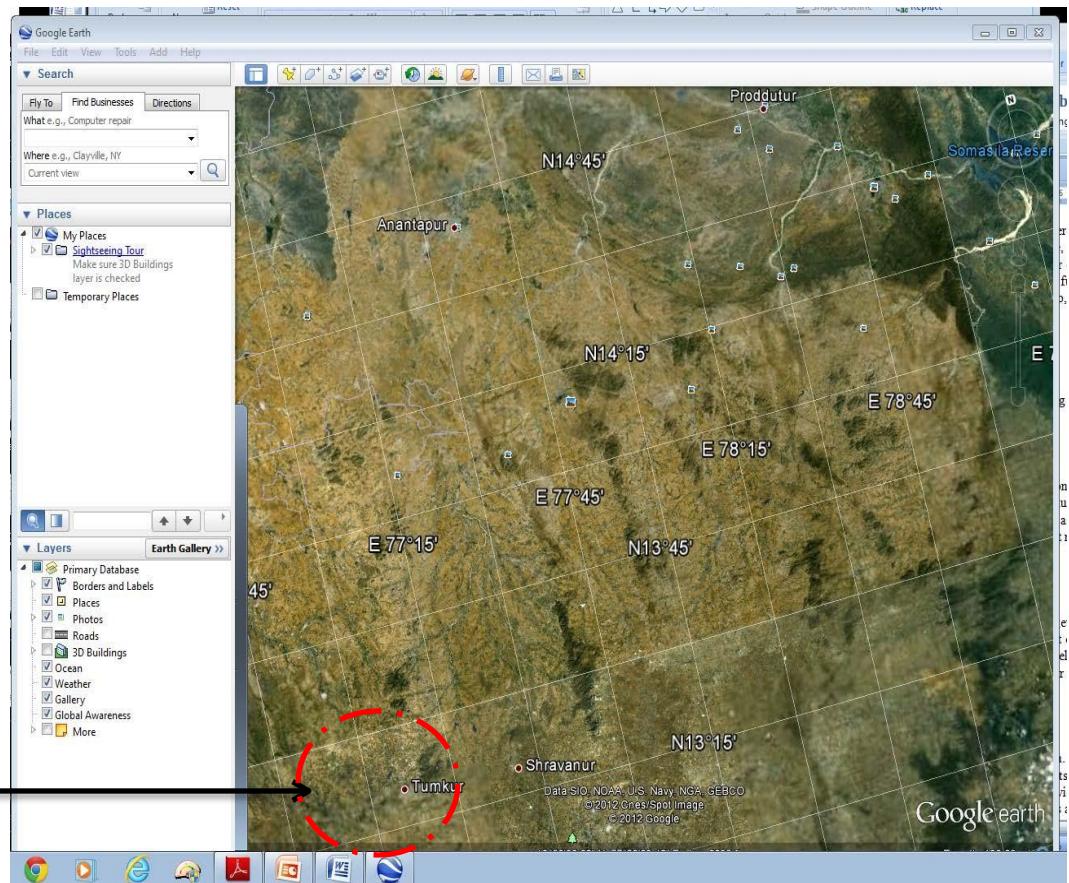
## 6. Recognition Rather than Recall

While most of the more useful functions are visible, but a user must click a button to get directions.

Navigating by panning/zooming often leads to being lost. Users often want to know the direction they need to pan.

Ex: On the right TUMKUR is visible. For a user to go to BANGALURU and (if the user is not sure where Bangalore is w.r.t. Tumkur - East, West, North, South) or is not familiar with the latitude - longitude of Bangalore – will have to recall (or)

recognise or resort to trial and error by first zooming out. Zooming out also leads to disappearing of small towns like Tumkur -. At lower zoom levels the user will have to ‘RECALL’ which is not an easy thing to do on a map of an unfamiliar geography.



It is here that recall is required or in order to become aware (recognize) where in the map one is with respect to overall map.

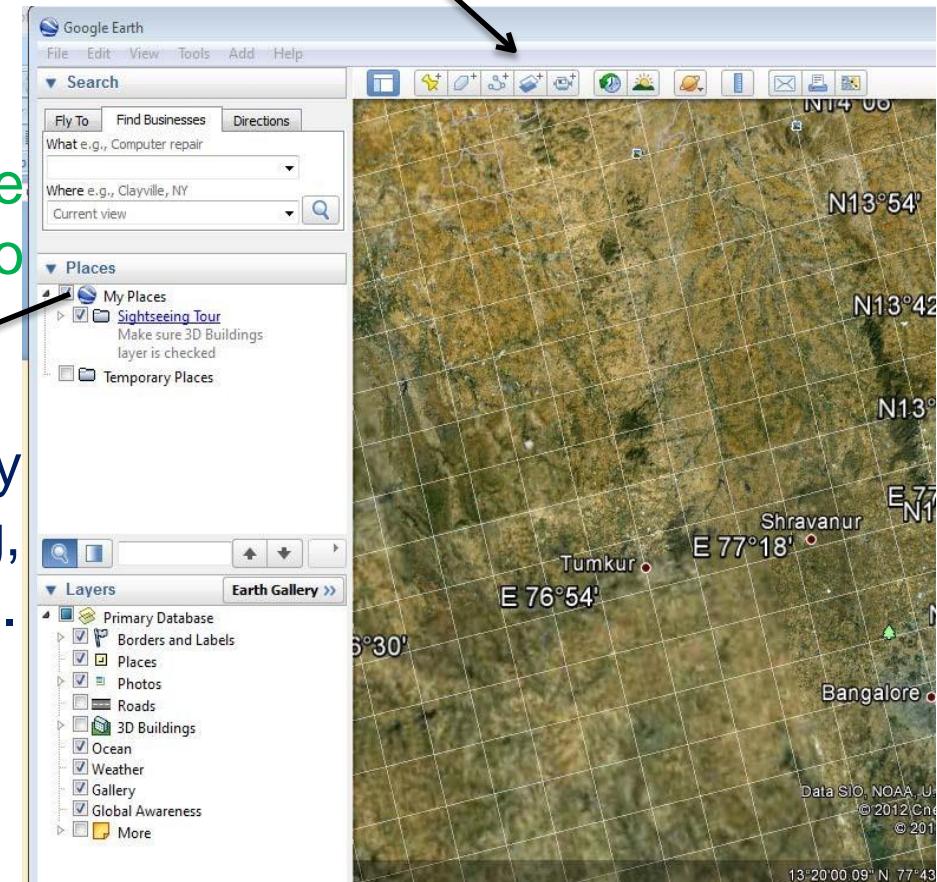
## 7. Flexibility and Efficiency of Use

The Google Earth is highly flexible & efficient. Even if a GPS connection is not available

A user can set a reference location “starting point” and also key in where one wants to go –

Software map navigates itself by either panning, zooming, turning, and makes the destination visible.

The buttons on the task bar reflect the flexibility that is built in.

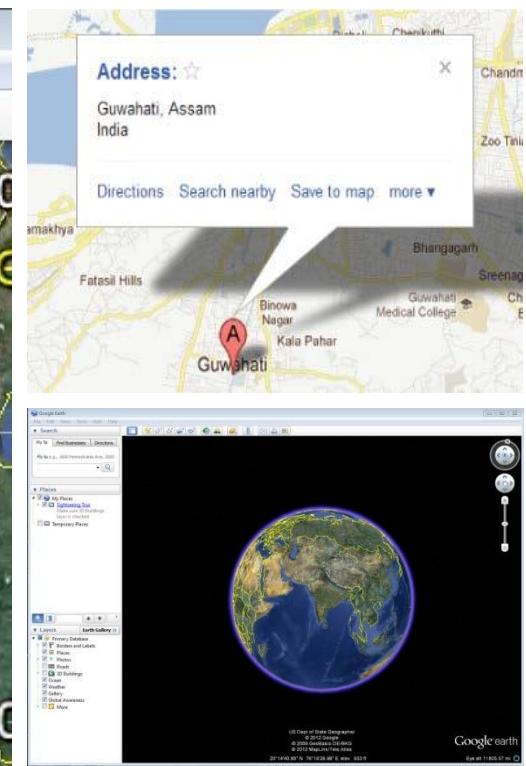
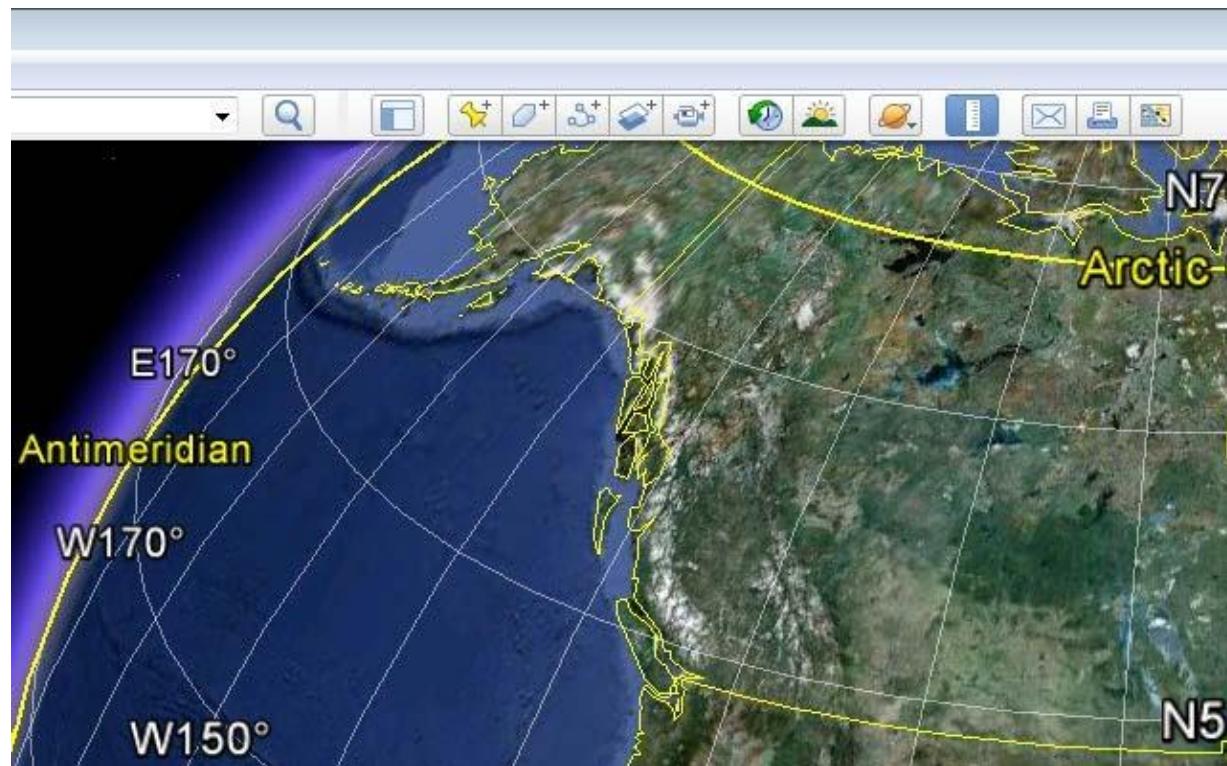


## 8. Aesthetic and Minimalist Design

The entire user experience is pleasant. The design of the interactive buttons is subdued and minimalist.

The colour tone of the labels, instructions etc. that appear are not loud and brash.

The whole graphics is tuned to make the Map window important (which is the main function of the software)



## 9. Help Users Recognize, Diagnose, and Recover from Errors

If a user enters an address that is not in the maps database, Google maps will suggest alternatives to help the user figure out the correct address they are looking for.

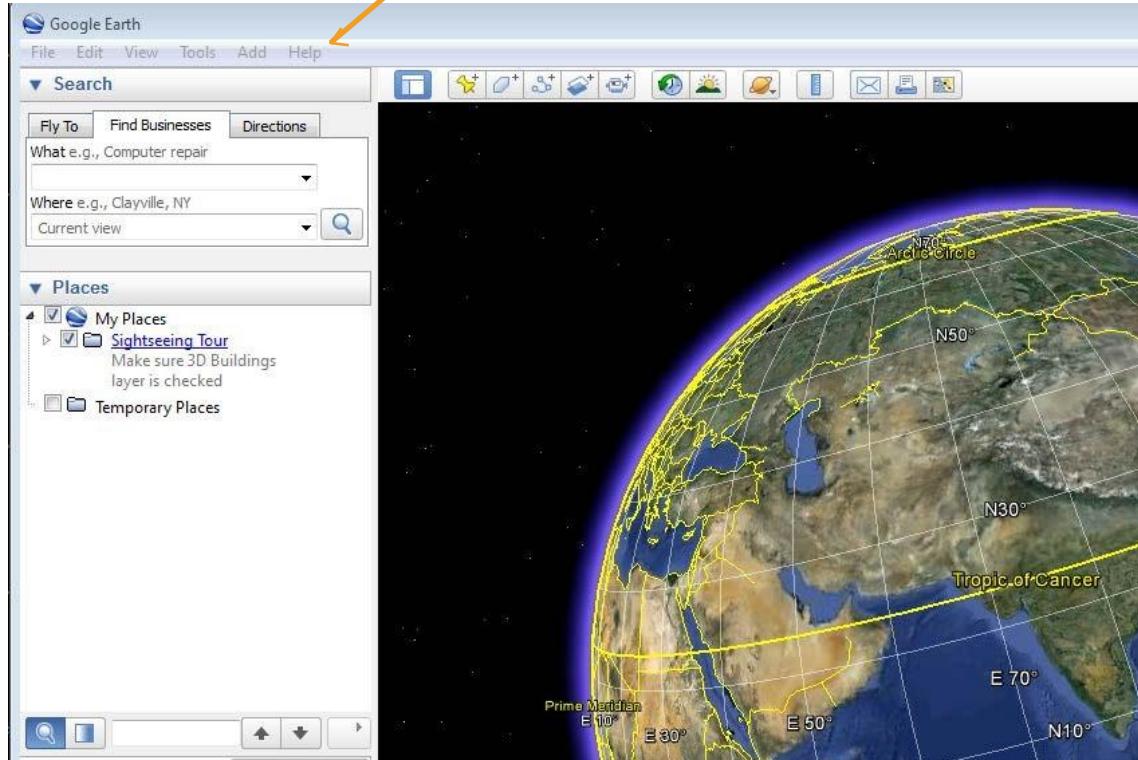
The software is user friendly.  
It does not penalise the user for not knowing how to use the software.

Even if an error happens, the software recovers quickly without the user having to do much more than going back to a previous screen by pressing “Back” button.



## 10. Help and Documentation

In the Documentation under 'Help' it is easy to use and figure out where to find the information you're looking for.



# Conclusion

Here the Heuristic evaluation is finally compiled into a consolidated report by including results of other evaluators.

Intensity of the problem may also be indicated in terms of severity.

High Severity: It is a HCI problem.

Medium: Here the Problem needs the attention as it is partially resolved.

Low: Improvement can still be done to the existing state.

Heuristics	Evaluator 1	Evaluatr 2	Evaluatr 3	Evaluatr 4
<b>1. Visibility of System Status</b>	System status if the Network connection is lost or it is absent	Severity: Medium		
<b>2.Match between System and the Real World</b>	Good. No intervention required	NA		
<b>3. User Control and the Freedom</b>	For a novice user disappearing zoom slider bar can be confusing!	Severity: Low		
<b>4. Consistency and the Standards</b>	Good. No intervention required			
<b>5.Error Prevention</b>	Good. No intervention required			
<b>6. Recognition Rather than Recall</b>	Navigating by panning and zooming often leads to being lost. Direction of movement is required	Severity: HIGH		

# Homework:

For the same Google Earth application conduct a Heuristic evaluation for all Ten Nielsen's Heuristics and fill up the space under Evaluator 2 in the Table. What new aspects did you (expert) identify that the first evaluator did not?

Heuristics	Evaluator 1	Evaluator 2	Evaluator 3	Evaluator 4
1. Visibility of System Status	System status if the Network connection is lost is absent	Severity: Medium		
2.				
3.				
.				
.				
.				
10				

## References:

- Molich, R., and Nielsen, J. (1990). Improving a human-computer dialogue, *Communications of the ACM* 33, 3 (March), 338-348.
- Nielsen, J., and Molich, R. (1990). Heuristic evaluation of user interfaces, *Proc. ACM CHI'90 Conf.* (Seattle, WA, 1-5 April), 249- 256.
- Nielsen, J. (1994a). Enhancing the explanatory power of usability heuristics. *Proc. ACM CHI'94 Conf.* (Boston, MA, April 24-28), 152-158.
- Nielsen, J. (1994b). Heuristic evaluation. In Nielsen, J., and Mack, R.L. (Eds.), *Usability Inspection Methods*, John Wiley & Sons, New York, NY.
- The Ten Heuristics and description have been adapted from the authors publication. Copyright 2005 by Jakob Nielsen.

# **HCI Guidelines: Contextual Inquiry and Cognitive Walk Through**

Professor Ram Mohana Reddy Guddeti  
Information Technology Department  
NITK Surathkal, Mangalore, India

# **HCI Guidelines:**

## **Contextual Inquiry**

## Introduction

Contextual Inquiry is a field-based data collection technique employed to capture the detailed information about how users interact with the product in their work environment or in other words - interact with the product in its **context of use**.



Contextual Inquiry is a prototyping and user testing dominated method.

In Human Centered Designing (HCD) methodology, understanding the users, their needs, the context in which these needs raise and the context in which the user attempts to fulfill needs - is the first step.

Specific techniques have been developed to identify and specify the context of use. **This process is called “Contextual Inquiry”.**

Contextual Inquiry is a scientific way of understanding the users needs, their intentions and their practices.

**Def:** Contextual inquiry is the systematic analysis based on observations of users performing tasks / activity in a context.

Hypothesis is made by linking the cause (effect) based on these observations of users. The hypothesis is tested in discussion with the users. As a result of this the context itself gets understood in all the dimensions.

By 'Context' is meant the anchoring environment / situation / reference / work activity - with respect to which a designing process (solving a problem or conceptualizing a new product) is underway.



**CONTEXTUAL:**  
YOU INTERACT WITH  
THE PARTICIPANT IN  
HER/HIS NORMAL  
ENVIRONMENT, CONTEXT.



**INQUIRY:** YOU ASK QUESTIONS  
→ TO CLARIFY WHAT YOU'VE  
OBSERVED  
→ TO BETTER UNDERSTAND  
THE TASKS



**GOAL:** IN-DEPTH UNDERSTANDING OF THE  
 ↳ CONTEXT  
 ↳ ACTIVITIES, TASKS  
 ↳ PAIN POINTS, ISSUES, FRUSTRATIONS  
 ↳ BEHAVIOR  
 ↳ PROCESS, ROLES, INPUTS & OUTPUTS

WHILE IT IS IMPORTANT  
TO LEARN ABOUT  
PAIN POINTS,  
YOU SHOULD

GO FOR A BALANCED SESSION:  
YOU ARE THERE TO UNDERSTAND  
HOW TASKS ARE ACCOMPLISHED  
(AND WHAT ISSUES OCCUR DURING  
COMPLETION)



### PREPARATION

- > SCHEDULING; PLANNING TRAVEL (TIME & COST)
- > BASIC UNDERSTANDING OF THE DOMAIN  
(E.G. FROM STAKEHOLDERS, DOMAIN EXPERTS)
- > YOUR FOCUS - IDEALLY, YOU'LL OBSERVE MANY UNEXPECTED THINGS, BUT IF YOU NEED TO OBSERVE TASKS THAT RARELY OCCUR: EXPLICITLY ASK PARTICIPANTS TO PERFORM THEM
- > LISTING OUT YOUR ASSUMPTIONS
- > CONSENT FORMS
- > INTERVIEW SCRIPTS (E.G. FOR MULTIPLE SHORT ONES)
- > TESTING / PREPARING RECORDING EQUIPMENT
- > EXPLAINING THE ESSENCE OF THE METHOD:  
PARTICIPANTS SHOULD KNOW WHAT TO EXPECT



AFTER THE CONTEXTUAL INQUIRY: ANALYZE DATA  
E.G. WITH AFFINITY DIAGRAMMING - LOOK FOR PATTERNS!

# CONTEXTUAL INQUIRY

UX Knowledge Base Sketch (Ref: UX Knowledge Base's Infographic)



THIS UNDERSTANDING  
WILL HELP TO MAKE  
INFORMED  
DESIGN DECISIONS



### CONDUCTING A CONTEXTUAL INQUIRY HELPS

- (IN)VALIDATING YOUR ASSUMPTIONS (E.G. IS THAT USER NEED EXIST?)
- FINDING CLUES, SIGNALS IN THE ENVIRONMENT TO POSSIBLE ISSUES (E.G. CHEAT SHEETS ON A STICKY NOTE) → POTENTIAL AREAS FOR IMPROVEMENT!
- DISCOVERING ARTIFACTS & ACTORS YOU'VE NOT THOUGHT ABOUT
- CREATING MORE REALISTIC PERSONAS, SCENARIOS, JOURNEY MAPS ETC.



### DURING THE CONTEXTUAL INQUIRY



BE A FLY ON THE WALL!  
OBSERVE, CAPTURE  
EVERYTHING - AT THIS POINT  
YOU DON'T KNOW WHAT  
THINGS ARE RELEVANT!



CONDUCT "TRADITIONAL"  
SHORT INTERVIEW(S)  
→ GET AN OVERVIEW OF  
THE ACTIVITY  
→ BUILDING RAPPORT  
WITH THE PARTICIPANT



### "MASTER - APPRENTICE" RELATIONSHIP

- ASK FOR DEMONSTRATION  
INSTEAD OF EXPLANATION
- AGREE ON THE "RULES"  
OF INTERRUPTION BEFOREHAND



DOCUMENTATION!  
 - NOTES & SKETCHES  
 - PHOTOS  
 - RECORDINGS  
 (IF ALLOWED)



SUM UP WHAT YOU'VE OBSERVED,  
ASK WHETHER YOUR UNDERSTANDING IS VALID OR NOT  
→ THE DIFFERENCE BETWEEN WHAT YOU SAW &  
HOW THE PARTICIPANT INTERPRETS THAT  
IS A VALUABLE INSIGHT!



A TOO BIG GROUP CAN DISTURB THE PARTICIPANT,  
HOWEVER, A TEAMMATE CAN HELP YOU CAPTURE MORE,  
AND YOU CAN DISCUSS YOUR IMPRESSIONS LATER.



MAIN ADVICE: GO WITH THE FLOW, YOU MIGHT GET  
UNEXPECTED INSIGHTS! E.G. NEW USER NEEDS



### SOME CHALLENGES :

- MOTIVATING THE PARTICIPANT  
TO SHOW EACH RELEVANT  
STEPS (VS. RUSHING THROUGH)
- PRIORITIZING WHAT YOU  
PLAN TO OBSERVE (SO YOU  
DON'T RUN OUT OF TIME)
- EXPLAINING WHAT THE  
METHOD IS ABOUT & WHAT  
YOU ASK THEM TO DO
- IT CAN BE HARD FOR THE  
PARTICIPANTS TO SWITCH  
FROM A MORE PASSIVE  
POSITION TAKEN DURING  
THE INITIAL INTERVIEW  
TO AN ACTIVE ROLE  
OF PERFORMING TASKS
- MINIMIZING THE NUMBER  
OF OBSERVERS



ALSO CALLED:  
SITE VISIT

YOU CAN GET INSPIRATION  
FROM ETHNOGRAPHIC RESEARCH



ADDITIONAL METHOD:  
SHADOWING

Contextual Inquiry is predominantly a qualitative method. In some cases it is a qualitative cum quantitative method of research. The techniques used in Contextual Inquiry are rooted in diversified areas: Ethnography, Psychology, Ergonomics and the Design.

The Results of Contextual Inquiry are used to formulate the Users' *conceptual model* based on the visualization of the users' *Mental Maps* of tasks, intentions, interpretation and action.

# **Advantages of the Contextual Inquiry method over other user data collection methods.**

Marketingbased data or information on the user as a ‘customer’ or ‘consumer’ is of limited use for a HCI designer as it does not give mental & psychological insights while the user is using the device.

This method being open ended makes it valuable deep-mining of tacit knowledge from the user. Tacit knowledge is that knowledge which normally the user is not consciously aware of themselves.

It helps to develop a shared understanding between the device interface creator and the user.

Even though both the qualitative as well as quantitative data is involved, this method of Contextual Inquiry is reliable and scientific.

# Disadvantages of the Method

Disadvantages are few. Since majority of information is qualitative it is not provable statistically significant.

The inquirer needs to be highly skilled in multiple disciplines such as Ethnography, Psychology, Culture, Design and HCI.

Some field based difficulties:



Gaining confidence of shy and suspicious users can pose a problem

Users may not want to be seen as stupid and hence may exhibit extra smartness (mislead). It is well known that when observed humans do things different from the way when alone.

# Methods

In short the method involves:

- Going to the user's environment
- Observe real work in natural conditions
- Seek clarifications and confirmations through questions.
- Conceive the field observed data into a model.

The user is treated as an expert.

Interviewer observes users in real time as they perform the tasks. Questions w.r.t. users' actions are asked so as to understand their motivations and approach to a given set of interactions with system.

Care is taken NOT to 'lead' the user by prompting while inquiring or assisting them in completing their answers.

Interviews / observations are conducted at the users actual work place / environment.



# Data Gathering Processes

Inquiry alternates between observing and discussing / clarifying from the user as to what the user did and why.

In this technique the researcher interprets and shares insights with the user during the interview / discussions.

Often the researcher's understanding stands corrected by the user.

Researcher needs to take care that the discussions do not move away from the focus of the contextual inquiry.



# Planning for a Contextual Inquiry

Define the issue / problem / context as well suppose well for which the Inquiry is being planned.

Plan for identifying users, their location, their numbers, and their willingness to cooperate.

Work on the briefing that will be given to the participating users. Prepare a list of possible questions to start the dialogue with the users.

Prepare documenting mediums such as cameras, voice recorders etc.

# Tools / Instruments Used in Contextual Inquiry

- Open Ended Questioning based on Observations
- Pre-prepared Questionnaire (User Survey)
- Ethnographic Observation Dairy with Notes  
(These Notes are Converted into Affinity Diagrams)
- Focus Group Interviews
- Structured Discussions
- Photo / Video Documentation
- **Hierarchy Diagrams (Affinity Diagrams)**
- Story Boards
- Mind Maps
- Affinity Diagrams: It is the simplest way to organize field data. It arranges the Notes from Interpretation Sessions into a hierarchy that reveals common issues & themes across all users.
  - The Field Data that has the affinity to each other based on are grouped together to form a category.
  - The Affinity Diagram contains one or more categories.
  - Cards with Labels / Words describing a characteristic are moved around on a board lead to formation of the “Groups”. Groups are given labels.

# Exit Interview Summary

## Checkout operators leaving job

### Inefficient checkout process

#### Unhappy customers

Customers became impatient

There were always long queues

It's a fuss just to sell a few things

#### Till difficult to use

The till could be easier to use

I kept breaking my nails

There's a lot of button-pressing

### Not happy in job

#### Jealous of others

I could have done Jim's job better

The manager just swanned around

Some other jobs looked better

#### Boring job

I got sleepy sometimes

By the end of the day I'd be scowling

I just don't care any more

### Seats not comfortable

I just couldn't keep still

It's a poky little hole to sit in

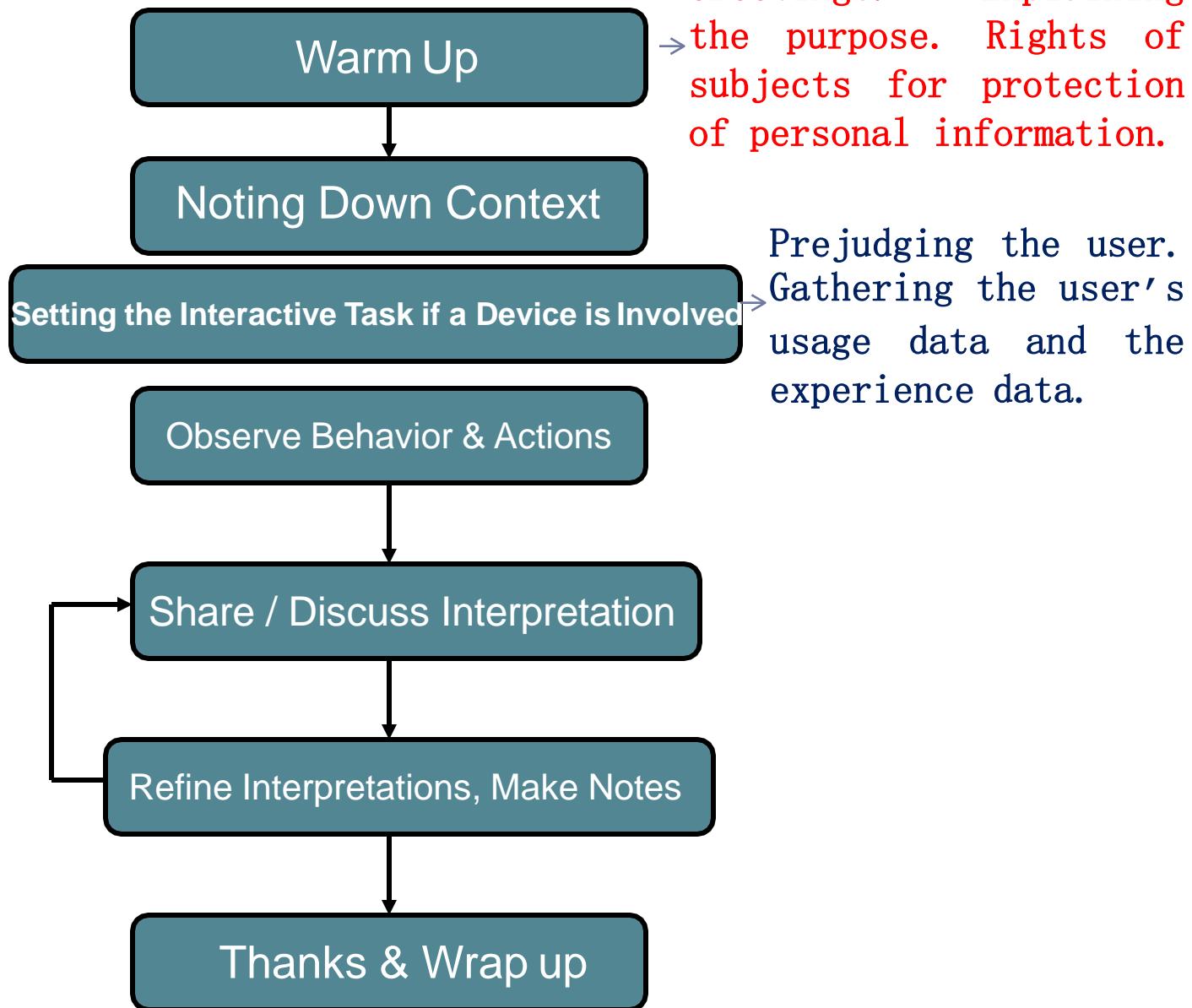
I kept getting stiff

I've always had problems with my back

# The Hierarchy Diagrams or Affinity Diagrams



# Stages of a Contextual Interview



# Analyzing the Data Collected in Contextual Inquiry

Data collected from contextual inquiry is analyzed, interpreted and finally visualized and represented by the researcher using one or all the following models which are part and parcel of the HCD process.

- Flow Model
- Sequence Model
- Cultural Model
- Artifact Model
- Physical Model

Descriptions of these models are given in the following slides.

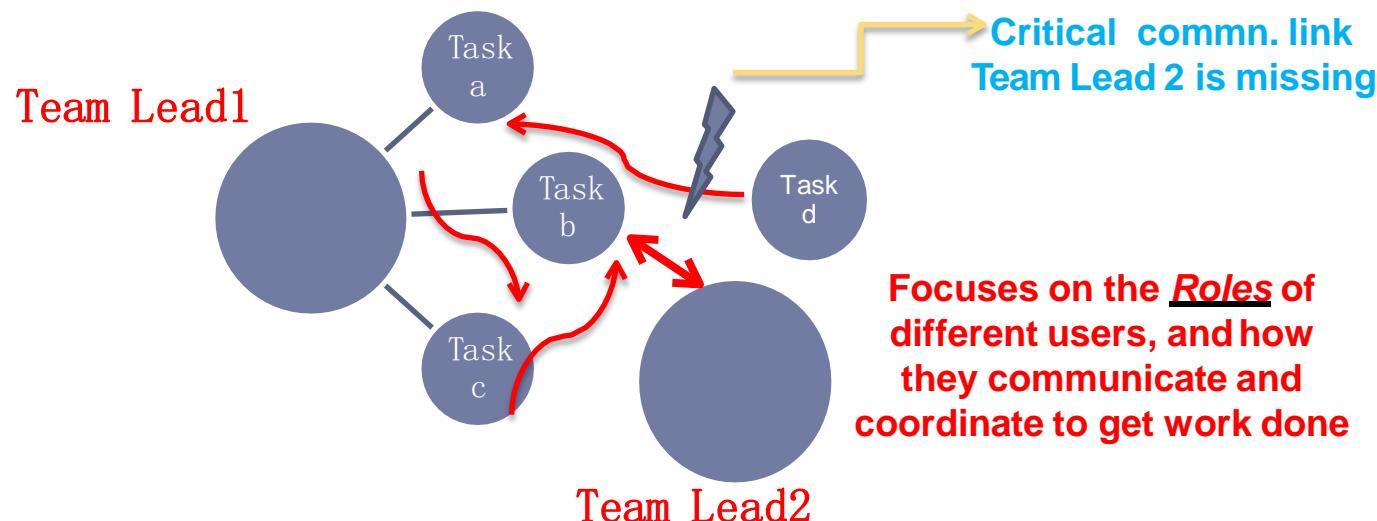
# Descriptions of Models

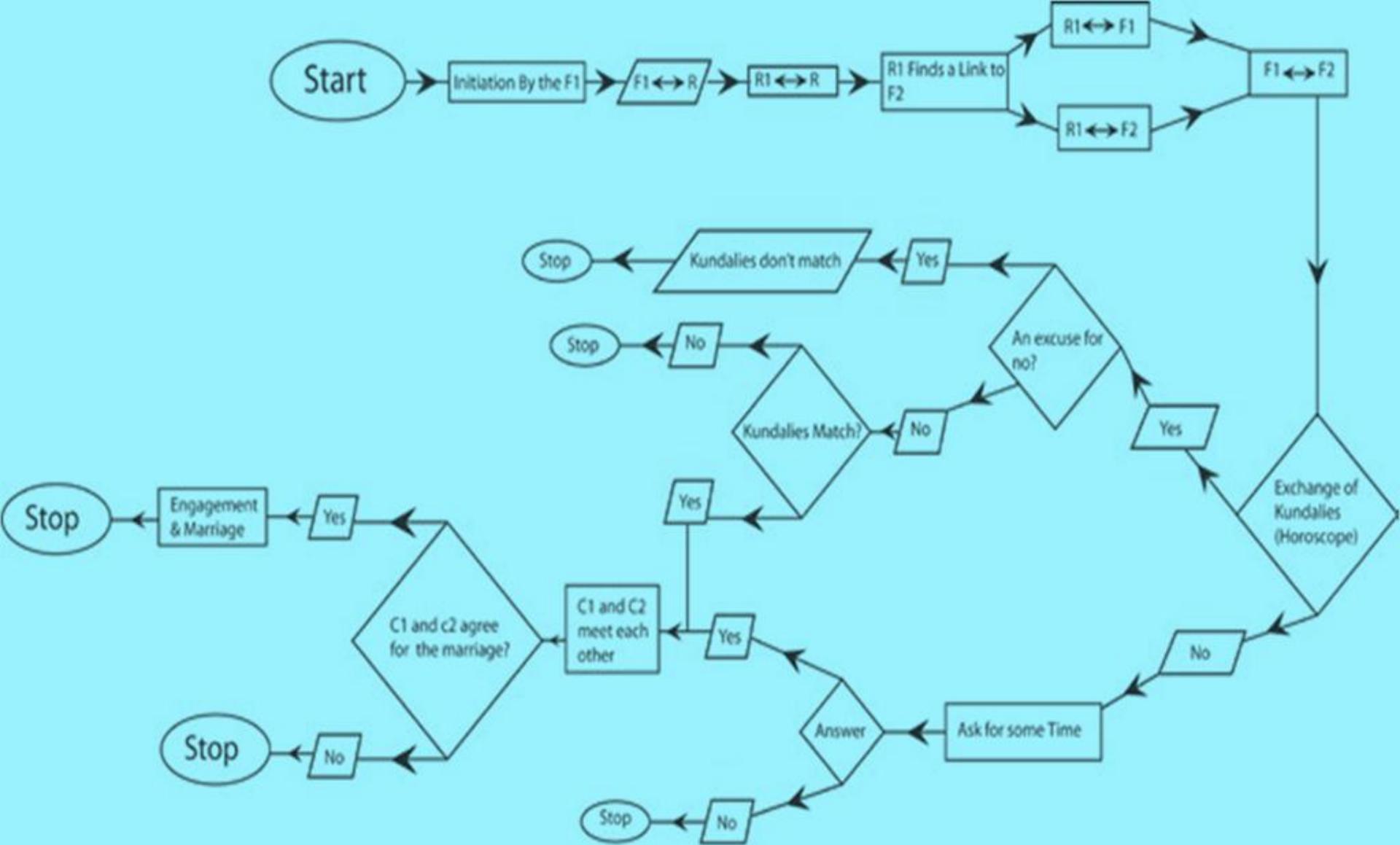
- **Flow Model**

Flow model represents the coordination, communication, interaction, roles, and the responsibilities of the people in a certain work practice.

It is based on the logic of flow of the information between different entities making up the system within the context.

This flow model is mainly used for depicting the logic behind the flow of information



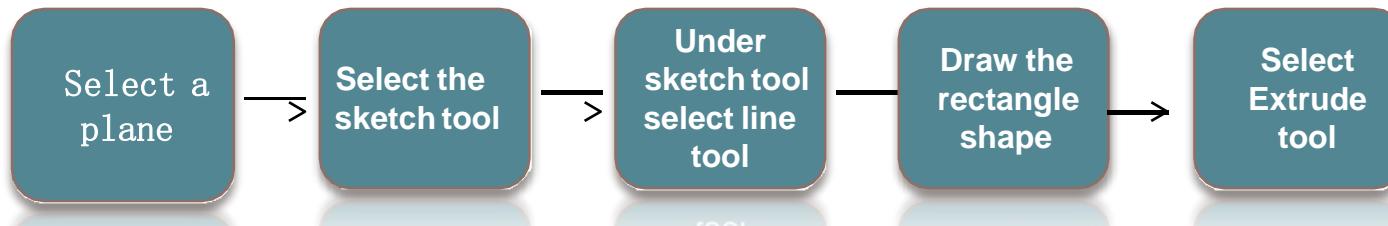


Flow chart for the Arranged Marriage Process in India

# Sequence Model – The Sequence

Model represents the steps users go through - to accomplish the specific task or an activity.

Sequence models are generally linear and sequential in nature. Sequence models of a number of smaller tasks when integrated represent the interconnected sequence within a larger system as shown in figure:



Enters the Code with the Specific Error

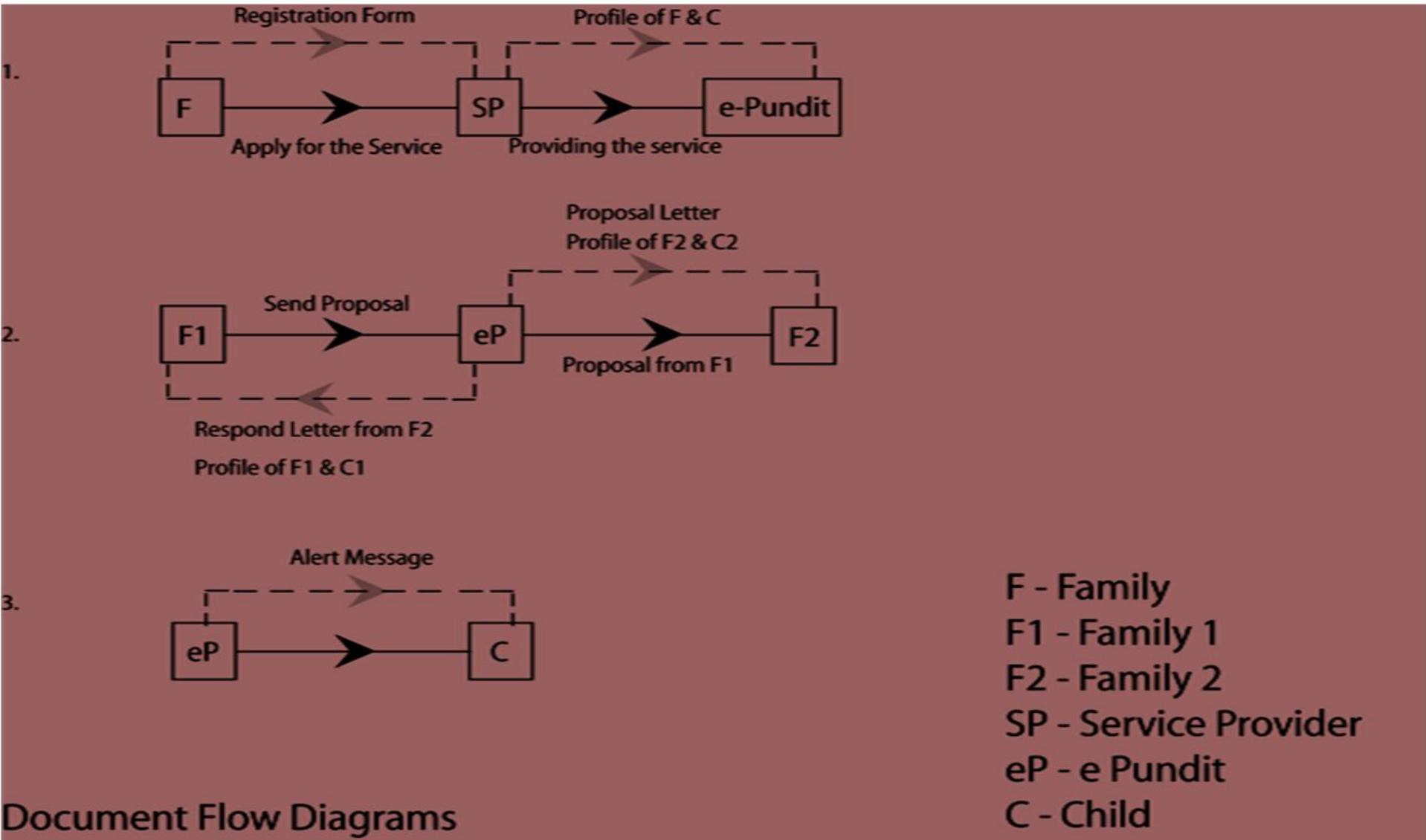
User Error

HELP

Low-level, step-by-step information on how work is actually done".

Includes the intent behind the action, the trigger that led the user to this action, and breakdowns that create problems.

# Document flow in an Indian Matrimonial Pre-Exchange



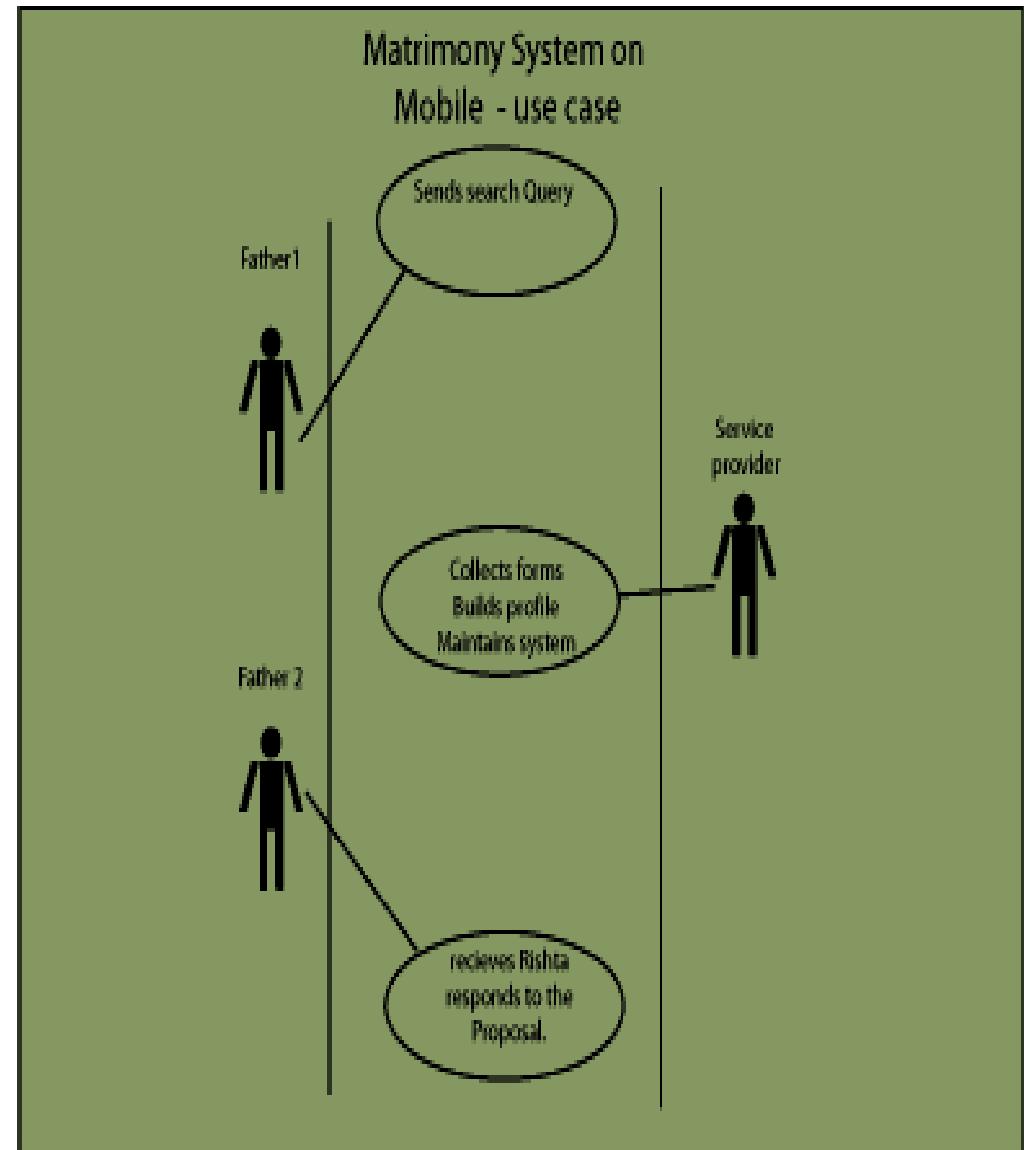
# Cultural Model -

Cultural Model represents the norms, influences, and the practices that are present in the work environment and which are specific to a particular region or traditionally followed as local norms.

Often culture specific comments or differences are mentioned using either flow diagrams or sequence diagrams or both.

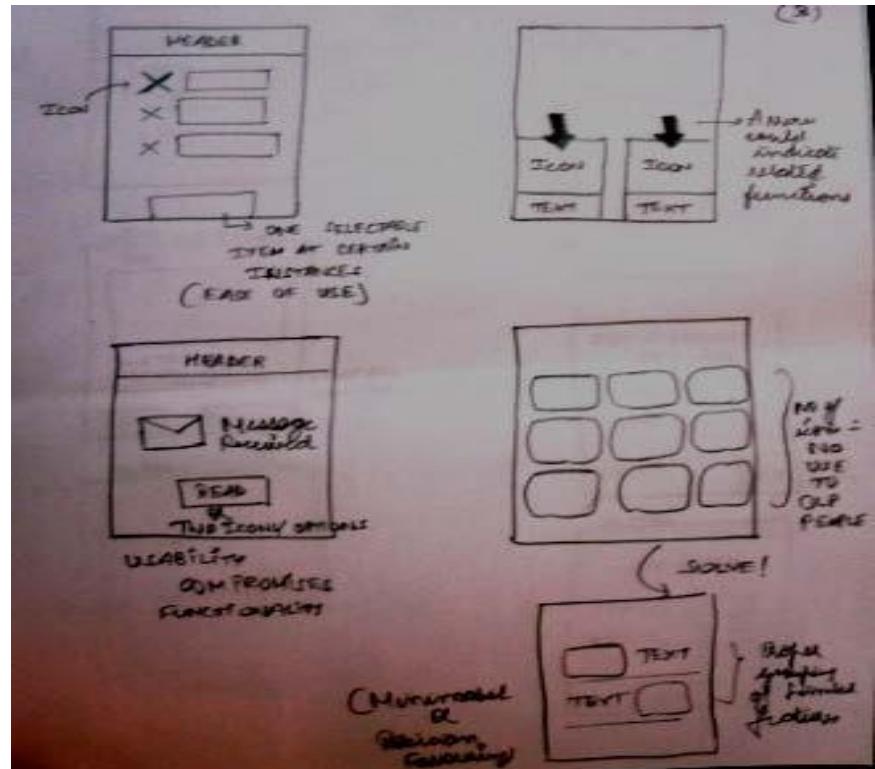
Language for example is a Cultural Model Variable.

In Indian culture, parents establish the first contact and then collects the information of a prospective son-in-law or daughter-in-law.



This flow model is not evident or observed for a person from another culture.

**Artifact Model** - It represents the documents or other physical things that are part of the work / task execution. These artifacts are aids to the tasks created while working or used to support the work. Example would be a Paper based voucher simultaneously filled in a particular step of a sequential task flow.



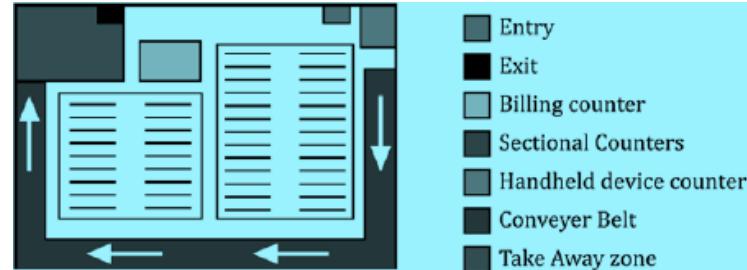
**Interviewers should inquire into the structure, content, presentation and the usage of the artifact.**

# Physical Model –

It represents the physical lay out of the environment where the work tasks are accomplished.

Simple examples would be an office layout, network topology, or the layout of icons on a computer display environment.

The flow of work as it moves in the physical environment can be represented as a map. Example would be of a retail shopping.



Selects,  
Pick

Hands over  
to Counter

Who weighs  
and enters  
item code

Packs  
/  
Seals

Sticks  
Label

Hands Back  
to Customer

Who puts  
it in his  
trolley



Includes the organization of space, the grouping of people, and their movement in the space

# Consolidating Work Models:

All flow models are taken for each user and their interconnectedness to form a whole system is attempted. The groups are formed based on roles played by individual users. The Extracts from the flow models represent the abstracts of communications, responsibilities and the constraints. The same thing is done with other models.

A report on the contextual inquiry is generated for designing team.

Model	Group	Abstract
Flow	Roles	Responsibilities, Communications, Constraints.
Sequence	Tasks	Triggers, Activities, Intents
Artifact	Role	Parts, Structure, Intent, Usage
Physical	Work Spaces	Places, Structure, Movement Patterns
Cultural	Influencers	Influences

# Homework:

Form a Group (3-4 people).

Choose a Project for Contextual Inquiry.

(Example: Course Registration System at the Beginning of the Semester)

Identify Users' / Stakeholders' Categories.

Conduct a Contextual Inquiry and Draw the Flow, and Other Models.

Draw the Affinity Diagram

Generate Five Work Models

# **HCI Guidelines:**

## **Cognitive Walk Through**

# Introduction

Cognitive Walk Through (CWT) is a usability method that focuses on evaluating a design (existing or proposed) for ease of learning particularly through explorations by the User.

Cognitive Walk Throughs [CWTs] are:

- Usability Inspection Methods [UIM]
- Focus on Evaluating a Design's navigation.
- Basis is 'Ease of Learning' by self exploration by the user

# Background

Cognitive Walk-Through (CWT) has the same basic structure and rationale found in software walkthroughs (Yourdon 1989).

In a software **code** walk-through, the sequence represents a segment of the program code that is ‘walked through’ by reviewers so that they can check certain **characteristics** (for example, that coding style is strictly adhered to, conventions for spelling variables versus procedure calls, and to check that system-wide invariants are not violated).

- In cognitive walk through [CWT], the sequence of actions refers to the steps that a user will require to perform on the interface so as to accomplish a task.
- The evaluators then ‘walk through’ that action sequence to check it for potential usability problems.
- Focus of the cognitive walkthrough is to establish how easy a system is to learn by operating it. The main focus is on learning through exploration.

Ref: Yourdon E, "Structured Walk Throughs", (4<sup>th</sup> edition) Englewood cliffs, NJ Yourdon Press.

## CWT Questions: (Wharton et al 1994)

1. Will the user try to achieve the right effect?
2. Will the user notice that the correct action is available?
3. Will the user associate the correct action with the effect that the user is trying to achieve?
4. If the correct action is performed will the user see that progress is being made towards solution of the task?

# Significance

Walk Throughs help answer interface design questions like:

- How will the user approach a task?
- What is the correct action sequence for each task and how it needs to be described to the user.
- How to achieve the desired action sequence form the user with minimum human cost and maximum efficiency
- How quickly will the user learn and becomes comfortable with the interface?

# Example of where the CWT is useful

When ATMs were first introduced one of the questions on the design of operational sequence was –

Should balance in account be displayed simultaneously every time the user access the ATM? (Or) Is it better to display the balance after the transaction is over.

A walk-through reveled both the above assumptions are out of sequence as far as the user is concerned.

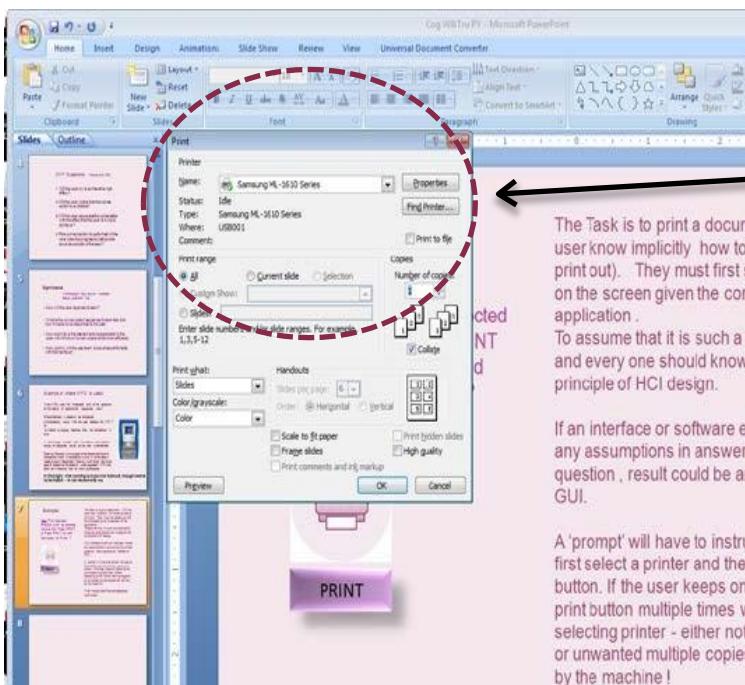
Seeking ‘Balance’ is a sub goal either before starting of a transaction or after a transaction is over. In either case it needs to be an independent Goal by it self rather than a sub goal of accessing the account. Users approach ATM more often for Withdrawal of money rather than for knowing Balance amount.

**In hindsight – this wanting to know the ‘balance amount’, though seems to be implicit - is not necessarily so.**



# Example 2

Task: Print the Document.  
Should a printer be selected first and then Press PRINT or Press PRINT first and then Select the Printer ?



The Task is to print a document. Will the user know implicitly how to do so (take a print out). They must first select a printer on the screen given the context of the application.

**To assume that it is such a simple action and - every user should know that – it is against the principle of HCI design.**

If an interface or software engineer makes any assumptions in answering this simple question, result could be an ineffective GUI.

A 'prompt' will have to instruct the user to first select a printer and then press print **button**. If the user keeps on pressing the print button multiple times without selecting the printer - either nothing happens or unwanted multiple copies are spit out by the machine!

**Cognitive Walk through will help to identify such gaps and prevent errors.**

# Cognitive Walk Through has Two Phases:

## Preparation Phase

i) Building A prototype {paper; mock up; screen based} with description. It need not be perfect or complete in all request.



ii) Making a list of selected tasks you want the user to 'walk through' the interface along with you. The task should have ready well defined sequences for Goals and sub goals with written actions used to complete each individual task.



iii) A clear understanding of the user, his / her background; level of expertise in the domain; prior experience of using similar software etc.

## Evaluation Phase

- i) Conducting the Walk through session
- ii) Recording the Observations
- iii) Analysis
- iv) Inferences
- v) Recommendation to Interface Team

## The evaluator should prepare to look for answers to Questions in Table:

**The purpose of conducting the walk through must be clear to the evaluator prior to starting the walk through.**

Question	
Can the users understand & reach the goal –the very purpose of the assigned task?	This will yield what the user is thinking once a task is assigned. Most of the time users do not think or act the way as the interface designer expects or wants them to
Will users be able to locate the buttons / GUI elements for the action they are supposed to perform given the task?	Often it is very difficult for the user to find the control/element to start . This is even more confusing to the user when there are several or multiple possibilities to start the sequence - on the GUI.
Does the interface provide understandable feed back at every action in the task sequence?	Often even if the users are able to locate the right GUI control / element can they tell with high degree of confidence that this is the right control for the action they want to perform and that they will indeed reach the goal. Intermittent feed back assures users that they are indeed proceeding in the right direction. Feed back can be in the form of sound or labels or motion or change in status.

# Overview of the actual Walk Through Processes

## Pre-preparation:

- 1. Define Users :** Who are the users. Identify them.  
(Categorise them as Novices, Intermittent & Experts)

## 2. Identify the tasks for the evaluation

Ex: Evaluation for “Checking out Balance on an ATM”

Prepare notes on what the user must know prior to performing the task and what the user should be learning while performing the task.

## 3. Prepare action sequences for completing the Tasks

Make a “AND THEN“ list of Goals & sub goal.

Ex: Overall Goal: Find out balance from the ATM

Subgoal1 : Activate ATM [Physical action Insert Card]

Subgoal2: Identify self [ Input pin code]

Sub goal 3 : Get balance [ press action button with label]

Sub goal 4: Get a print out [ if required]

Sub goal 5: Log out from ATM.

## 4. Conduct the Walk Through Session

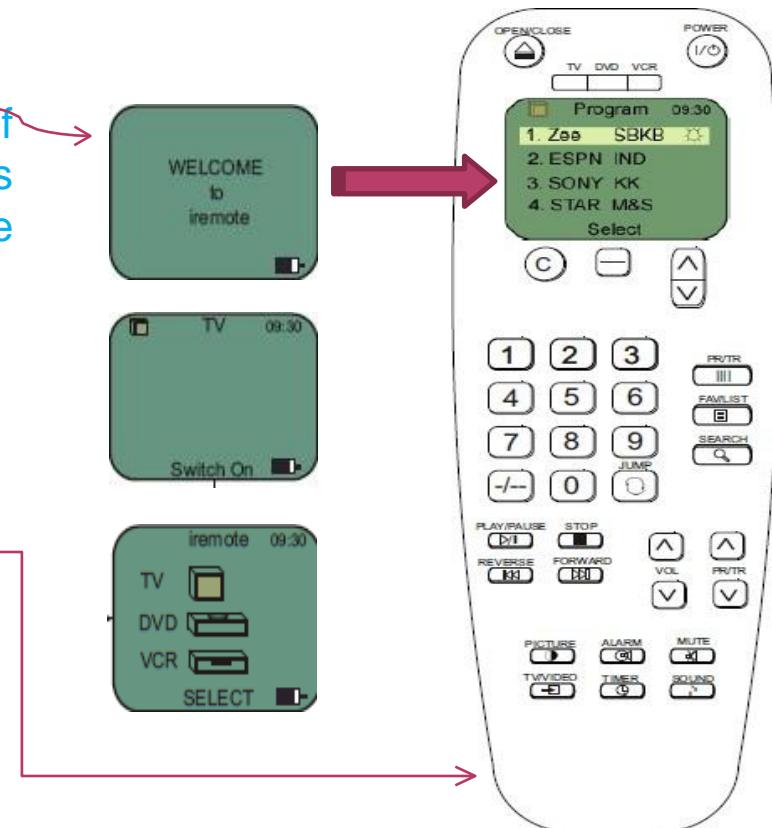
# Conducting the Walk Through Session

- Using the mock up prototype ask the user to perform a task. See Example of a mock up bellow
- Make the user walk through the action sequences for each task. See Example in next slide.
- Make a recording of observations in a Recording Sheet.



Paper mock up cut outs of screens are kept ready. User is asked to operate the remote control for a given task.

Depending on what the user presses, the corresponding paper display is slipped on the cardboard / mock up of the remote controller.



# Make the user walk through the action sequences for each task

Example of an Action sequence for forwarding Calls on a telephone.

Task: Forward phone calls to my office assistant / friends desk while I am out for a short period and reset it back to original state.

A1. Activate call interface.

R1. Sound feed back of activation done by tone 1

A2. Press #2 (Command to cancel call forwarding)

R2. Sound of registering press command by tone 1.

A3 . Listen to sound feed back confirming completion of action.

Time lapse Second Tone 2

Reverse cycle

A4. Activate call interface

R1. Sound feed back of activation done by tone 1

A6. Press \*2 (Command to cancel forwarding)

R2. Sound of registering press command by tone 1.

A7. Listen to sound feedback confirming completion of action.

Tone 2 End of sequence

End of Task.



# Observation during the Walk Trough

- The above task is assigned to a user. The user is asked to proceed executing the task on a mock up / paper prototype / wire frame prototype.
- The user is asked to achieve a goal (of forwarding a call in his absence and informed about the sequence of actions).
- The sequence of inputs as carried out by the user are observed.
- The errors committed (deviation from the expected sequence and corresponding action) are noted.
- The difficulties are mutually discussed with the user. Why a user acted in particular way and did not act in ways that was expected is explored.

# Make a recording of observations in a Recording Sheet.

Description of step.	Did the user try to achieve the end goal or did he give up At the start itself.	Did the user notice that the correct action choices are available.	Did the user confidently know that the choice being made by him/her is the right one?	Did the user understand the feedback after every action	Did the user Complete the Task With satisfaction	Comments / Alternative suggestions / solutions / discussion points.
	Yes – PARTLY - No	Yes – PARTLY - No	Yes No		Yes PARTLY No	
A1. Activate call interface.	YES	PARTLY	YES	YES	YES	
A2. Press #2 (Command to cancel call forwarding)	YES	YES	YES	NO	PARTLY	Was not paying attention to sound feedback as it is not expected.

# Analysis & Inference

## Evaluators Rating Sheet

Action in Sequence	System mismatch question	Potential Problem & Design solution	% Mismatch to ideal situation (qualitative estimation)
A1. Activate call interface.	Is it clear to the user that system has taken input	Low clarity of sound. Ambient Noise. Increase volume	30%
	Can the user resume control for the next action	YES	
	Are the systems response visible & interpretable	No	
	Is the end of the system action clear	YES	
A2. Press #2 (Command to cancel call forwarding)	Is it clear to the user that system has taken input	PARTLY	50%
	Can the user resume control for the next action	NO	
	Are the systems response visible & interpretable	PARTLY	
	Is the end of the system action clear	NO	

# Summarize the findings:



Percentage of mismatch 50%  
The Interface needs improvement.  
Sound Tone to be changed  
Sound Volume to be increased  
Additional Feed back to be incorporated in A2



End of Walk Through Testing Report

# Homework:

## What's the difference between a Heuristic Evaluation and a Cognitive Walk Through?

Conduct a Walkthrough for a new product being designed to train Computer (PC) servicing technicians.

**Users:** College dropouts (education upto Plus 2 + - 1)

**Context:** Undergoing training for routine computer maintenance

**Job:** Running Virus Scan Software (SW) in a Computer service centre.

**Level of expertise:** Novice. Users knowledge of computers includes starting a computer accessing files and folders, opening and closing files.

Task: Schedule a virus scan of System Files  
for a given time and date.

**List of Actions: As given below in sequence.**

1. Select target Scan from Virus scan SW files on PC
2. Select & Open MY Computer
3. Select Windows Folder
- 4. Select OK**
- 5. Select Schedule**
- 6. Select Enable**
7. Determine Time for Scan
8. Set Weekly as Schedule
- 9. Select Tuesday**
- 10. Select OK to complete task**
- 11. Check if Scan is Scheduled as per settings**

task walkthrough template					
1 <sup>st</sup> : break task down into steps					Task number: _____
Description of Step	Q1: Will users be trying to produce this effect?	Q2: Will the user notice the correct action is available?	Q3: Will the user know the correct action is the right one?	Q4: Will the user understand the feedback?	Comment / solution

# References:

C. Wharton, J Rieman, P. Polson & C Lewis; The Cognitive Walkthrough Method: A Practitioner's Guide. In J. Nielsen & R. L. Mack (Editors), Usability Inspection Methods, John Wiley & Sons, New York.

Yourdon E; Structured Walkthroughs (4<sup>th</sup> Edition)  
Englewood cliffs, NJ Yourdon Press.