Human Computer Interaction - An Overview

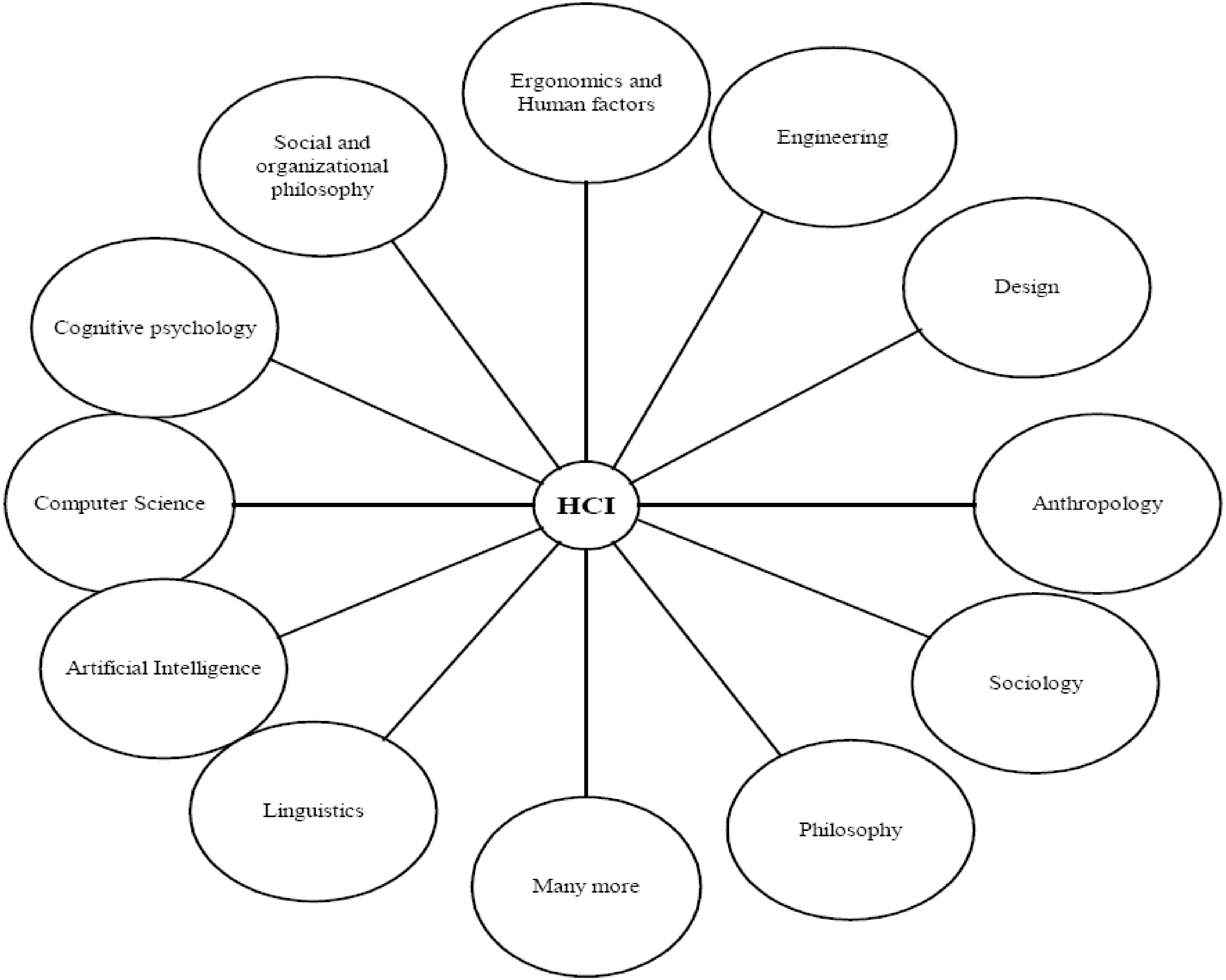
Professor Ram Mohana Reddy Guddeti
Information Technology Department
NITK Surathkal, Mangalore, India

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.)


Ergonomics and Human factors
Engineering
Social and organizational philosophy
Design
Cognitive psychology
Computer Science
HCI
Anthropology
Artificial Intelligence
Sociology
Linguistics
Philosophy
Many more

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 Guddeti
Information Technology Department
NITK Surathkal, Mangalore, India

Usability Engineering UE

<u>Overview of Usability Engineering (UE)</u>

- 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.
- Xristine 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

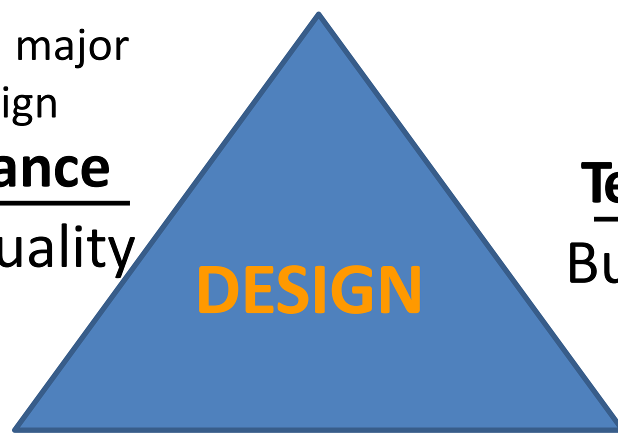
---

Usability is conceptualized into the product by DESIGN

Usability has three major components in Design

Appearance

Visual Quality



Technology

Build Quality

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

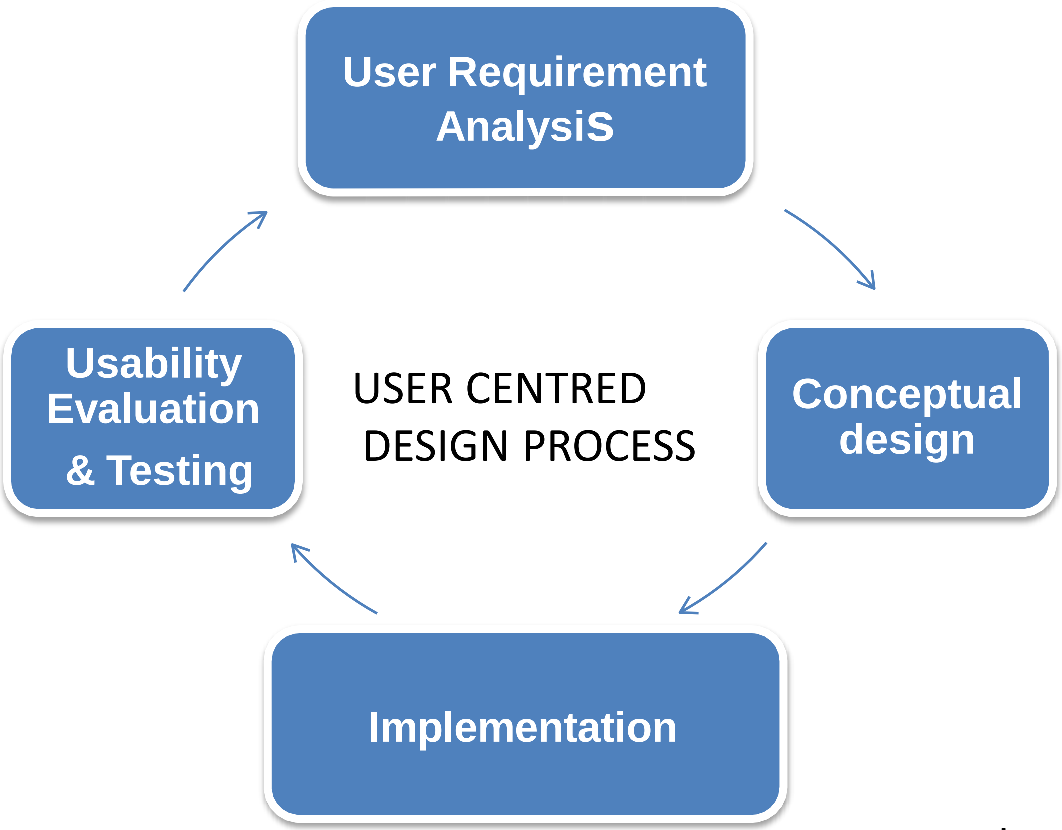
Achieves 5 times Enhancement in Engineering value.

Home Work

Usability Evaluation

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

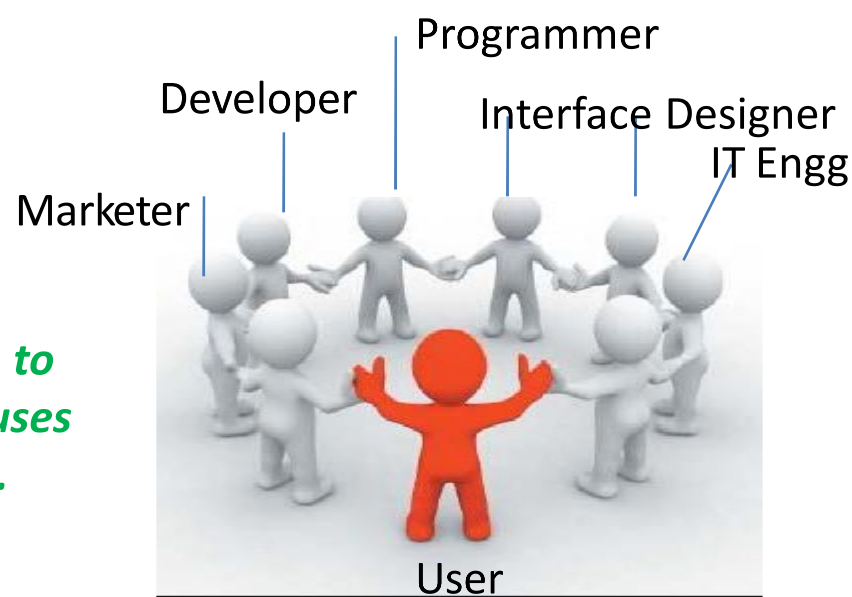
		Rating out of 10	
Effective to use	- Functional		
Efficient to use	- Efficient		
Error free in use	- **Safe**		
Easy to use	- Friendly		
Enjoyable in use	- Pleasurable		
	Total :		



•UE is based on a **User-Centered 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.

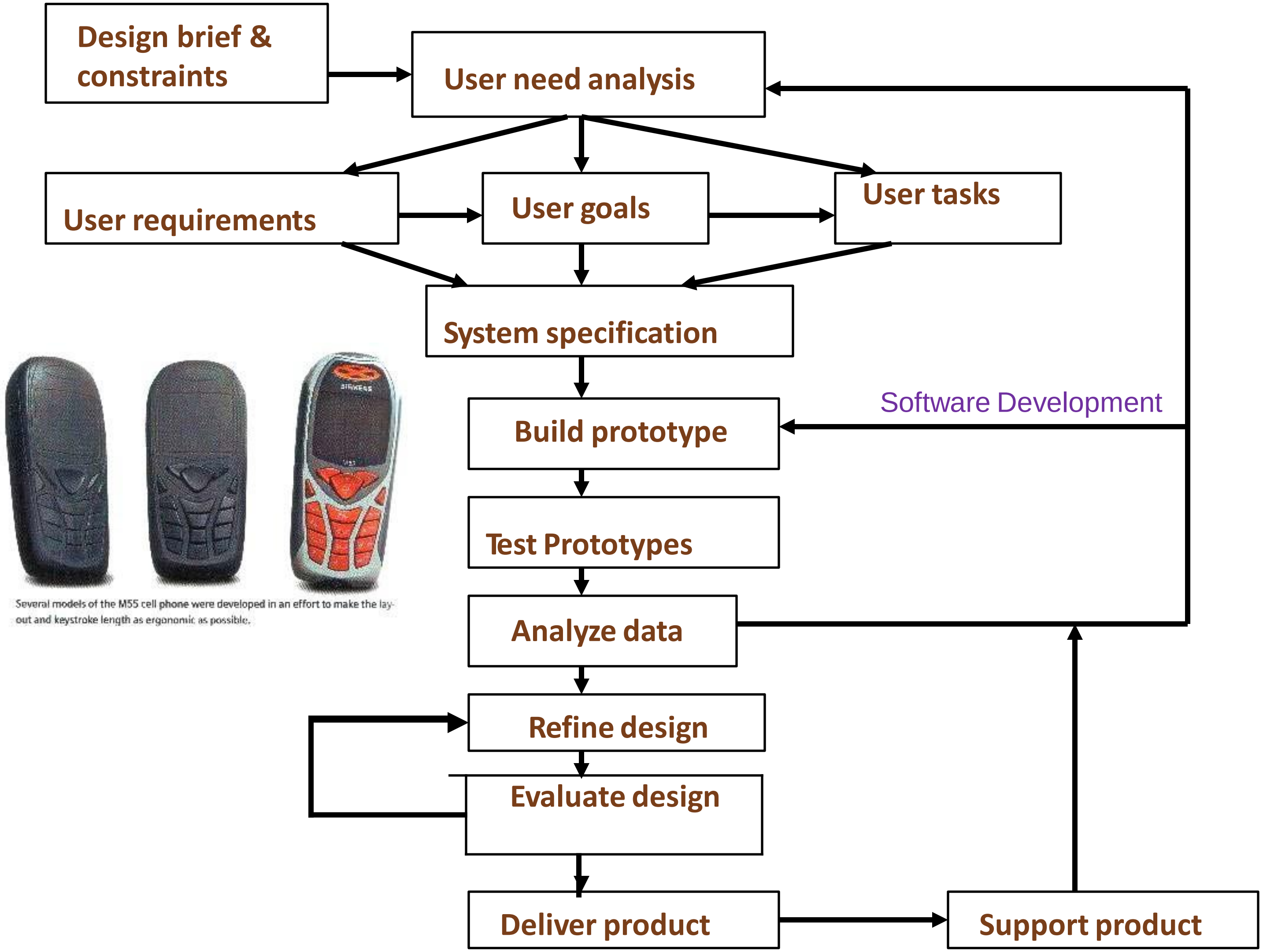
ISO 13407, 1999

"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



Several models of the M55 cell phone were developed in an effort to make the lay-out and keystroke length as ergonomic as possible.

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

Design

Engineering

$$UE = \{UX + UI + ID + HCI\}$$

Creative Design

Computer Science

Cognitive Science

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, <u>usability is not</u> 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

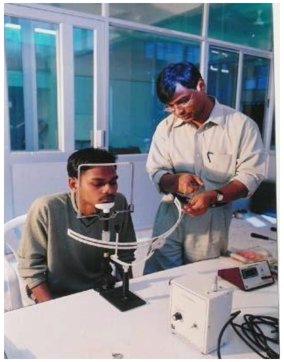
+ function + aesthetics

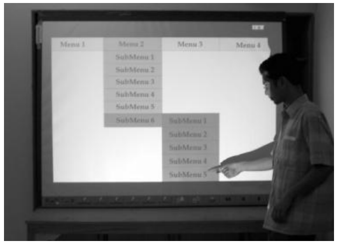
  - A way to deliver a product that works

- 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

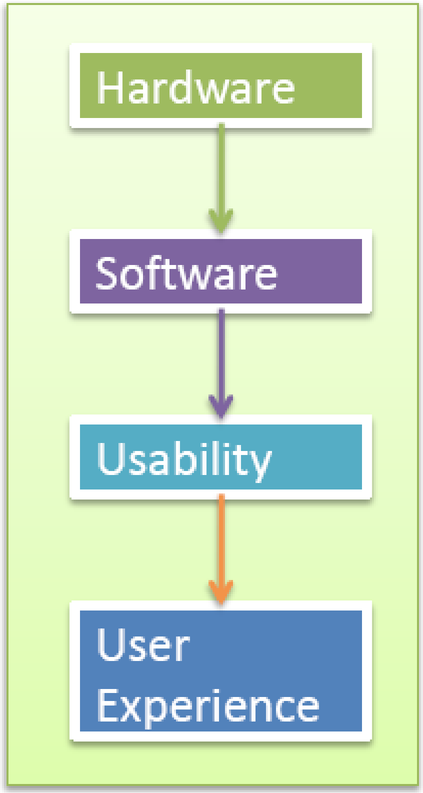
Loss of market share, good will

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



- User as 'cog' in the system – (1970's)
- User a source of error (80's),
- User a social actor (90's)
- User as a consumer now (2000's)

Previous approaches are insufficient

- When user is a consumer, his needs need to be understood in order for the product to stay in the market....

- User experience with interactive products start determining which ones will sell.

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.

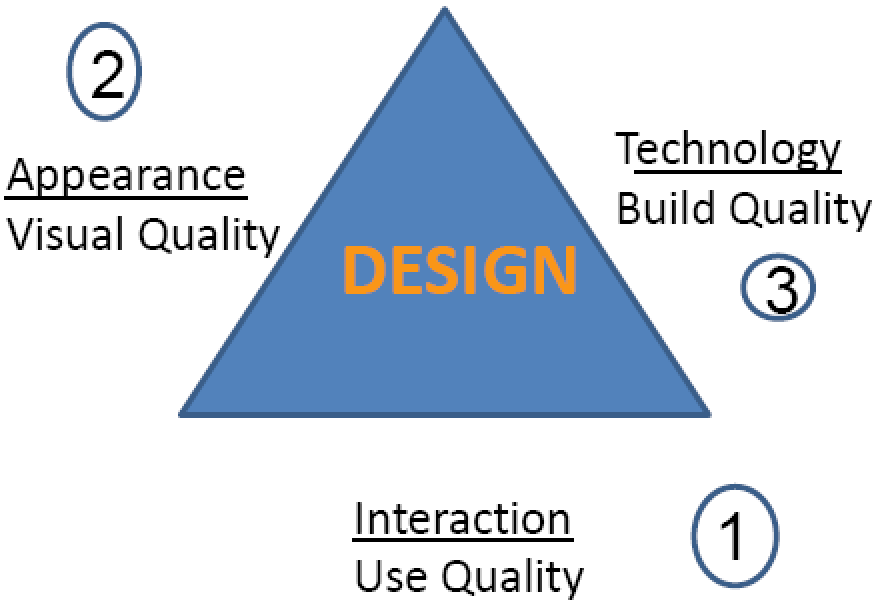
Learnability - Help the users to master the system
Consistency - Let the users have to learn only once
Familiarity - Build on users' prior knowledge
Standards - Respect established cultural and
- application specific conventions .
Self-descriptiveness - Make objects and controls intuitive
Help - Provide 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

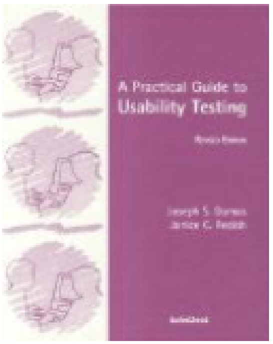
- Aesthetics

– User Experiences ....

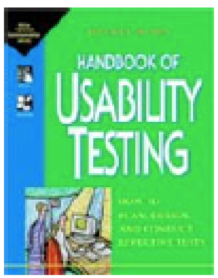
- Designing for contextual awareness

Culture , Behavior

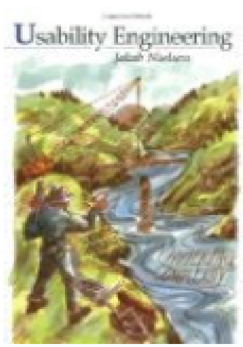
Some Usability Books



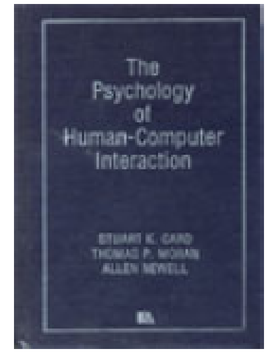
. A Practical Guide to Usability Testing by Joe Dumas & Ginny. Redish (1993)



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



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



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

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"

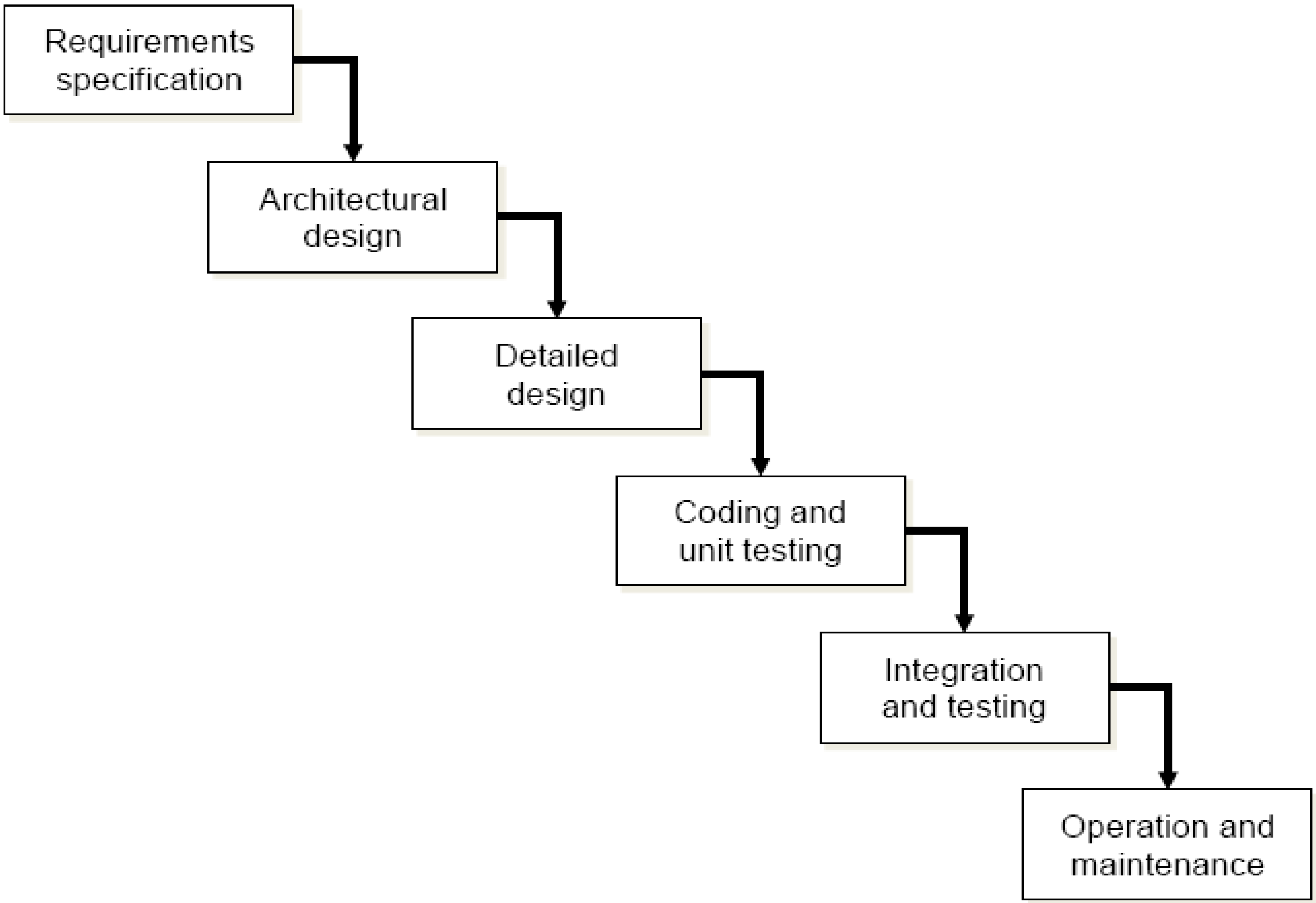
The Central Idea

- 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



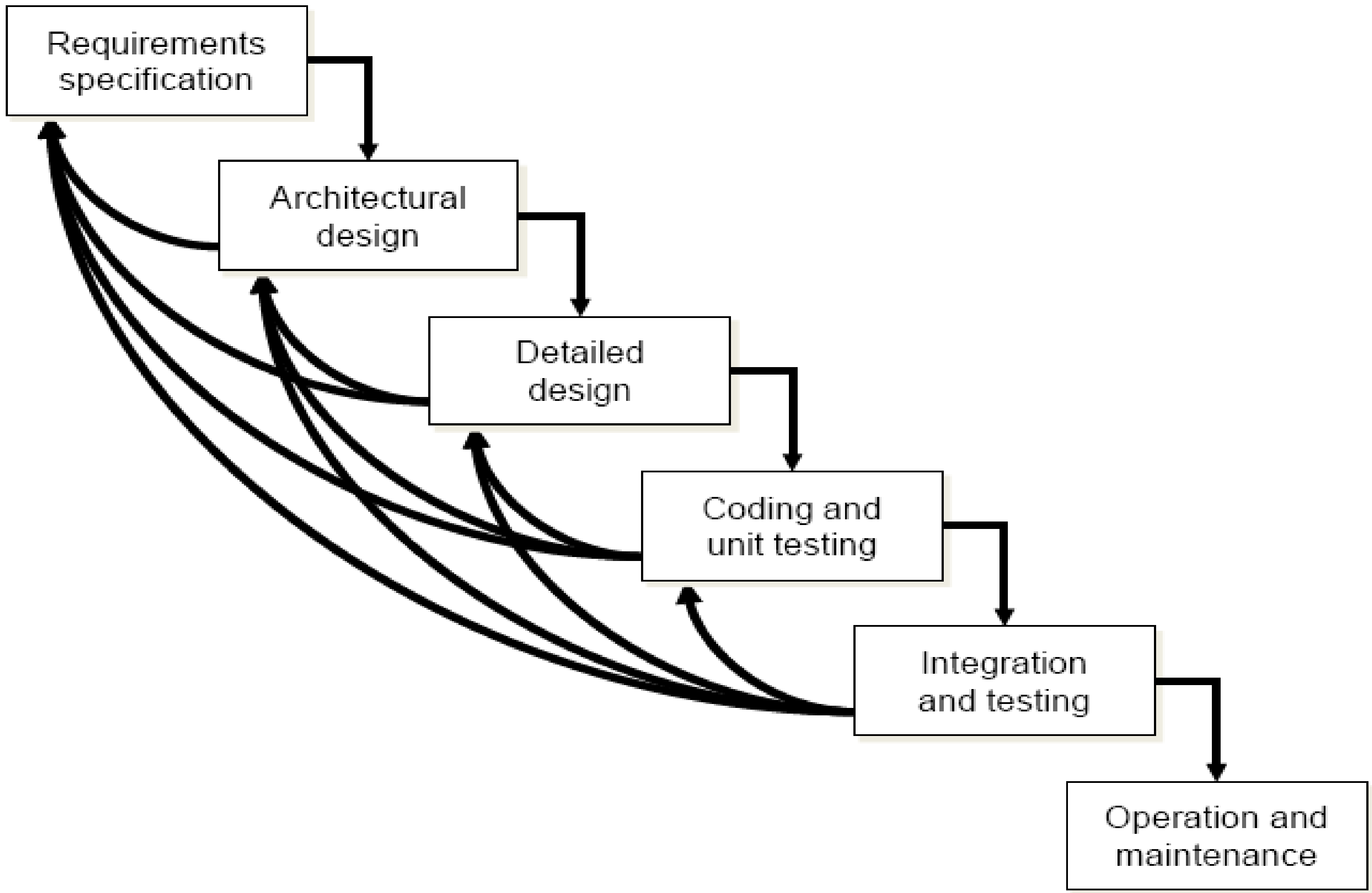
What Happens in Software Engineering

- 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”

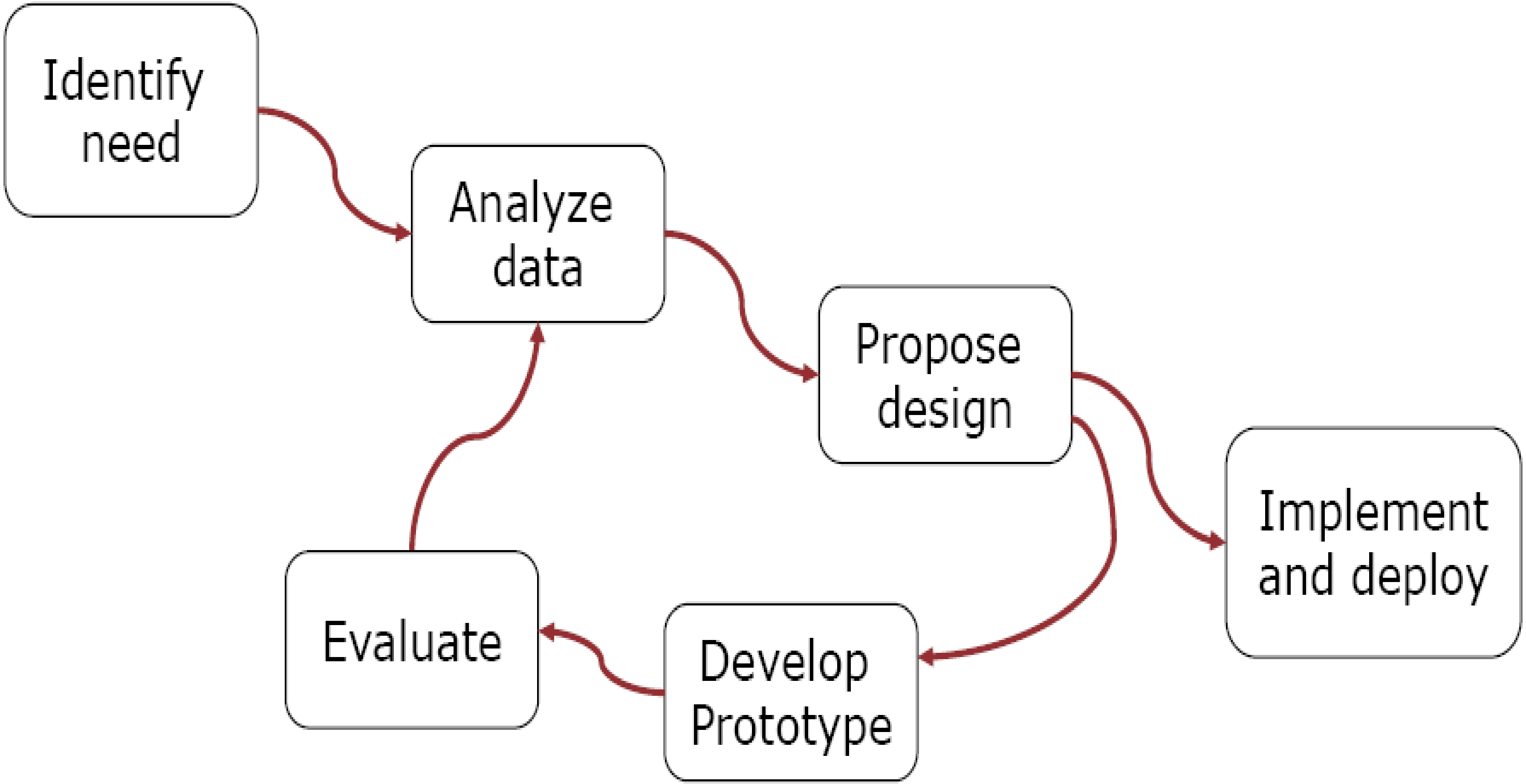
Participatory Design: Key Points

- 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

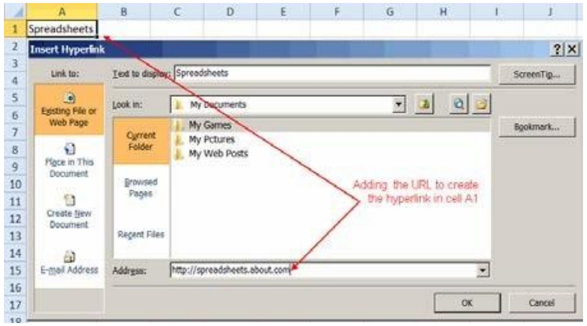
HCI: Interactive System Design (GUI Design and Aesthetics)

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 an 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

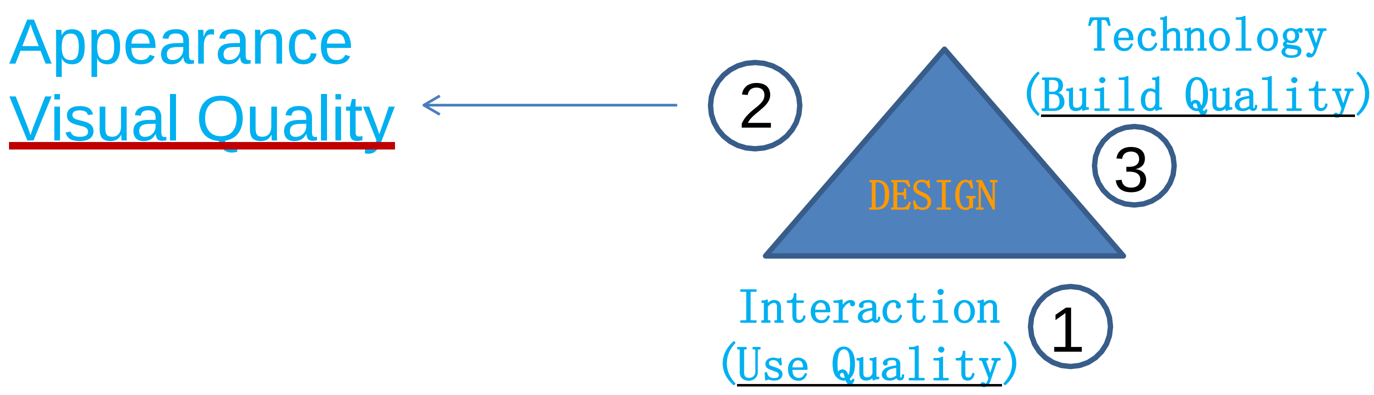
The Philosophical argument of aesthetics shown below is incorporated into Interfaces through Graphic designing

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

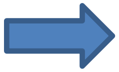
Principles of Design in Visuals

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.

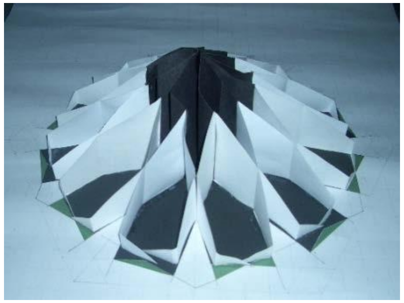
Form follows function is a design approach wherein the form (overall layout / composition/geometric shape) of a GUI is determined by what function it does.

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.

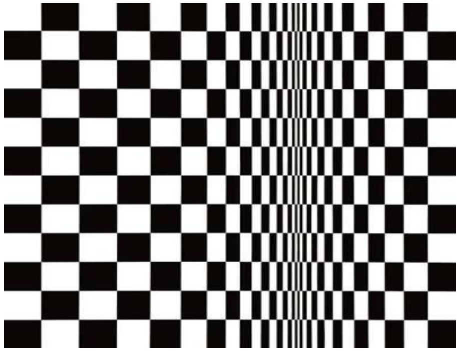
Notice that many of the terms on the right figure are also used in Mathematics.

Design, therefore has both, Aesthetics and Mathematics, underlying it.

- **Balance**
- **Unity**
- **Proportion**
- **Harmony**
- **Direction**
- **Rhythm**
- **Symmetry**
- **Pattern**
- **Emphasis**
- **Contrast**
- **Movement**

Description of some of the Principles

Rhythm & Movement

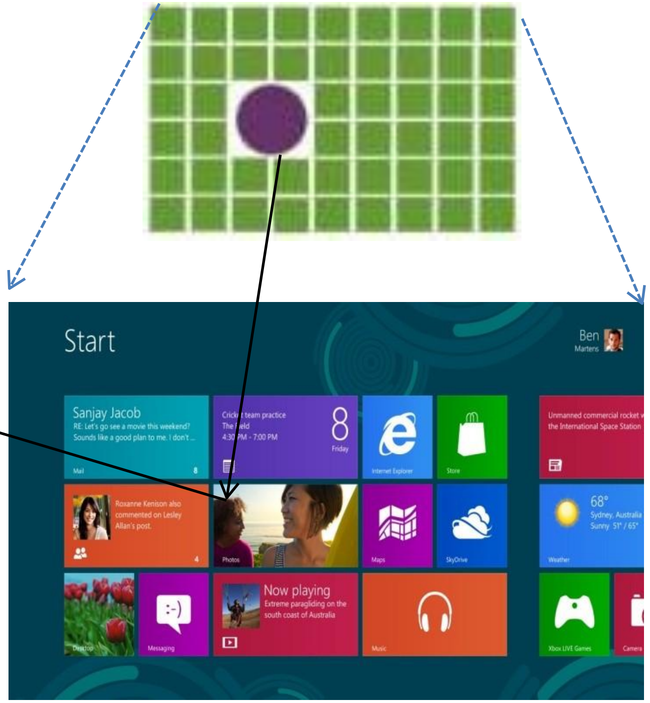


- **Movement** is the path the viewers eye takes through the artwork, often to focal areas. Animation is often used.

- **Rhythm** is created when one or more elements of design are used repeatedly to create a feeling of organized movement / direction.

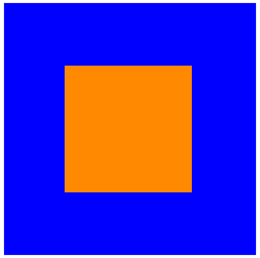
Emphasis

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.

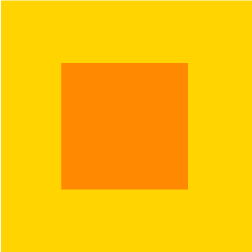


Contrast

- 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.

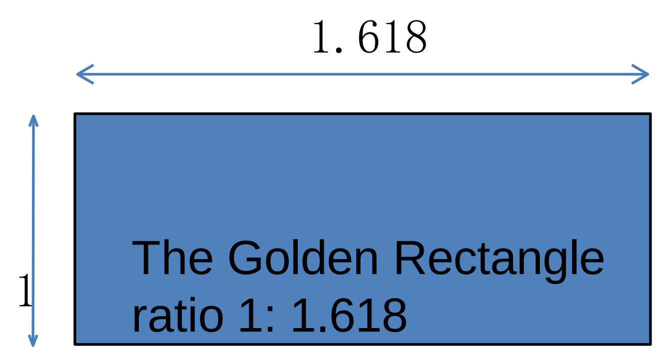
Ex: Windows 11 GUI **Ex: Windows 8 GUI**

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 ?

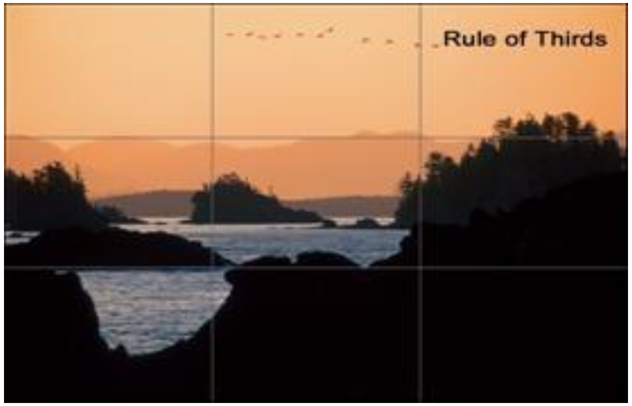
Proportion

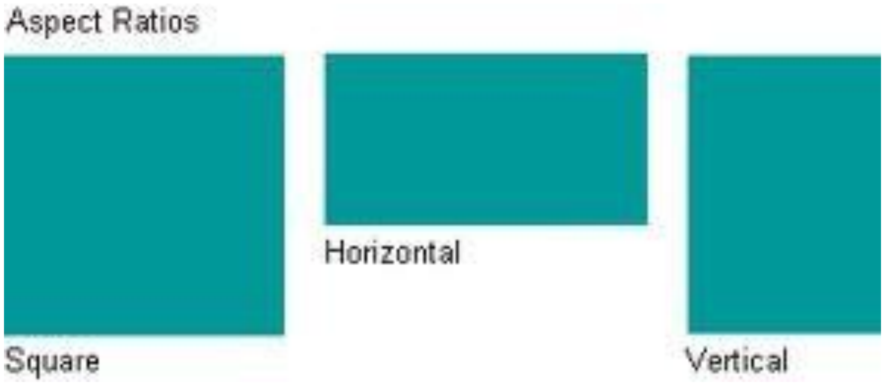
Size relationships found within an object or design. Also a comparison in terms of ratio of size, shape, etc with neighboring elements.
Example see proportions of various buttons within Windows 8 screen

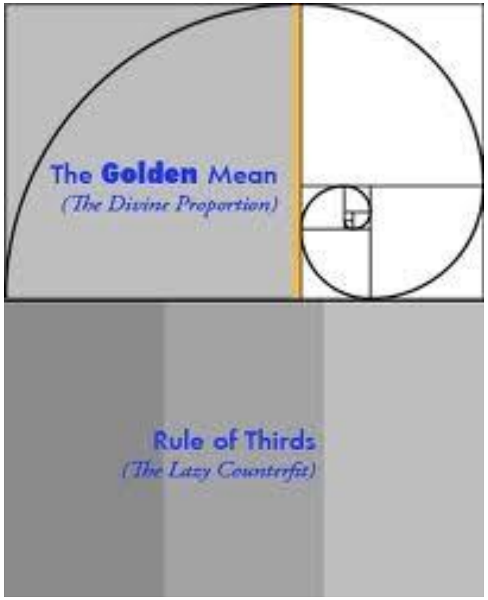


Proportion & Rule of Thirds Division of a Screen

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.







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).

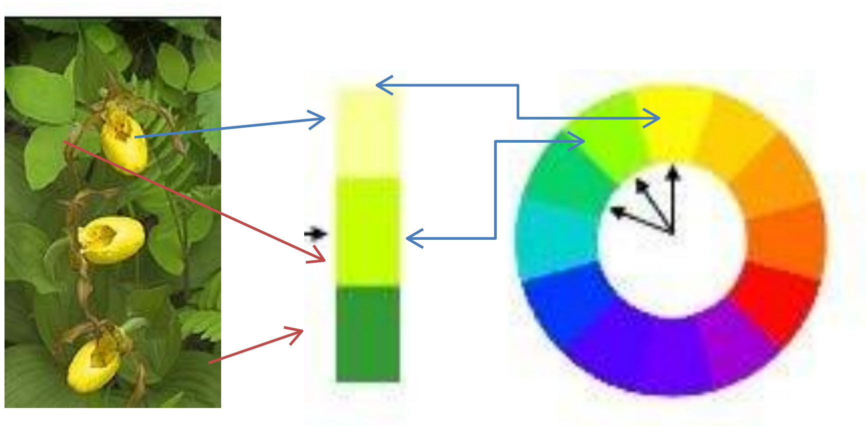
Value: lightness and darkness of the hue –

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.

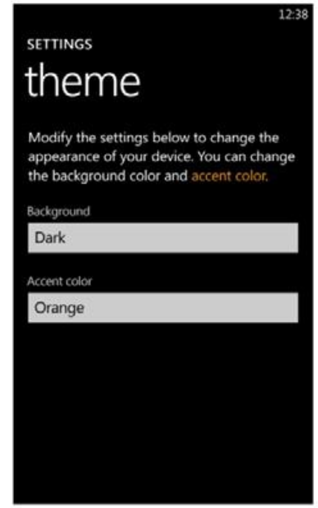
Colour Theme

Choices of colour given to the User. Simple Pick & Chose 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.

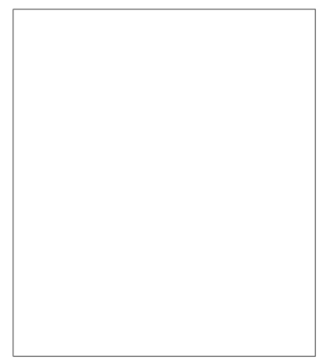
THEME SETINGS
Shown in light theme



THEME SETTINGS
Shown in dark theme

ACCENT COLOR PICKER
Shown in light theme

ACCENT COLOR PICKER
Shown in dark theme

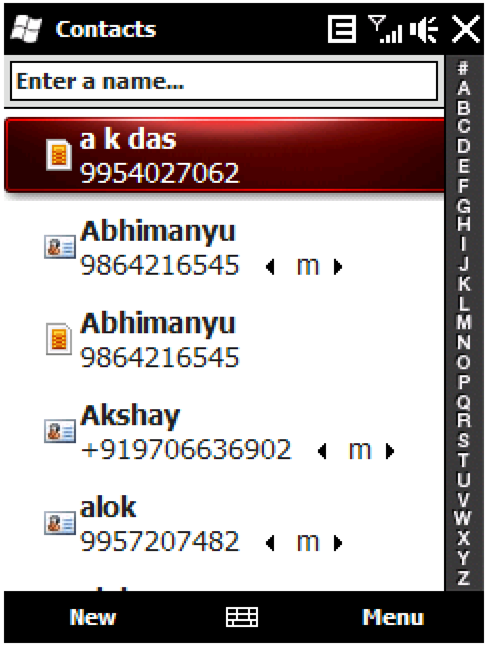


Graphic Design Principles: Example: Mobile Screen

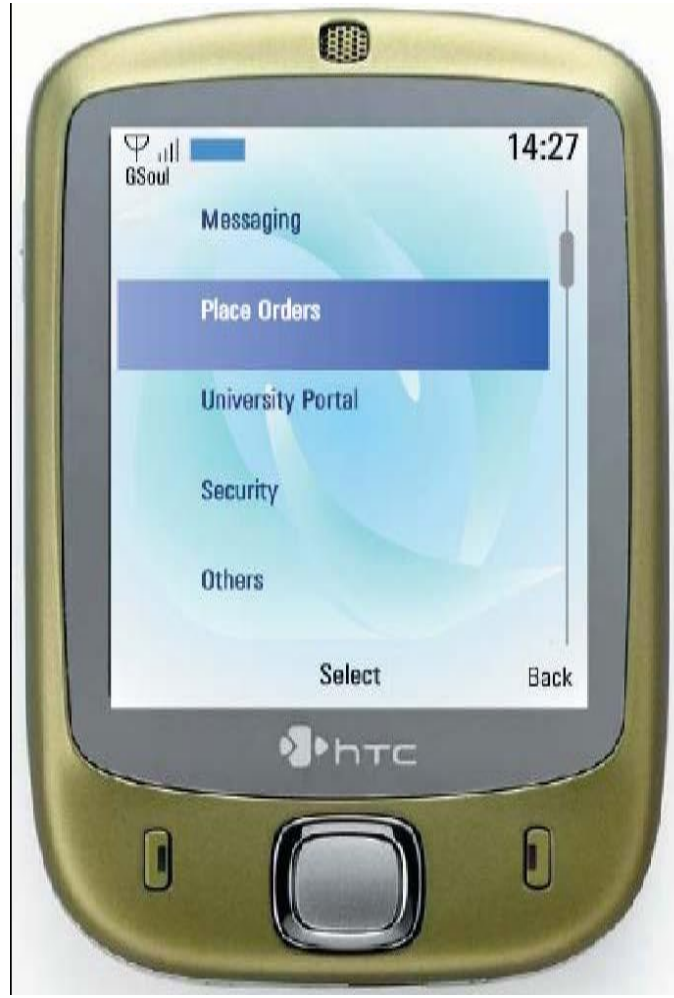
The Clustering Principle:

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 catagorised 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

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

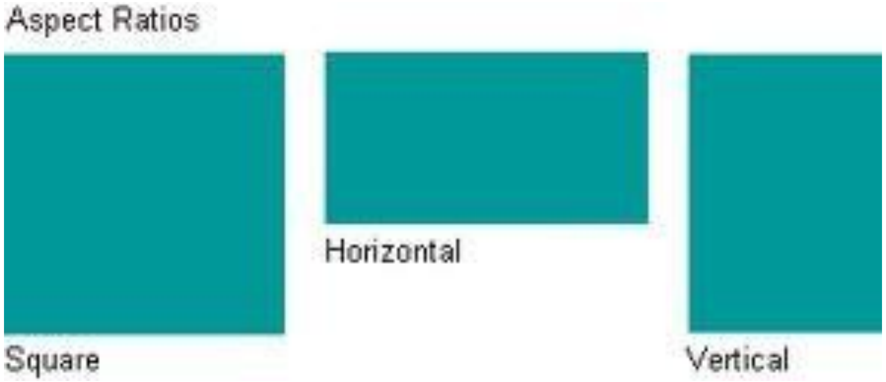
ಜನ◌ೀ ಜನ್ಮಭೂಮಿಶ ಸ◌ಾದ ಗರೀಯ जननी
जन्मभ◌ू मश्च स्वगार्द प गर यसी
ஜநந ஜந்ம மிஶ்ச
ஸ்வர்காதப க ய

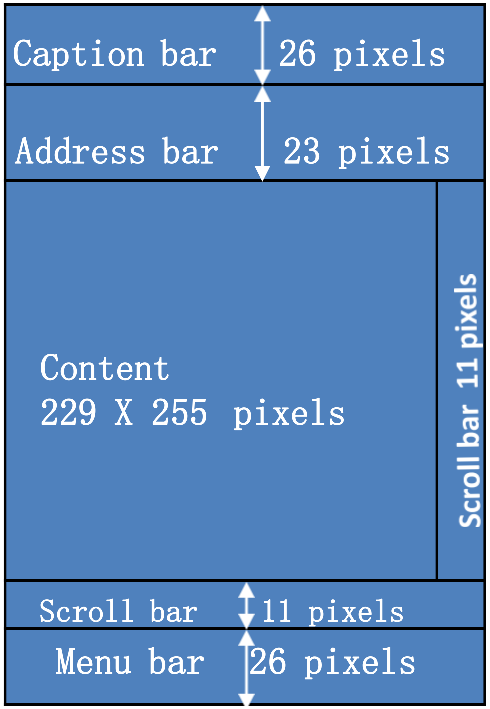
Weight of font matters

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

ಜನ ◌ೀ ಜನ ಭೂಮಿ ಶ ಸ ◌ಾದ ಗರೀಯ
जननी जन्मभ◌ू मश्च स्वगार्द प गरयसी
ஜநந ஜந்ம மிஶ்ச
ஸ்வர்காதப க ய
జ న జన్మ భ శ్చ స్వ గ్◌ాద ◌ి గరీయ
ജനനീ ജന്മഭൂമിശ്ച സ
ര്ഗാദപി ഗരീയസീ જનની
જન્મ ◌િ◌ૂમ સ્વગાર્દિપ ગરીયસી जननी
ਜਨਮਭੂ◌ਿਮਸ਼ ਸਵ◌ਾਰਦਿਪ ਗਰੀਯਸੀ

Screen Resolution & Aesthetics

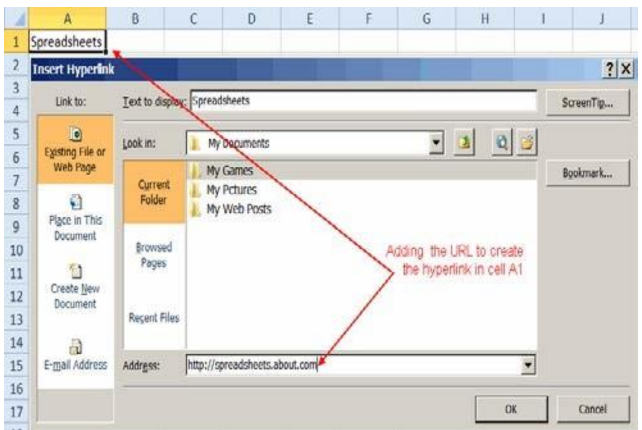




25~30% area is taken by buttons etc. Hence 229 X 255 is available from 320X240 display.

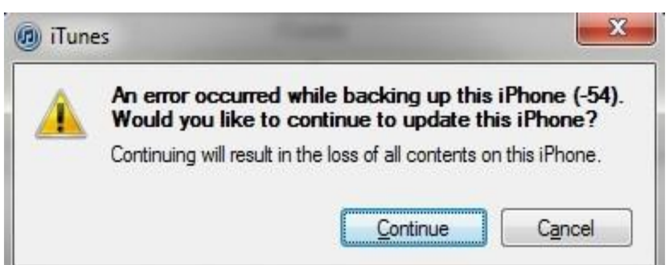
Some Unsatisfactory GUIs

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 & are 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.

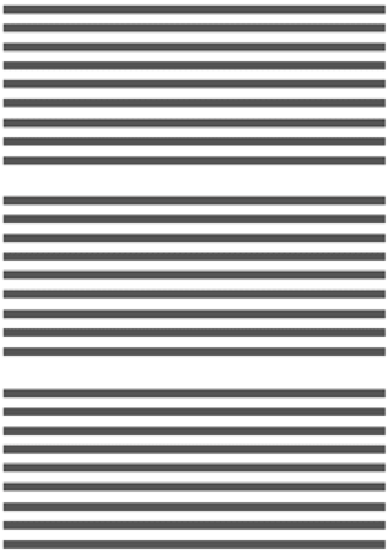
The state of 'please wait' is expressed in a pleasant peaceful unhurried manner.

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.

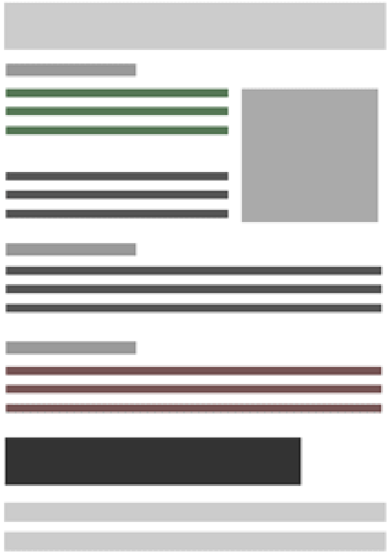
The icon has achieved this by employing aesthetic principles in their form, colour, shape, configuration, motion & composition – all of them put together holistically resulting in a simple 'design'.

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 & 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 & 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 slow
  - A video simulating the use of a system
  - A cardboard mock-up
  - A piece of software with limited functionality
  - Even a lump of wood

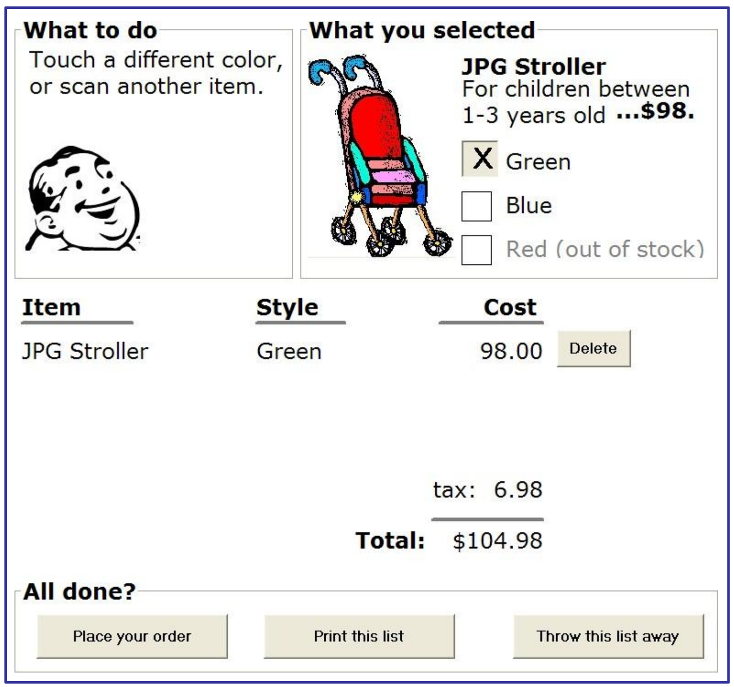
Prototypes in HCI

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

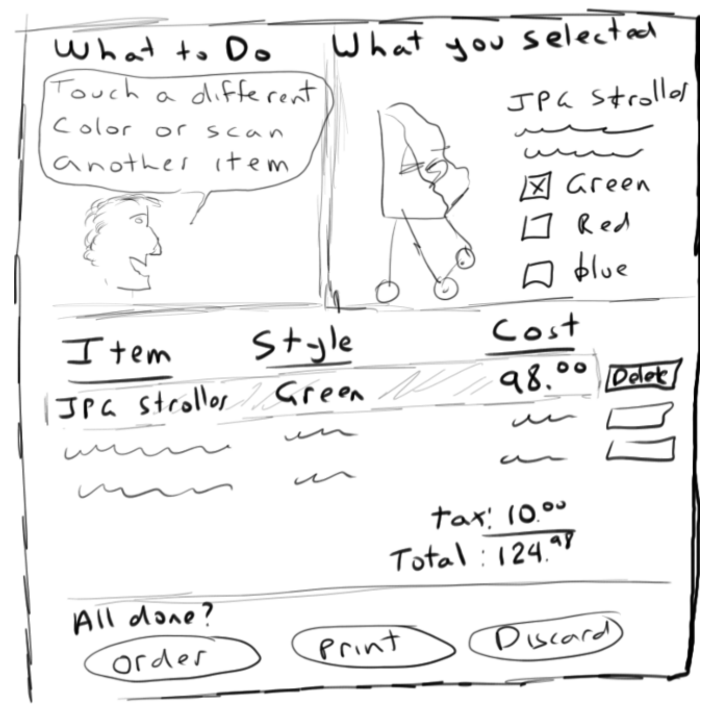
Low Fidelity Prototypes

- 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



Interface of a proposed system



A sketch of the interface

Low Fidelity Prototypes

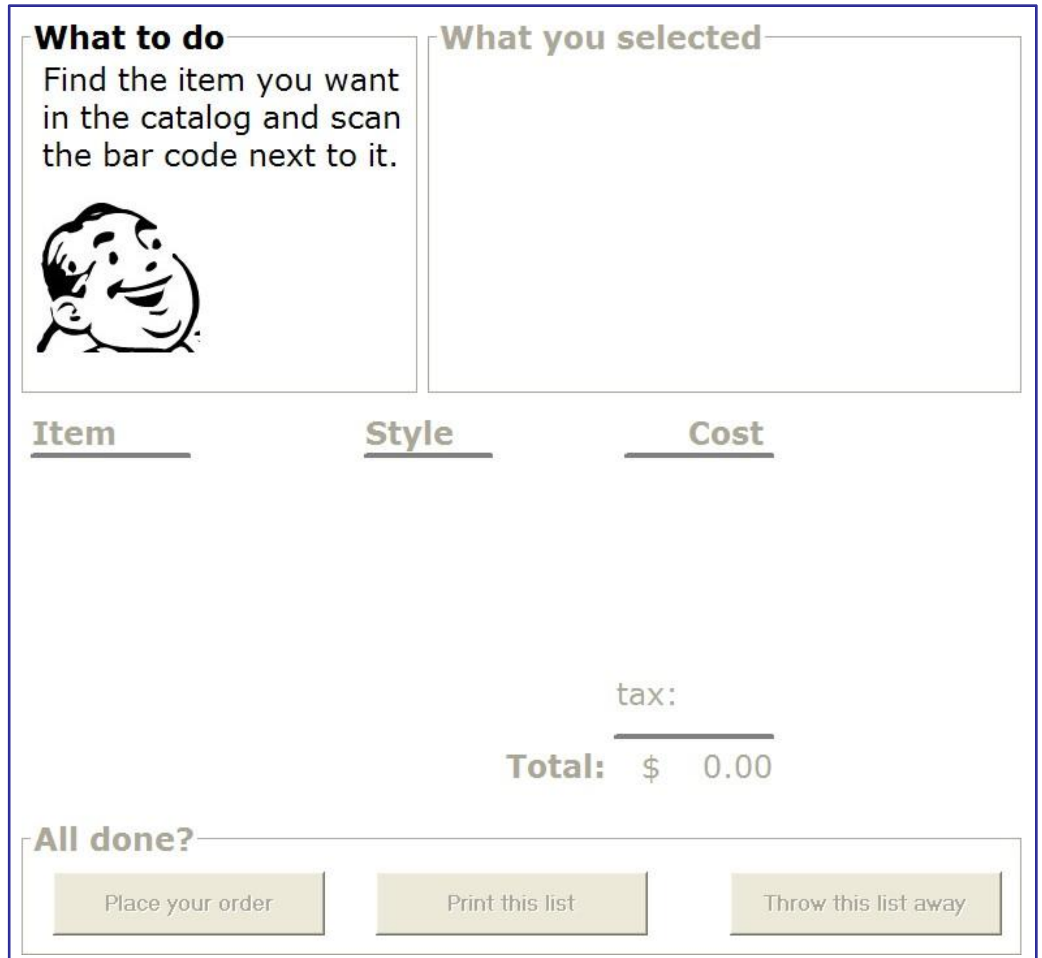
- In a sketch, the outward appearance of the intended system is drawn
  - Typically a crude approximation of the final appearance
- Such crude approximation helps people concentrate on high level concepts
  - But difficult to visualize interaction (dialogue's progression)

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

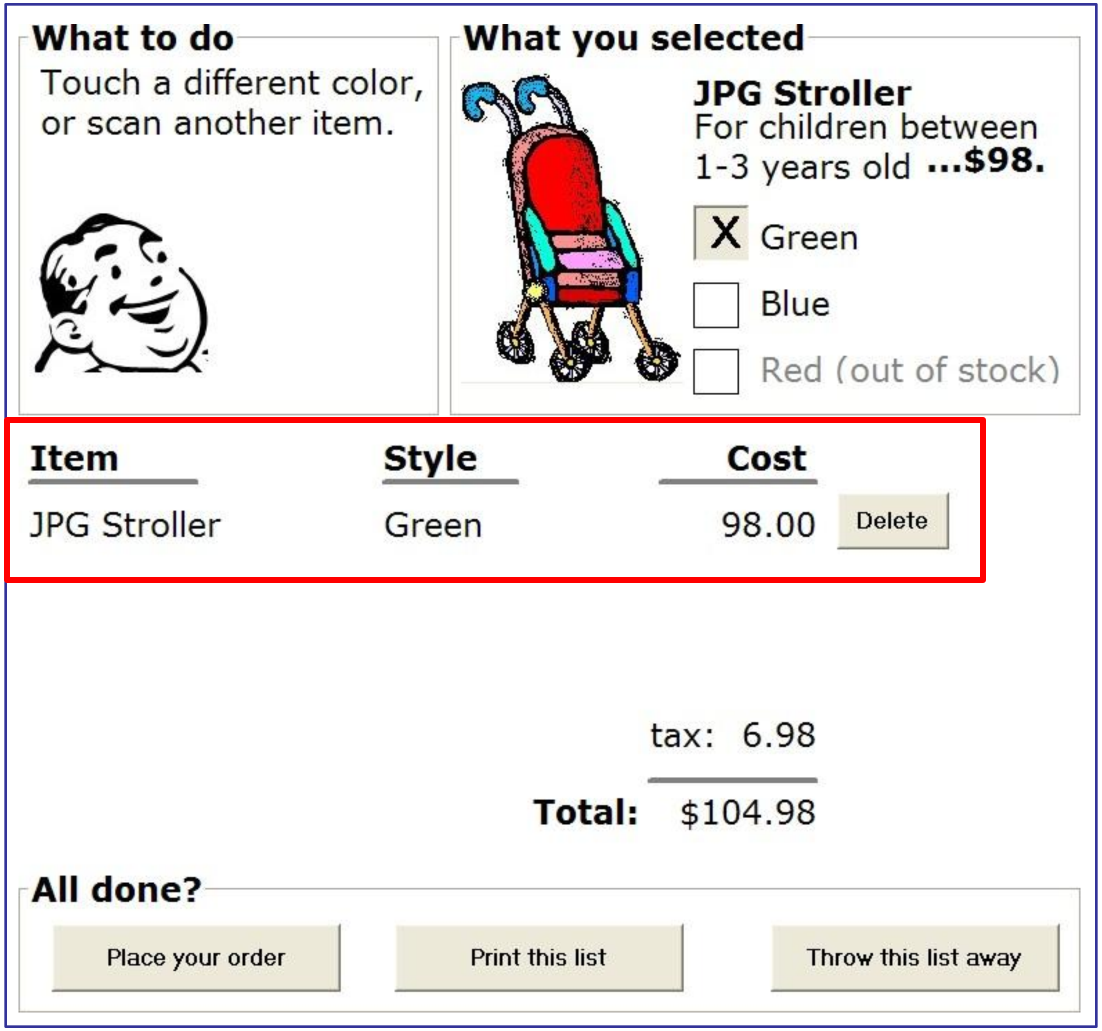
Low Fidelity Prototypes: Storyboarding

Initial screen. Shows the layout of interface options.



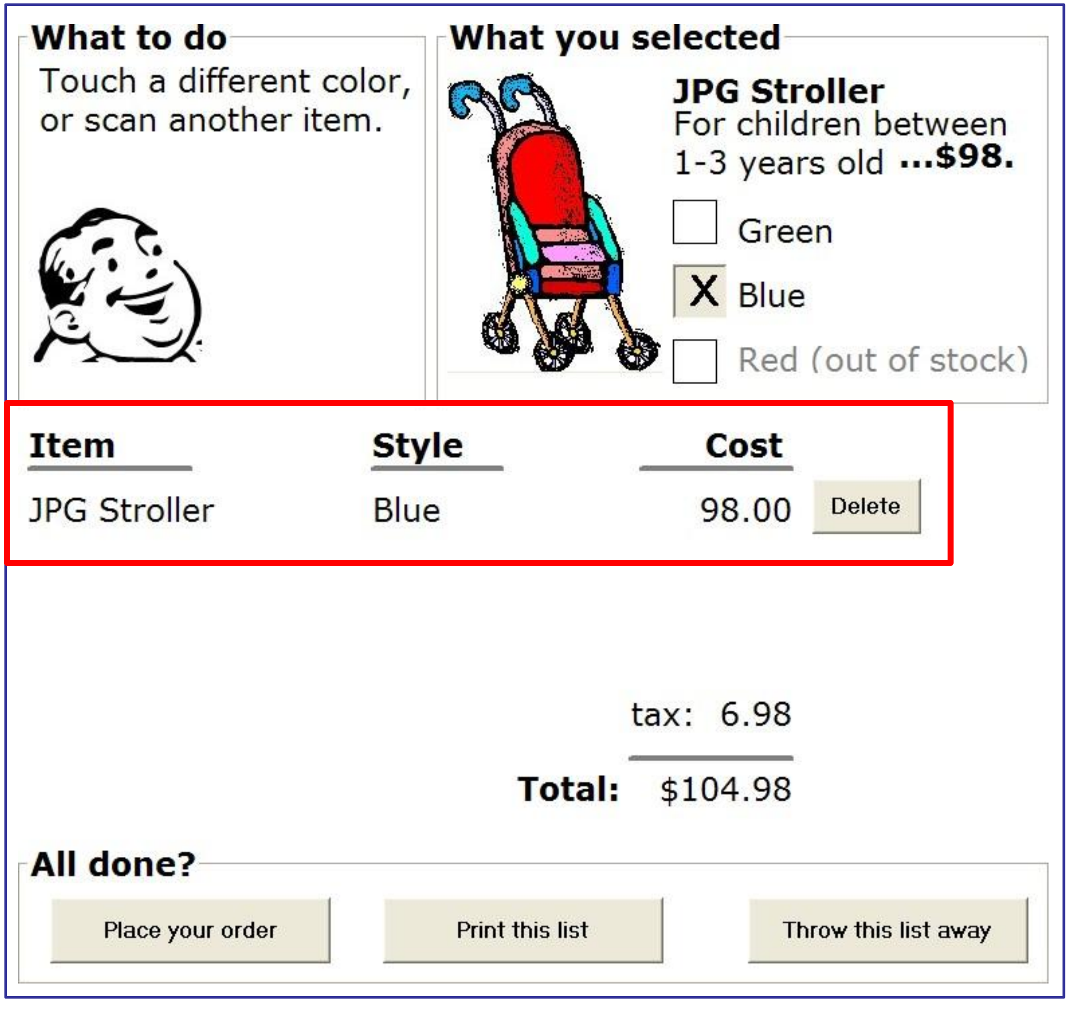
Low Fidelity Prototypes: Storyboarding

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.



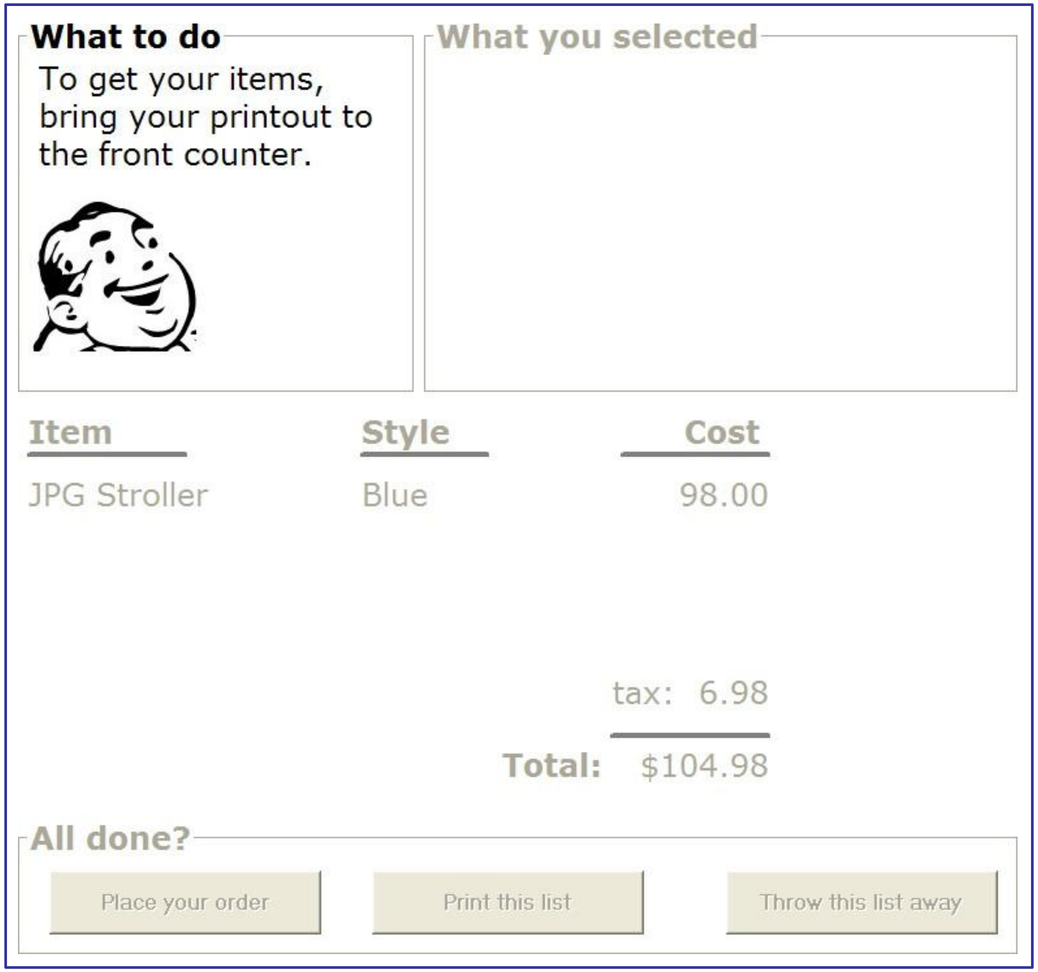
Low Fidelity Prototypes: Storyboarding

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.



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.



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 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.

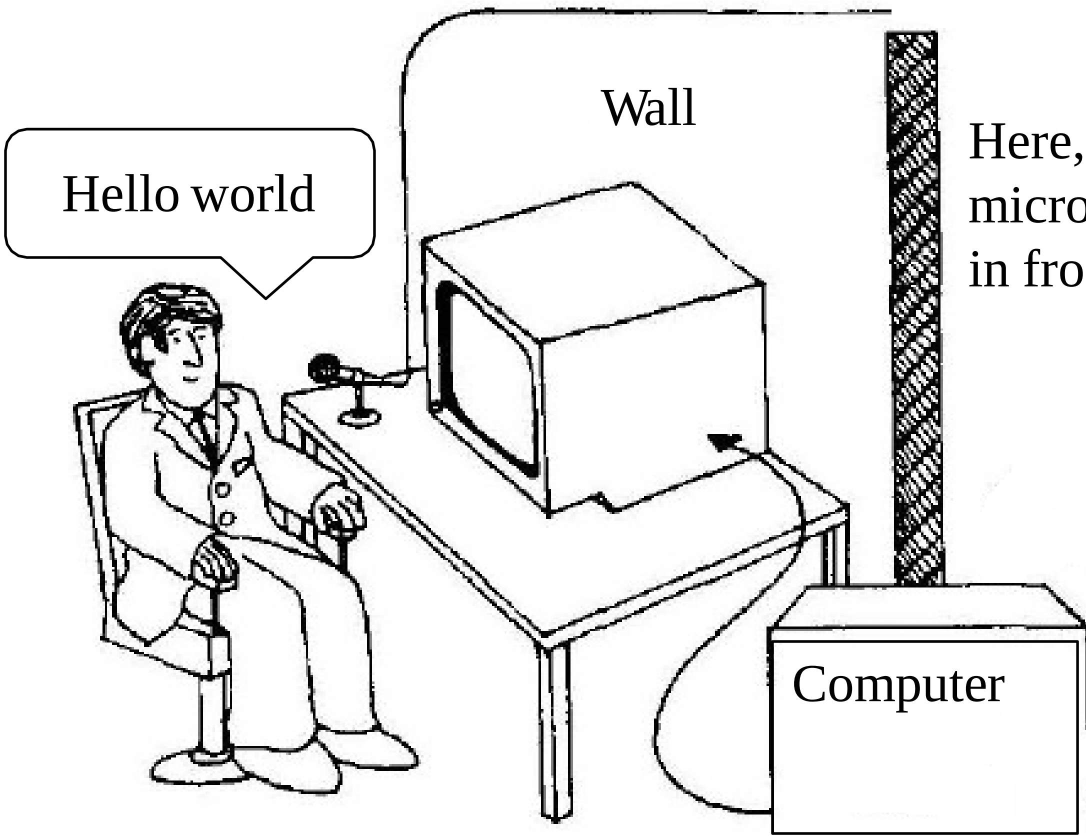
The Wizard of Oz Technique

- 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”

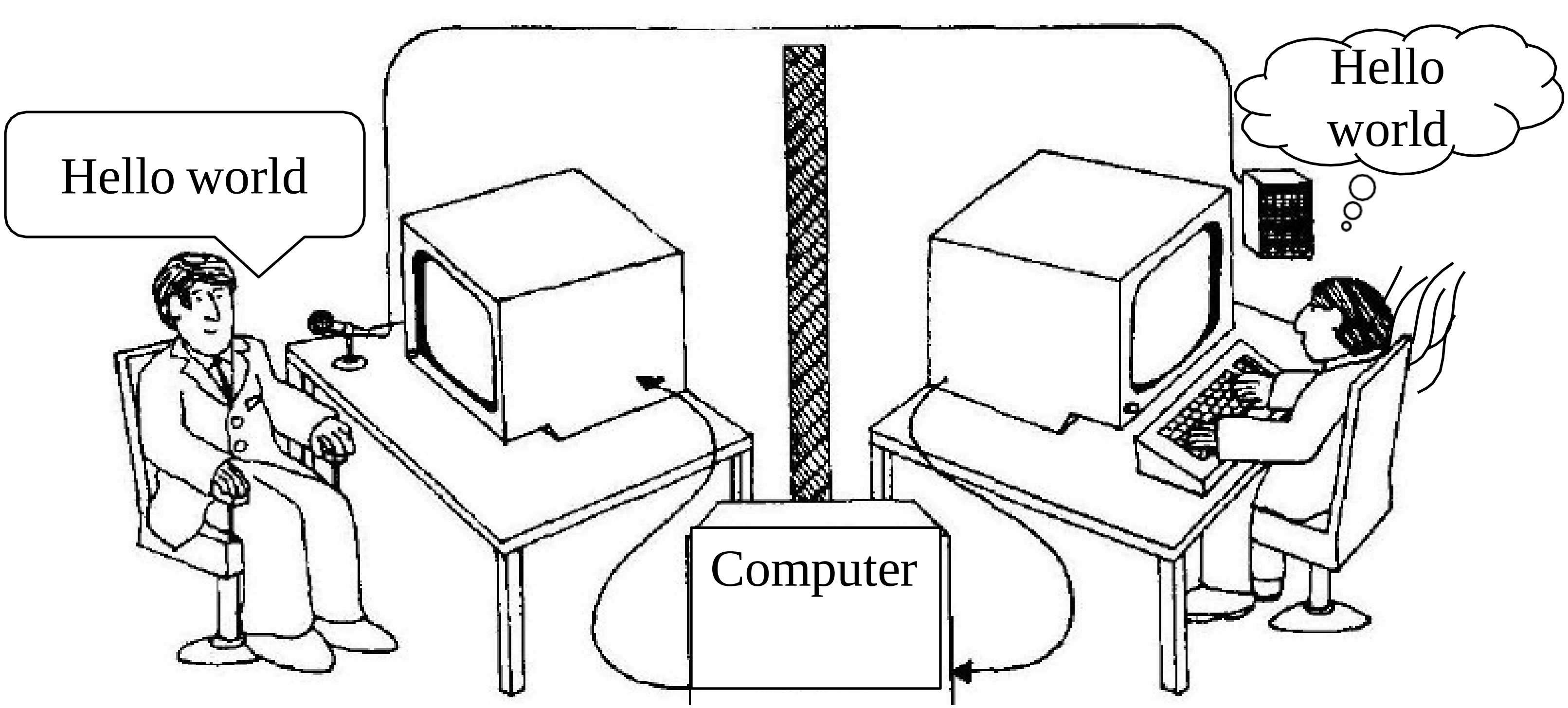
The Wizard of Oz Technique



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: 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* ***G****oals.*
- A *Method is a sequential list of* ***O****perators that the user performs and (sub)Goals that must be* achieved
- If there is more than one ***M****ethod which may be employed to achieve a Goal, a* ***S****election 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
  - **K**eystroke **L**evel **M**odel or KLM
  - Original GOMS proposed by **C**ard, **M**oran and **N**ewell, popularly known as (CMN) GOMS
  - **N**atural GOMS **L**anguage or NGOMSL
  - **C**ognitive **P**erceptual **M**otor or (CPM)GOMS [also known as **C**ritical **P**ath **M**ethod 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
K	The motor operator representing a key-press
B	The motor operator representing a mouse-button press or release
P	The task of pointing (moving some pointer to a target)
H	Homing or the task of switching hand between mouse and keyboard
D	Drawing a line using mouse (not used much nowadays)

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:
    - $\mathbf{T_{execute} = T_K + T_B + T_P + T_H + T_D + T_M + T_R}$
- **Heuristics for adding (handling) M**

KLM- GOMS: Operator Times

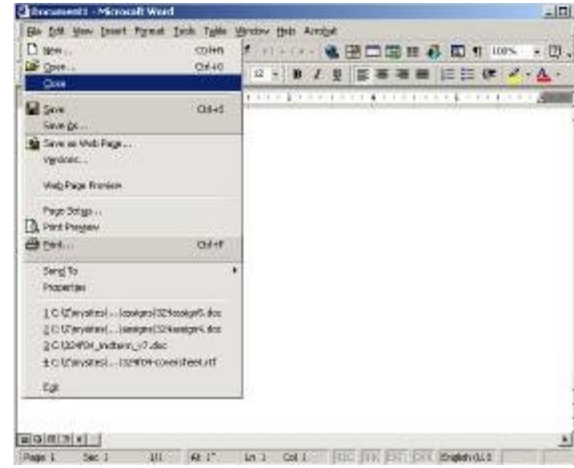
Operator	Description	Time (sec)
K	press a key or button (shift or control key count separately) best typist (135 wpm)	.08
	good typist (90 wpm)	.12
	average typist (55 wpm)	.22
	average typist (40 wpm)	.28
	typing complex codes	.75
B	click (press or release) a mouse button	.10/.20
P	point with mouse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

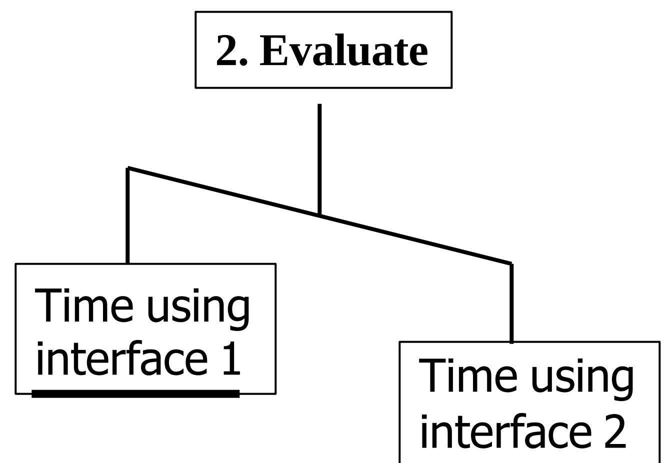


```
GOAL: ICONISE-WINDOW
[select
   GOAL: USE-CLOSE-METHOD
   .     MOVE-MOUSE-TO- FILE-MENU
   .     PULL-DOWN-FILE-MENU
   .     CLICK-OVER-CLOSE-OPTION
   GOAL: USE-CTRL-W-METHOD
         PRESS-CONTROL-W-KEY]
```

  - **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**

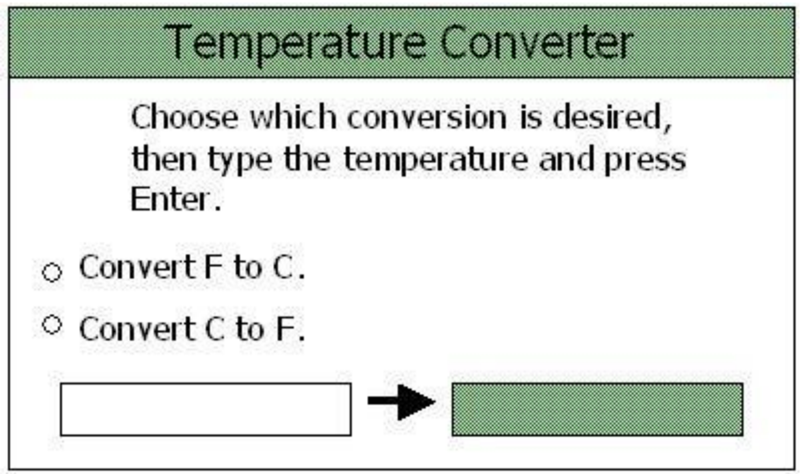


KLM- GOMS: Handling M

- **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 infront 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

HMPMBHMKMKMKMKMK — 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

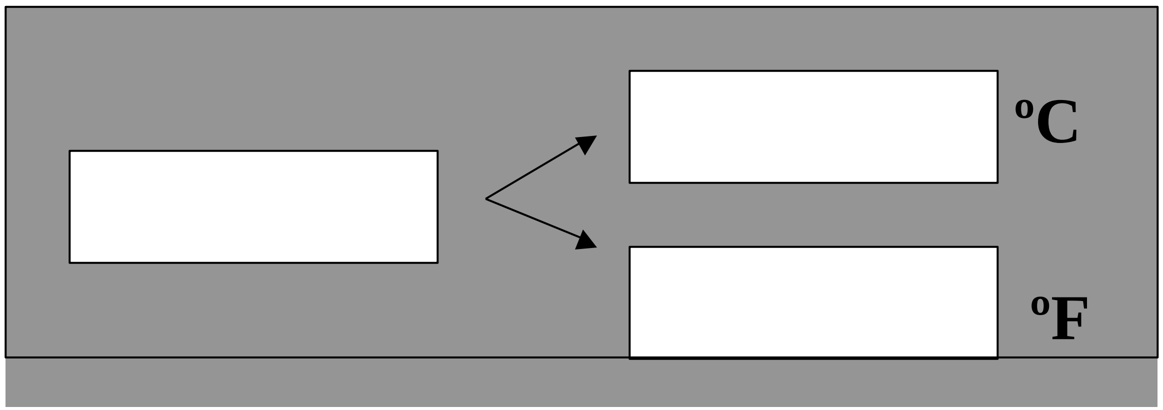
KLM: Handling M – An Example

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

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

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, **M**oran and **N**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 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 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 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 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 accomplice 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

- [SelectGoal: 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

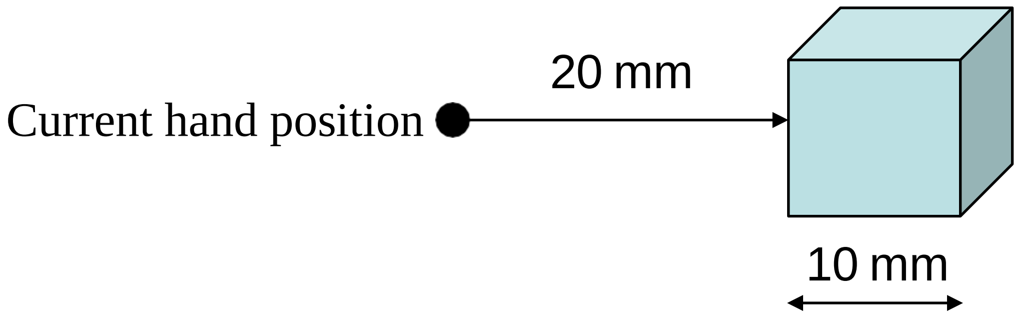
- From the analysis of empirical data, Fitts' law proposed the following relationship between three factors, namely: ID, D and W

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

(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 mm, W = 10 mm
- Thus, ID = $\log_2(20/10+1)$

  $= \log_2(2+1)$

  $= \log_2 3 = 1.57$ 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
			Mean	2.57

Each value indicates mean of 10 participants

The 6 tasks with varying difficulty levels

Throughput = 2.57 bits/sec (bps)

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 TP1 and TP2 for the mouse and the touchpad experiments, respectively
- You are asked to Compare TP1 and TP2
  - If TP1>TP2, the mouse gives better performance
  - The touchpad is better if TP1<TP2
  - Performance-wise they are the same if TP1=TP2 (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 α 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, pi be the probability of making the i^{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 = 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, $T = k \log_2 10$

    $= 0.66$ ms (assuming a = 0, b = 0.2)

Case Studies on Model-Based Design

Professor Ram Mohana Reddy Guddeti
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

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

MTij is the movement time from the source (i-th) to the target (j-th) key, dij is the distance between the source and target keys, wj 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

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

  - Pij is the probability of occurrence of the i-th and j-th key, whereas fij 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 ΔE indicates the energy difference between the new energy and old energy states
  (ΔE = energy of new state – energy of old state)

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

Metropolis Algorithm

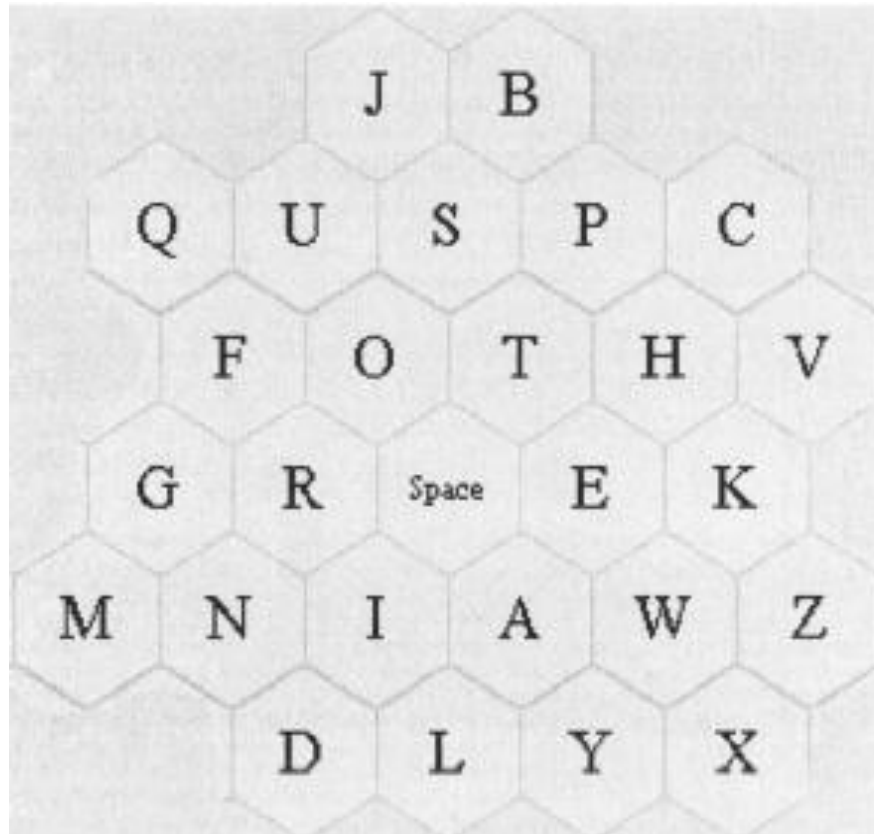
- W is probability of changing from old to new configuration
- k is a coefficient
- T is “temperature”
- Initial design: a “good” layout stretched over a “large” space

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 struck at the local minima further, "Simulated Annealing" is used
  - Bringing "temperature" through several up & down cycles

Metropolis Algorithm

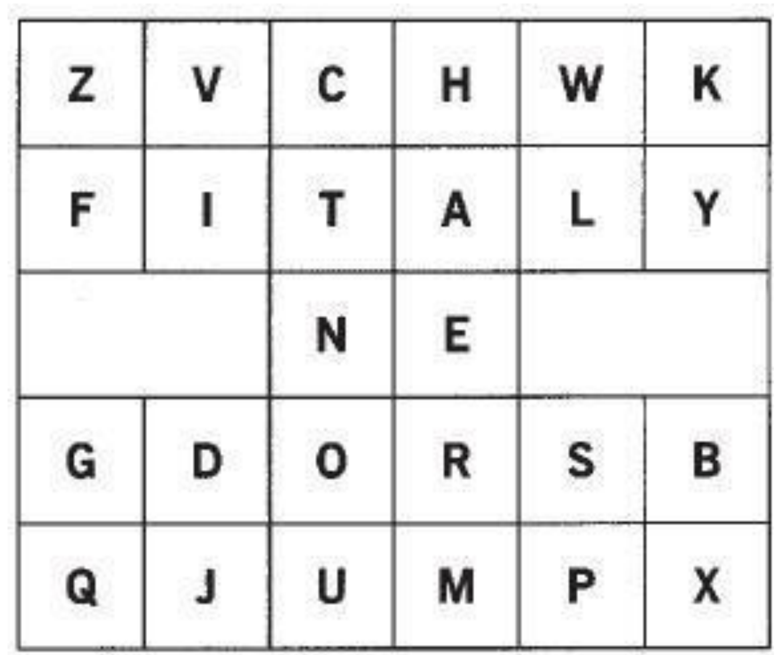
An example VK layout, called the Metropolis layout, is shown, which was designed using the Metropolis algorithm



Some VK Layouts with Performance

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

FITALY Keyboard



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: 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.

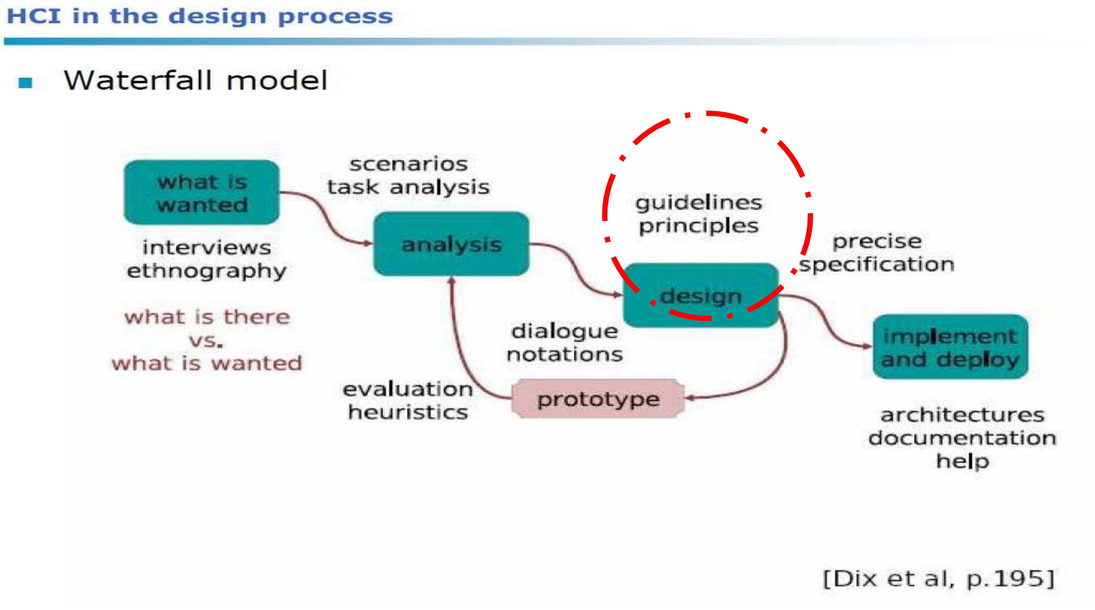
The following are the standard guidelines

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.



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

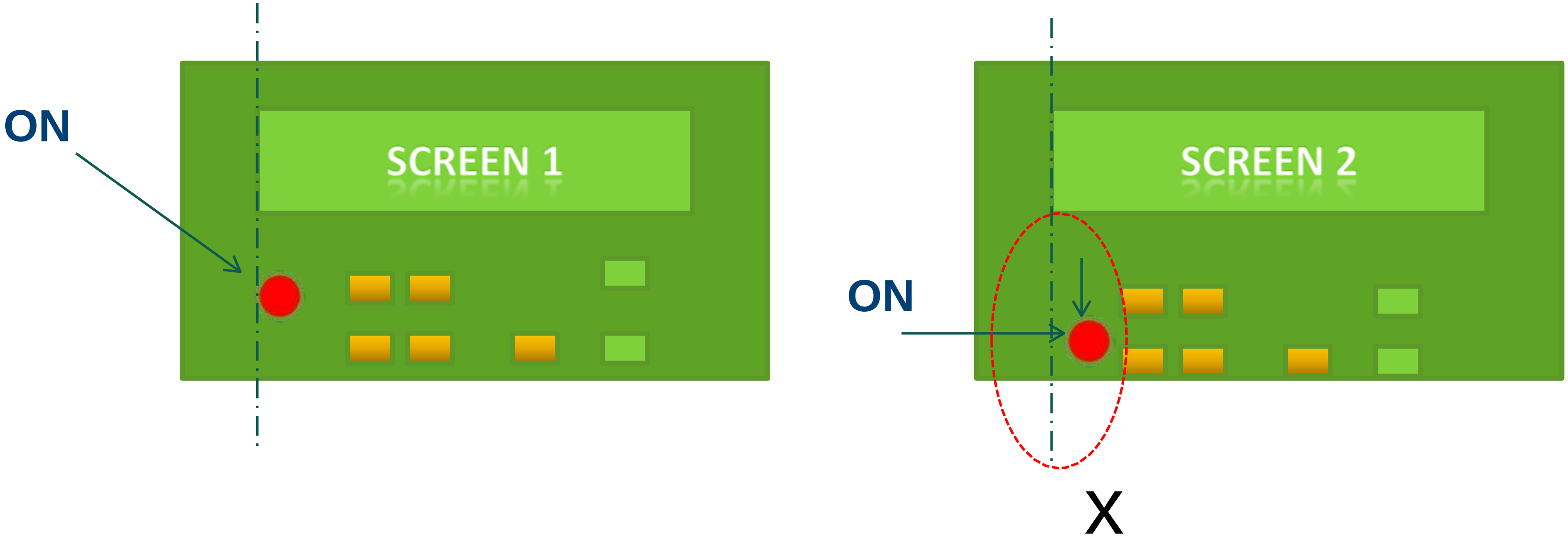
Explanations & Examples

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.

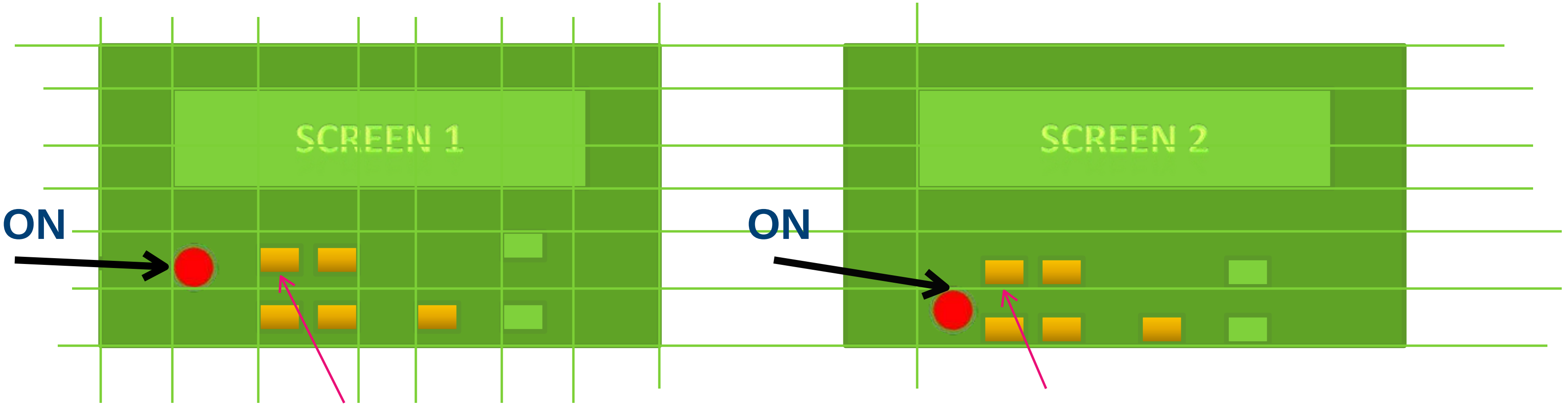
For example: If the **ON** button is on the right in the first screen and moves towards middle in the second screen then positional inconsistency is said to have occurred - however small the displacement is.



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 a 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, Shniderman's rule none the less needs to be followed so as not to leave out taking into consideration a section of users, other wise 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

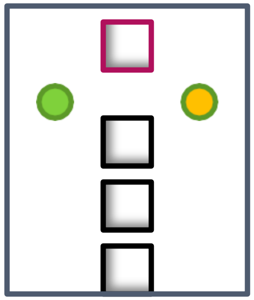
- Interfaces need to not just to be communicative but also need to inform the 'user' in terms of learning & feed back which tells them that they are proceeding in the right direction.
- For every action of the user there needs to be a feedback – only then 'interaction' (in HCI) is said to take place. 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**.**

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 3rd button and continue till green lamp stops glowing.

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

An Example of a closed dialogue:

Press ON button

Look at the green lamp.

If green glows press 2nd push button and yellow lamp will glow.

Press 3rd button and continue with other 3 buttons till green lamp stops glowing.

When Yellow lamp stops glowing it indicates sequence over.

End of task.

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 cloud be reasons for users 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.

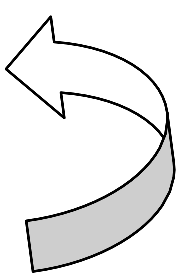
Even if the user makes an error the system needs top be designed to detect it, take corrective or precautionary steps to arrest it. It also needs to offer a way out for recovery from the error.

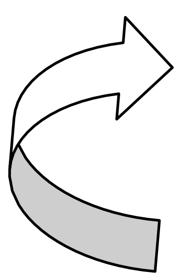
A default system unchanged message needs to be communicated to the user if an error has happened.

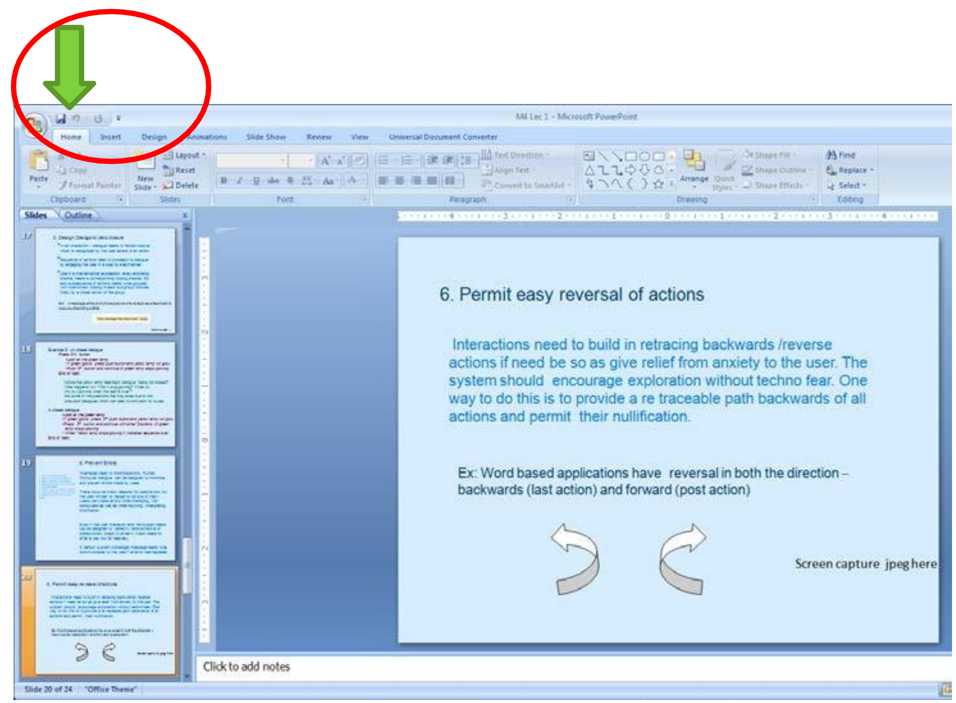
6. Permit Easy Reversal of Actions

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)





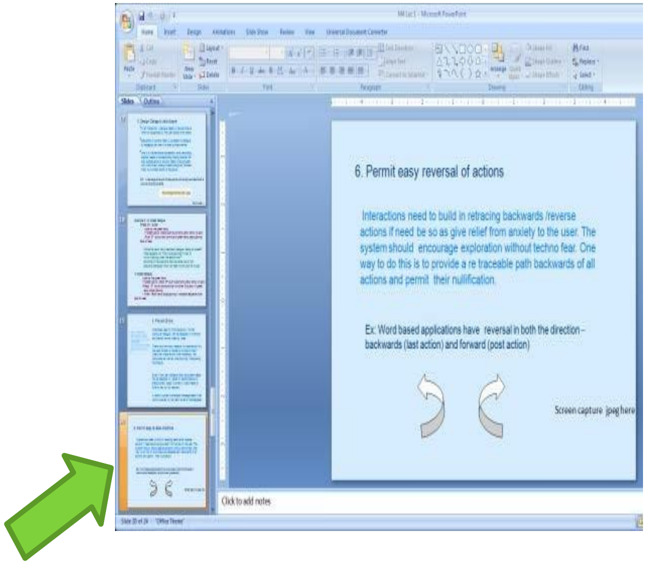


7. Support Internal Locus of Control

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**

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.

94 56 781029

Easier to remember if chunked into smaller sets

94 56 7 810 29

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 +- 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: 10th 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 Catherikne Plaisant, Addison-Wesley Publishers Treading MA. 2010)

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

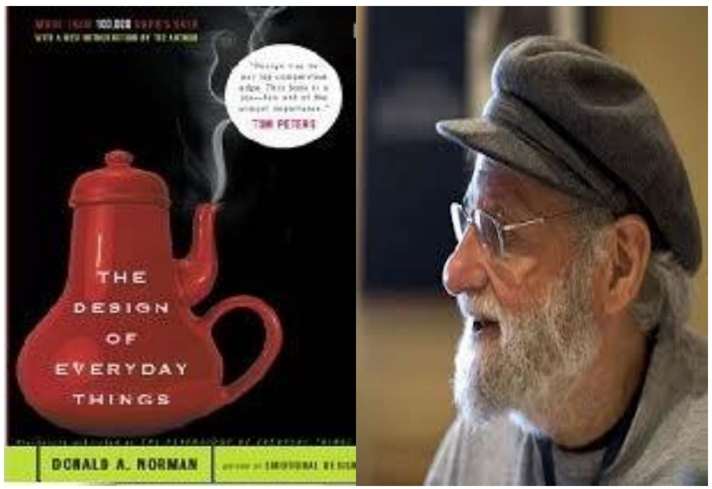
Professor Ram Mohana Reddy Guddeti
Information Technology Department
NITK Surathkal, Mangalore, India

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 bellow. 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.

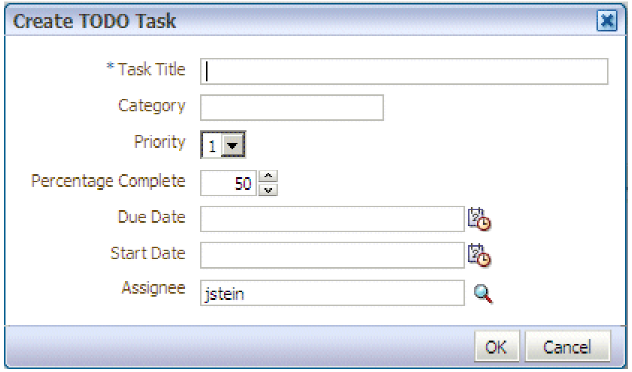
2. Simplify Task Structures.

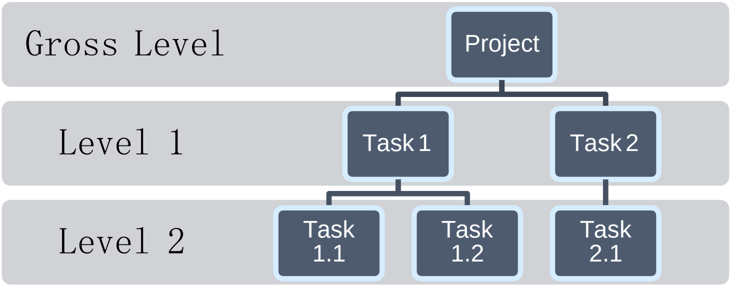
•**Task Level:** Task level is to analyze the user's needs and to structure the task domain in such a way, that a computer system can play its part. The task level describes the structure of the tasks which can be delegated to the computer system.

•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.

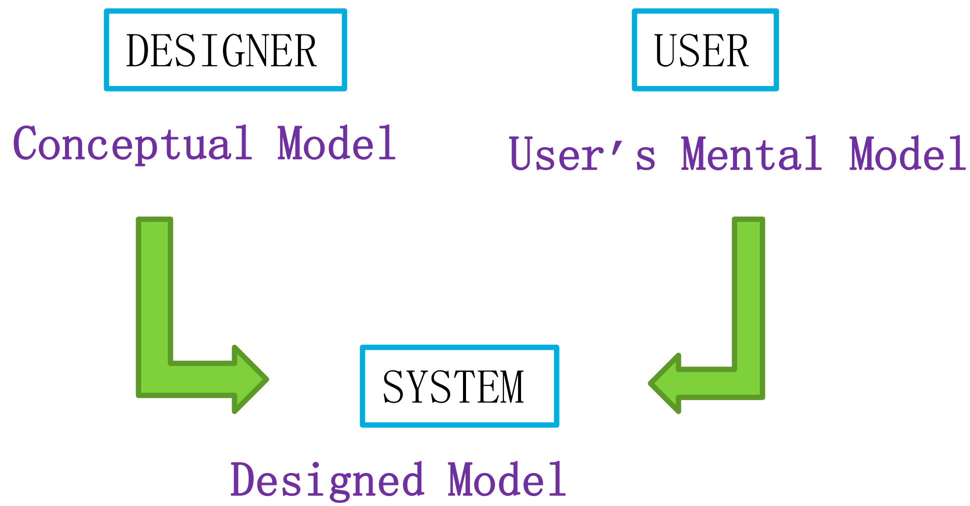
Interaction styles such as
WIMP (Windows - Icons - Menus - Pointer)
Three Dimensional Interfaces as in
Virtual Realty can be found in current software interfaces.

Example of ‘Visual’:
Windows 8 Interface.

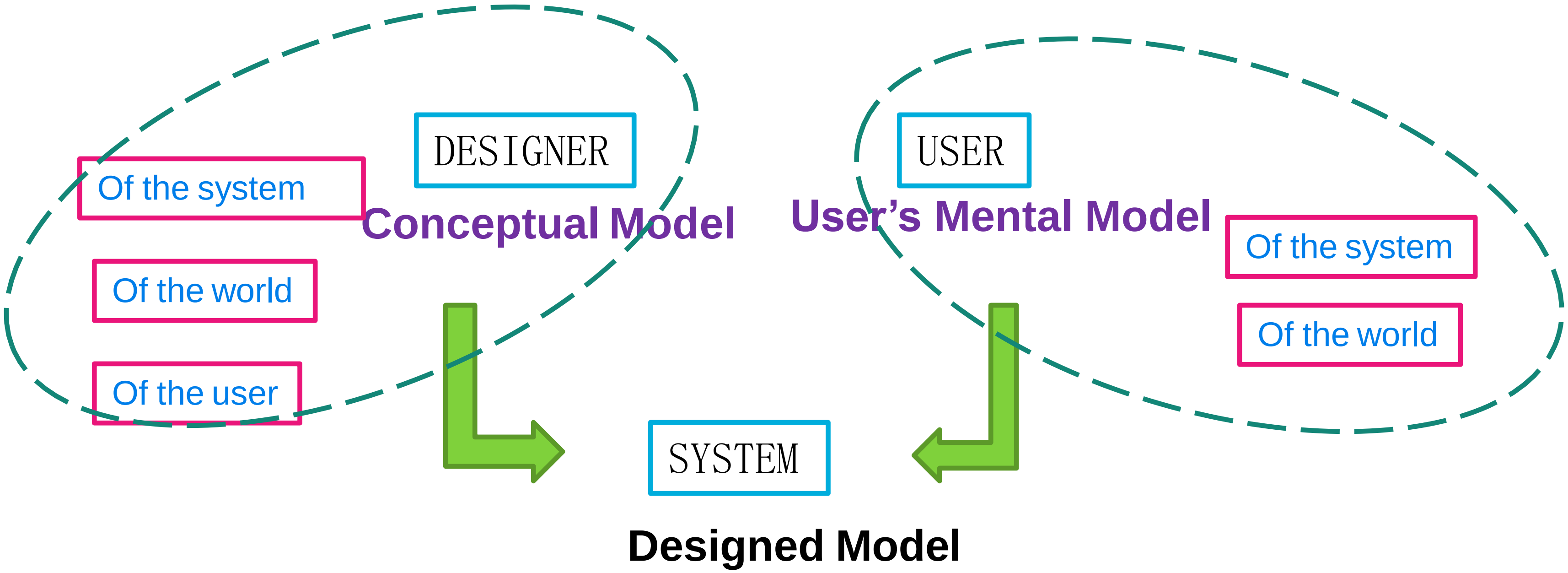
4. Get the Mapping Right

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.

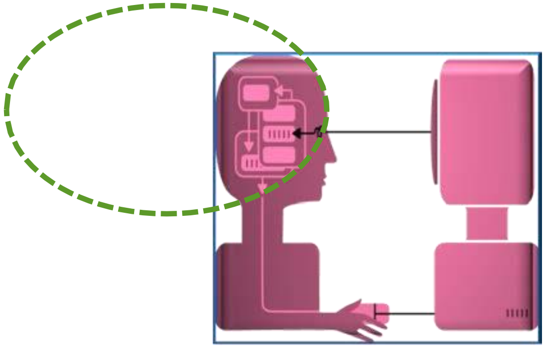


The **User's Mental Model** is the model of a system's working that a user creates when learning and using a computer. It is not technically accurate. It may also be not stable over time. User's mental models keep changing and evolving as learning continues.

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.

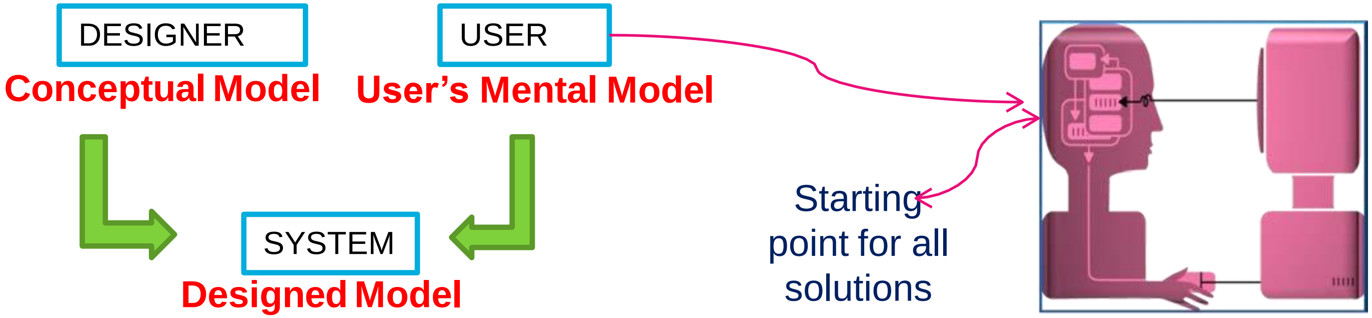


•This is a technically accurate model of the computer / device / system created by designers / teachers / researchers for their specific internal technical use.

•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.

Continued…



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)

Continued...

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 revels 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 a 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’.

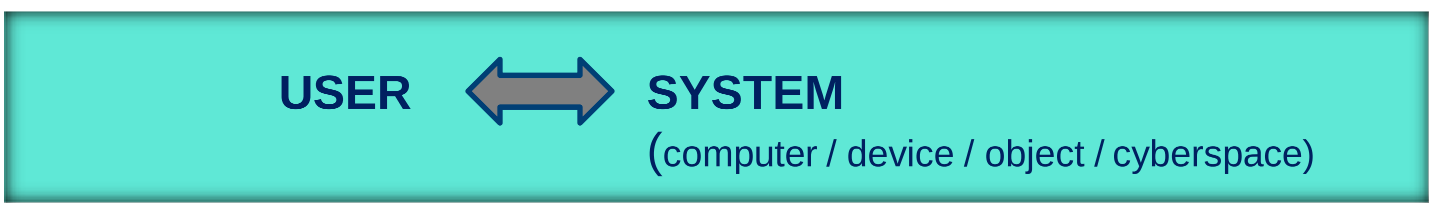
For example: Designers claim to be designing “Interactive web pages”.

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**

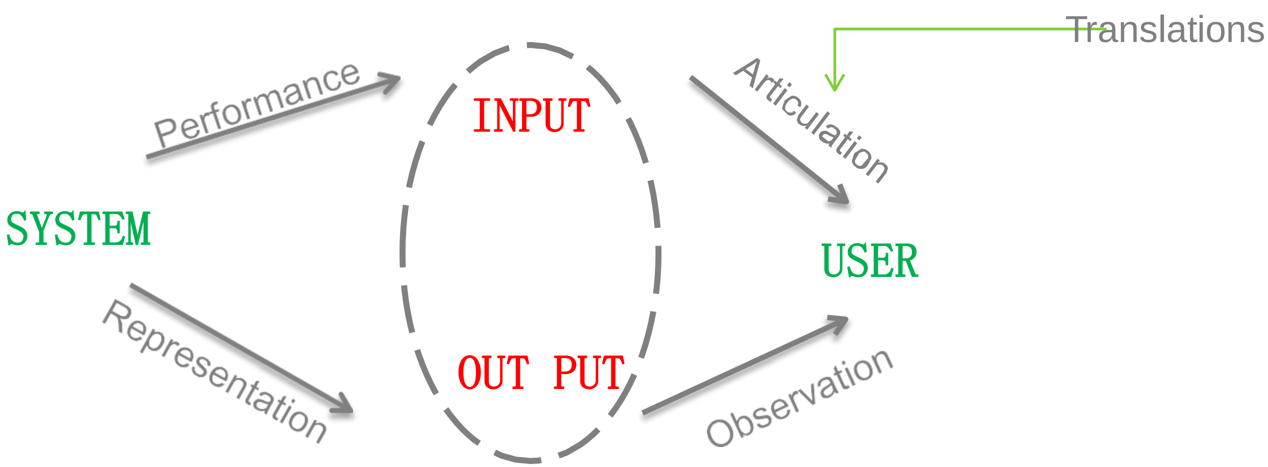
Task Analysis: **It is the study of the problem space**

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:

User to establish **Goal**

User to formulate **Intention**

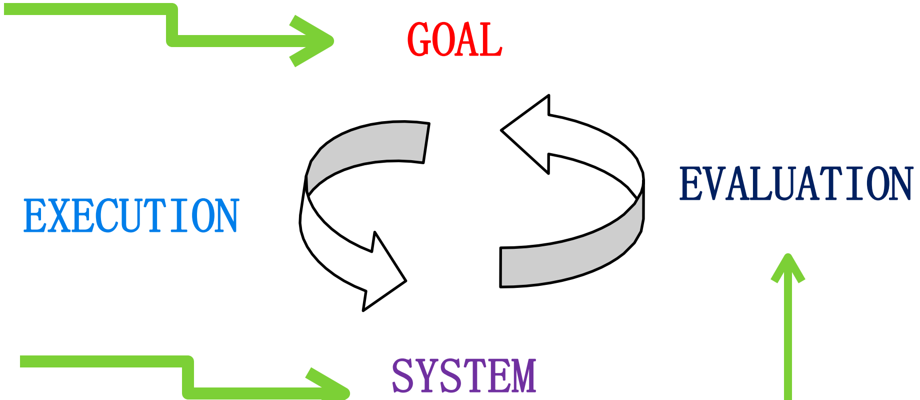
Decides on action

Executes the action

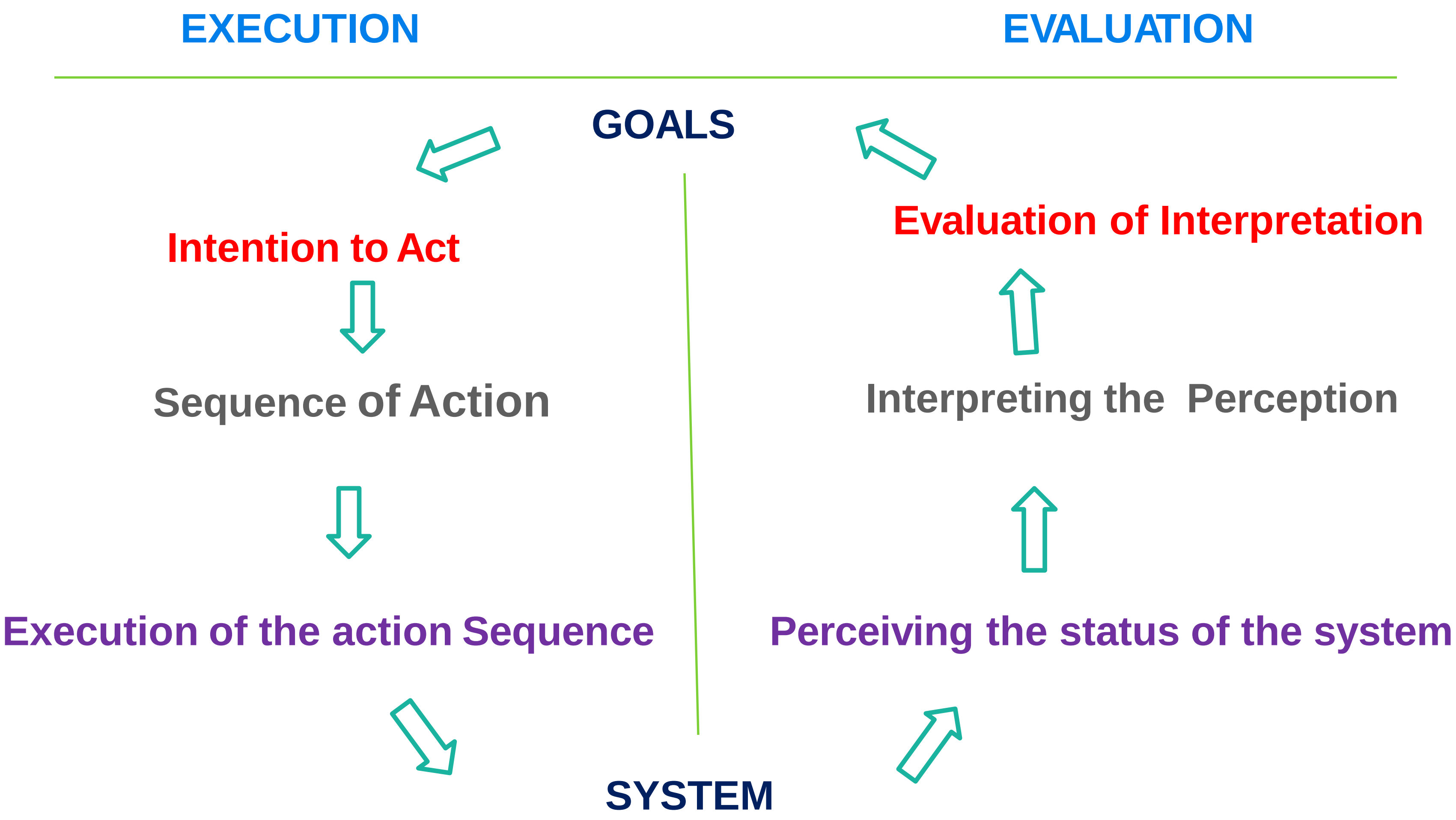
Perceives system state

Interprets system response

Evaluates system with respect to goal.

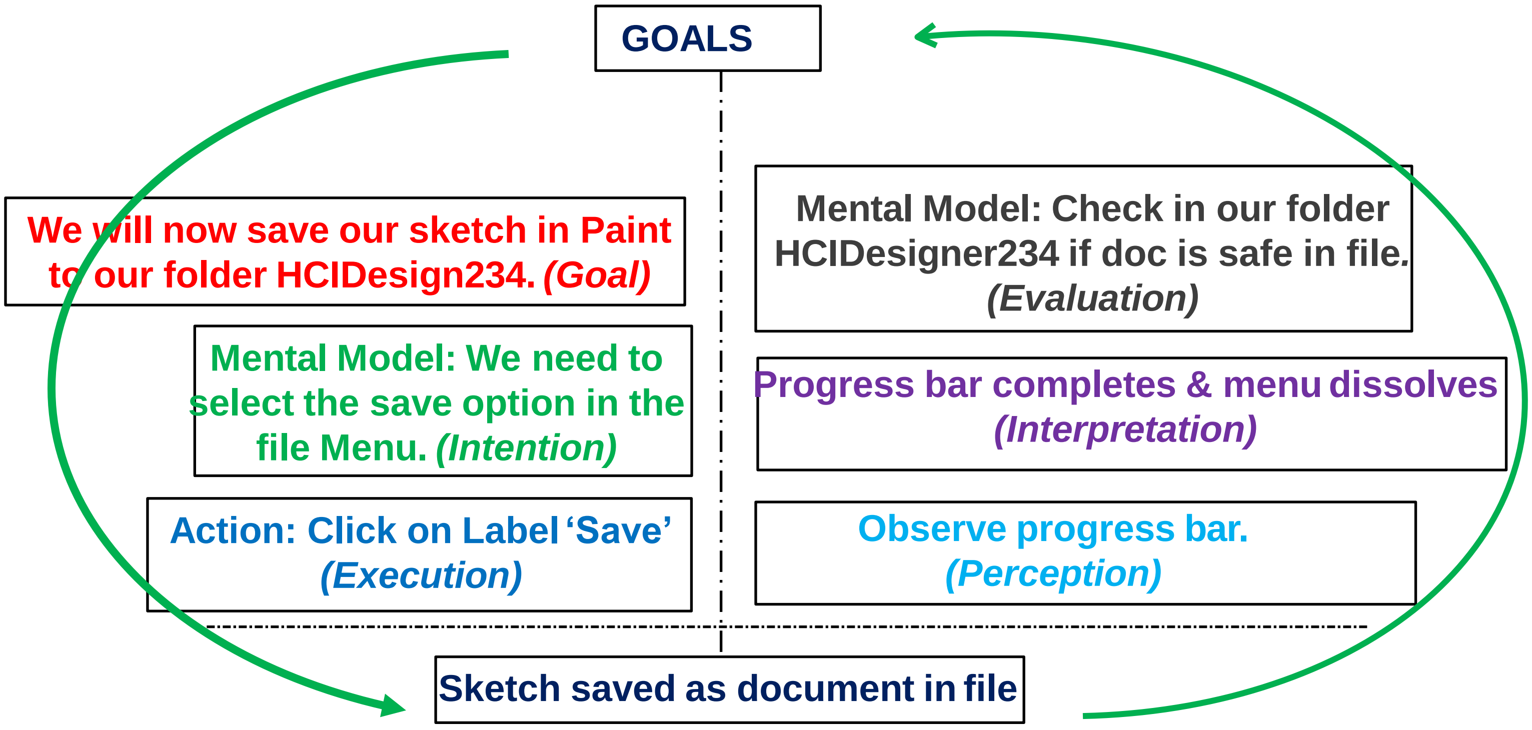


Another way of depicting Norman's 7 stage Action model



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 a 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.

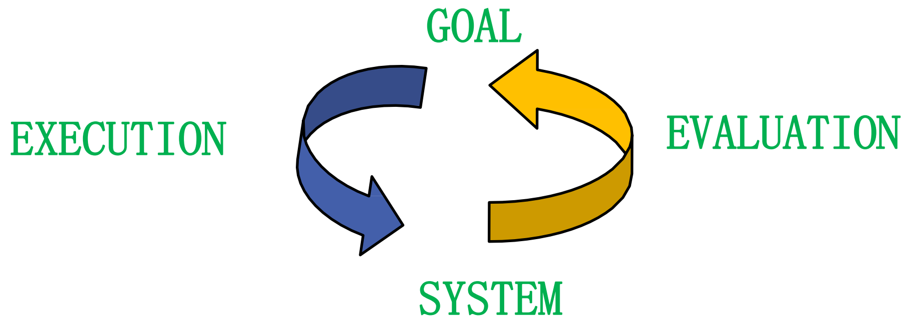
The Conceptualisation of the Designer had in his/her mind is called the "*design model*".

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 some times referred to as the Gulf Model) is useful in understanding the reasons of interface failures from the users point of view. The Seven stages of action model is an elaboration of the Gulf model.

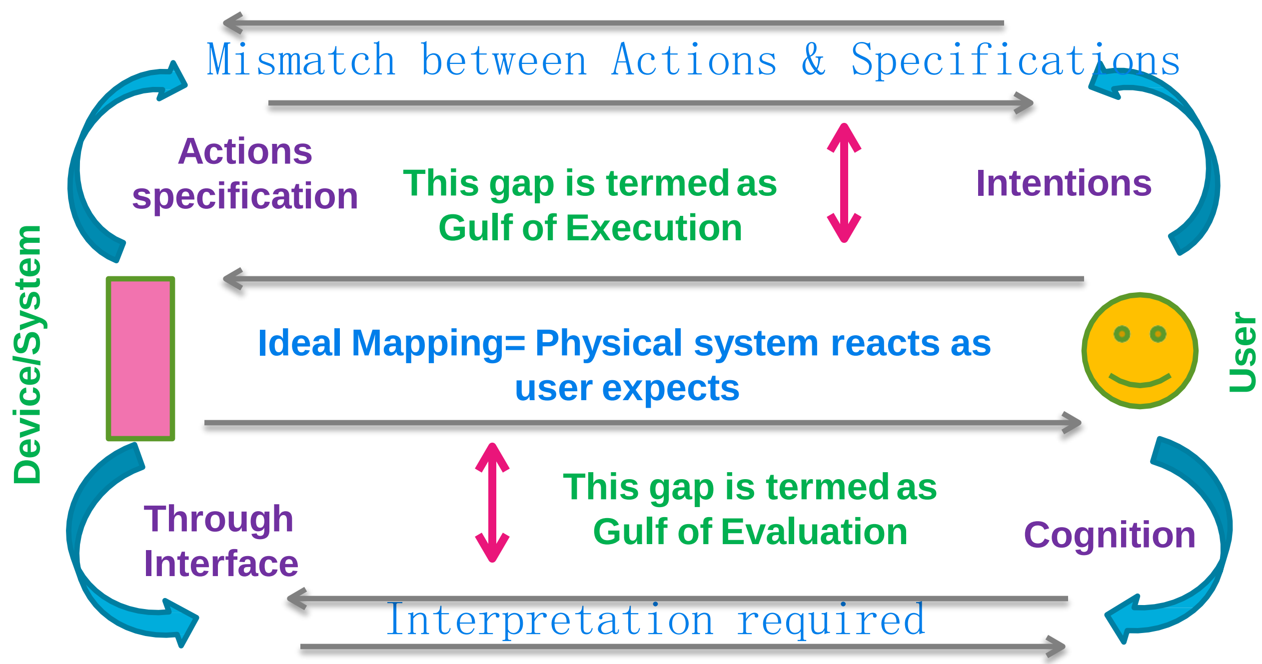


Gulf of Execution: It is the difference between user's formulation of the action to reach their goals and the actions allowed by the system.

User's formulation of action ≠ 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
- Vice 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, 3rd 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.

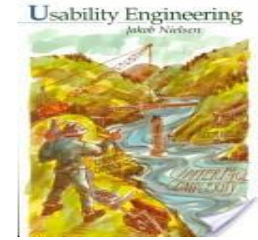
Since these ten guidelines were more in the spirit of "Rules of Thumbs" than the specific rules, they are referred to as 'Heuristics' rather than rules or laws that hold true in every case.

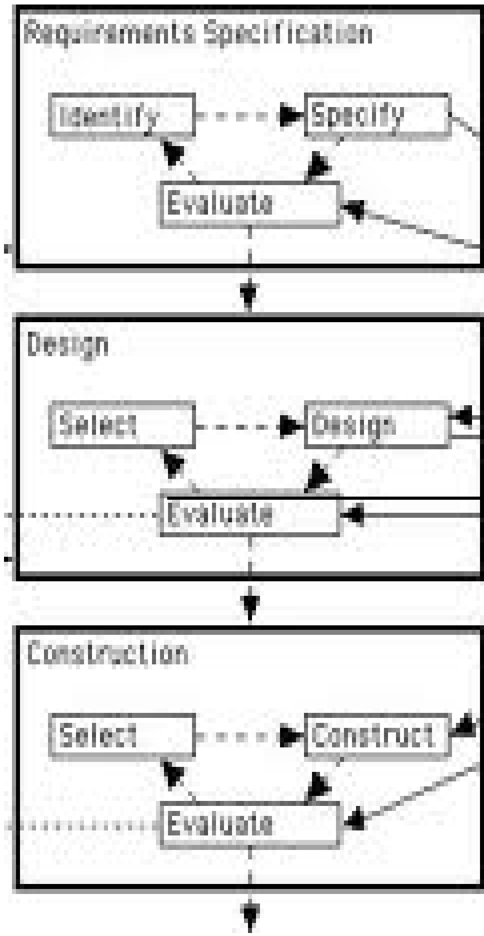
A heuristic is an approach to the problem solving or self discovery that employs a practical method that does not guarantee an optimal, perfect or rational solution, but produces a near optimal or an approximate or a short-term solution. Thus, heuristics can be mental shortcuts that ease the cognitive 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.

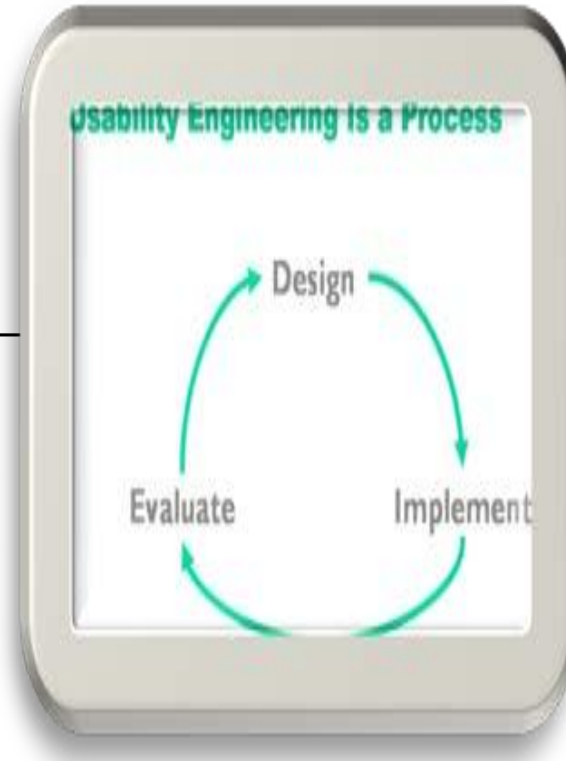




Usability Engineering involves

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.

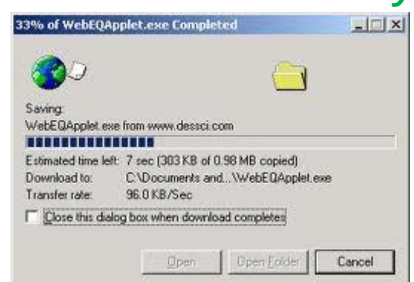
Visibility of System Status

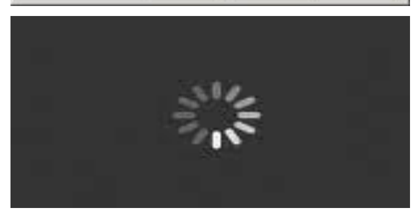
Users need to be kept informed by the system about what is going on, through appropriate feedback within reasonable time.

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.

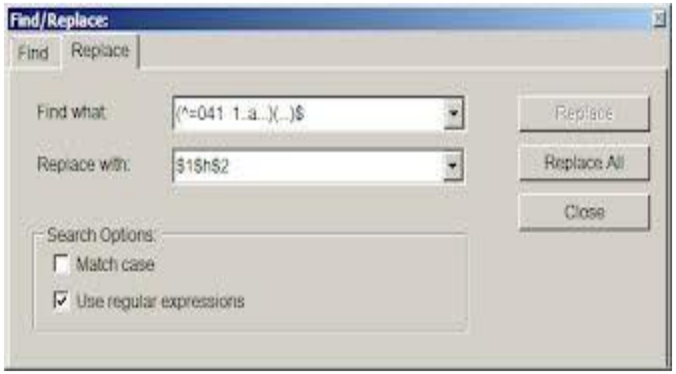
Match between System and the Real World

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 pace 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!

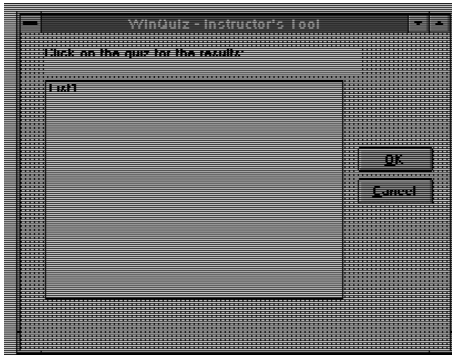
Can users select and sequence tasks? Can they easily return to where they were if they choose an inappropriate / action path? The first example "accuses" them of committing an error. The second example is much better but does not tell the user what to do next! The third example is inappropriate!

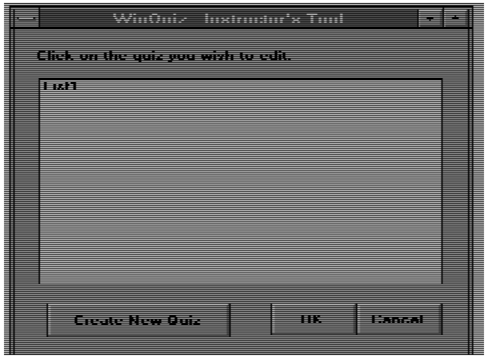
Consistency and Standards

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.



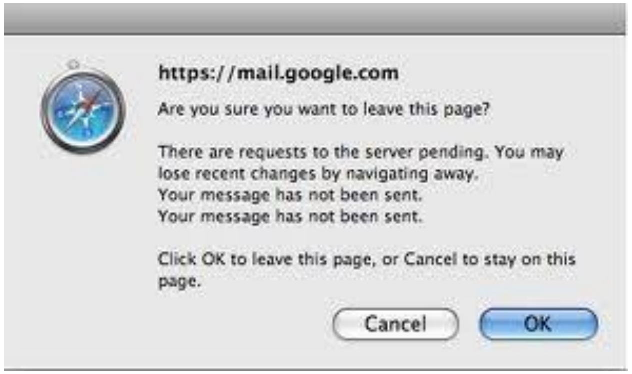


Inconsistent wording & windows / buttons can confuse users when the destination page has a different title from the link. The two screens belong to the same software but appear differently at different places within the website.

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'



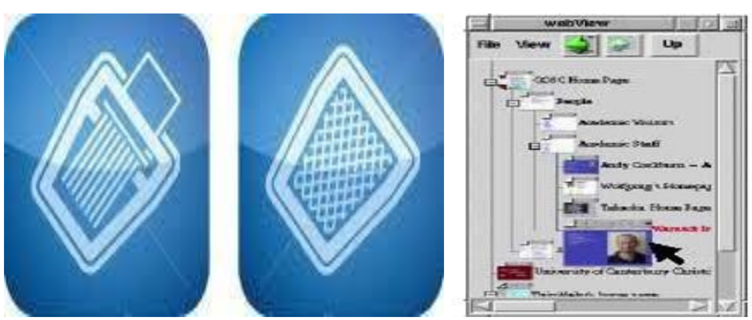
GUI-style widgets cut down on the errors but may still have to be double checked before confirmation

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.

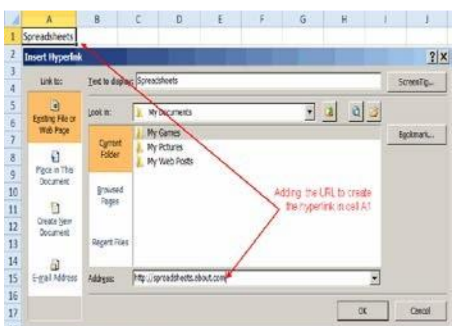


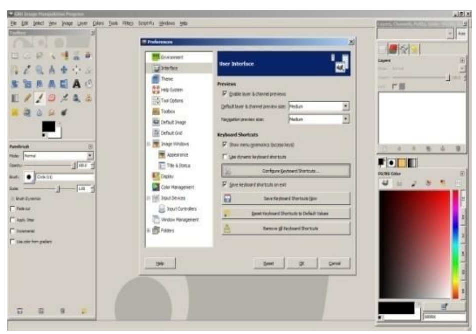
Good labels & the descriptive links are crucial for recognition. The first two icons as shown in the Fig. are difficult for the user to recognize or to recall. The third one helps the user to recognize where they are and recall which file is currently open.

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, un-cluttered 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 to7) 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

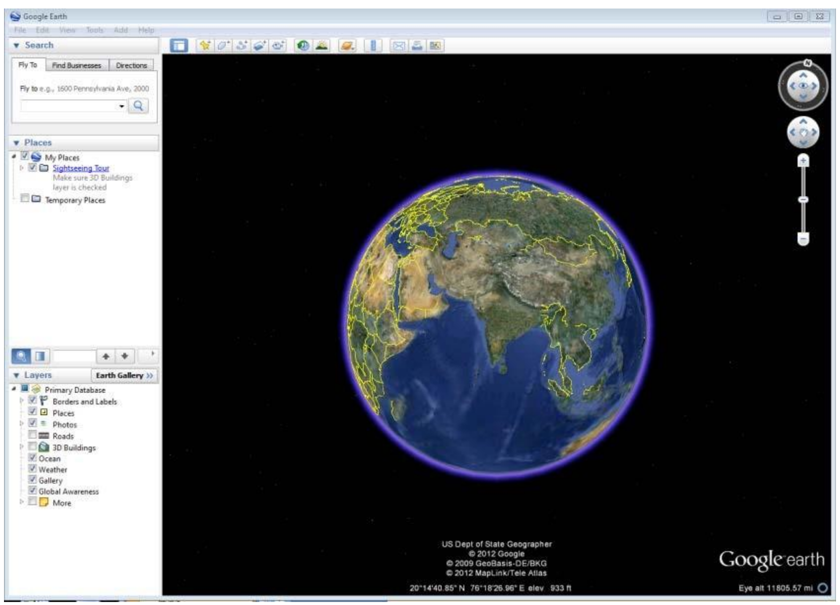
Case Study 1 **GOOGLE MAPS in Goggle Earth**

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.

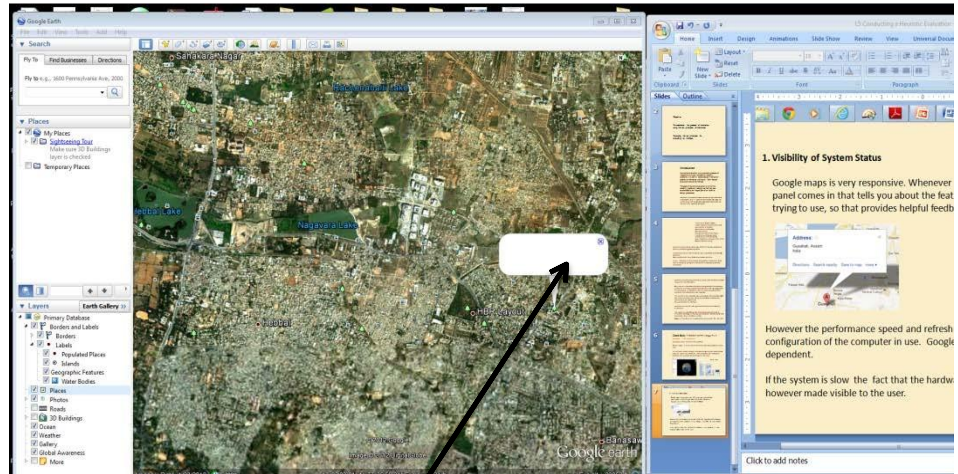


Controls & Views

1. Visibility of System Status

Findings of the expert on visibility & status capability of Google earth is explained bellow

"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."

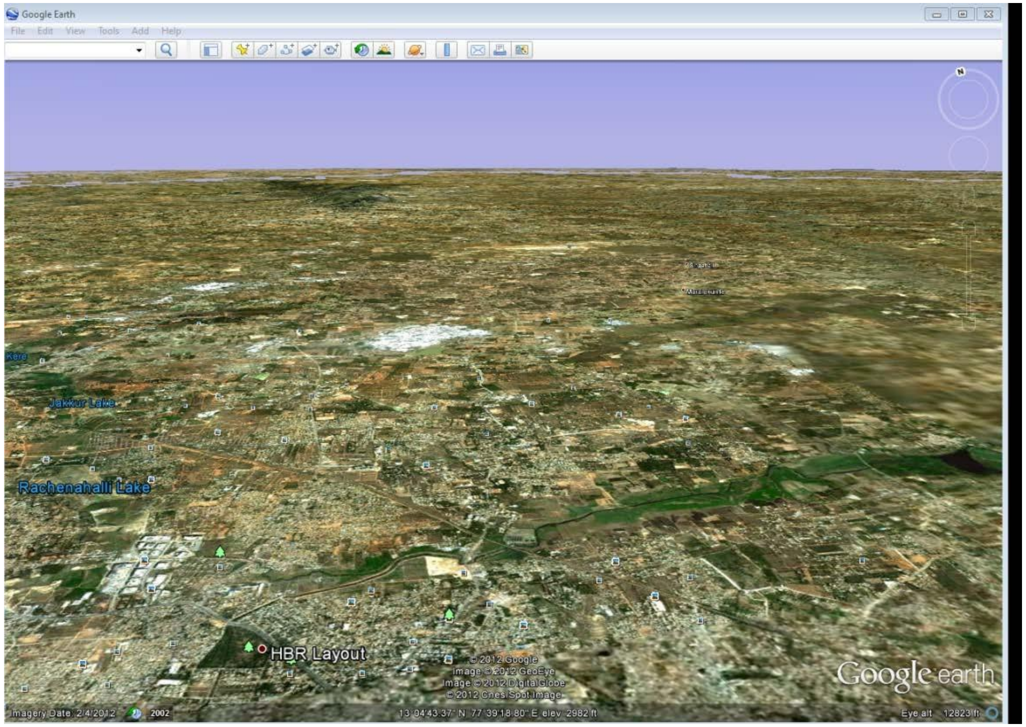


"System status if the Network connection is absent or suddenly is lost is not made visible to the user. The user sees a blank label" (second screen shot)

"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

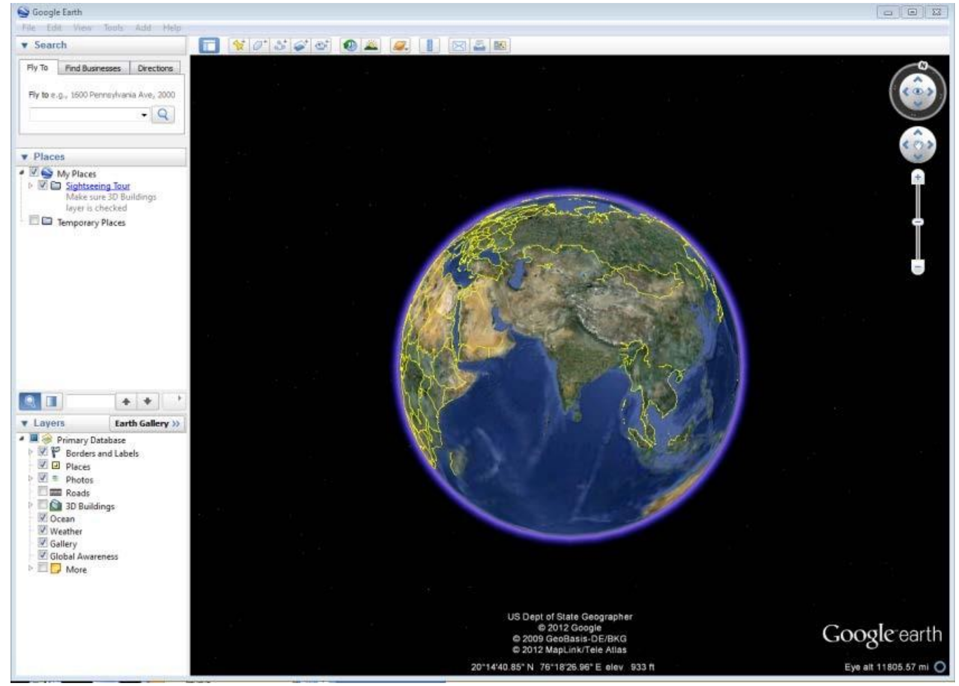


Closeness to Real-world is very good.

The window can be panned and horizon can be lowered giving a very Real-World view.

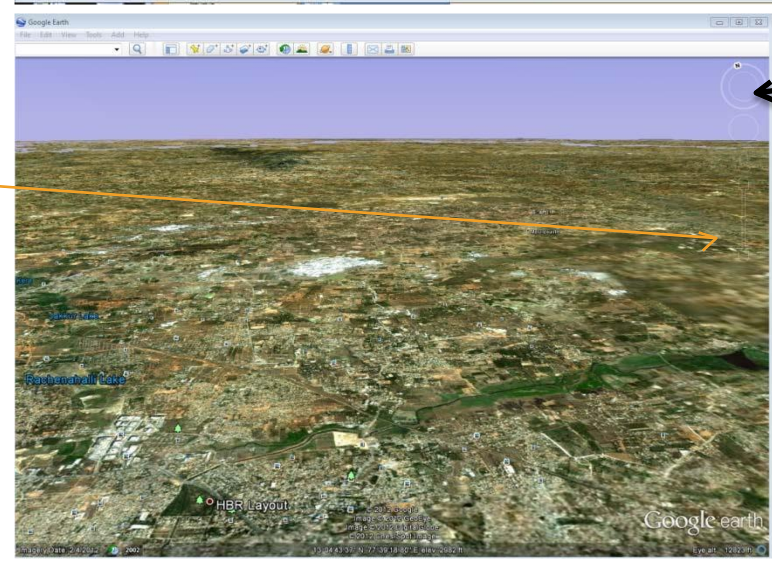
3. User Control and Freedom

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

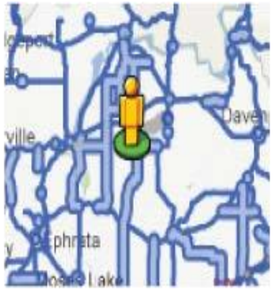
Google maps is pretty consistent with the words and phrases that they use.

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

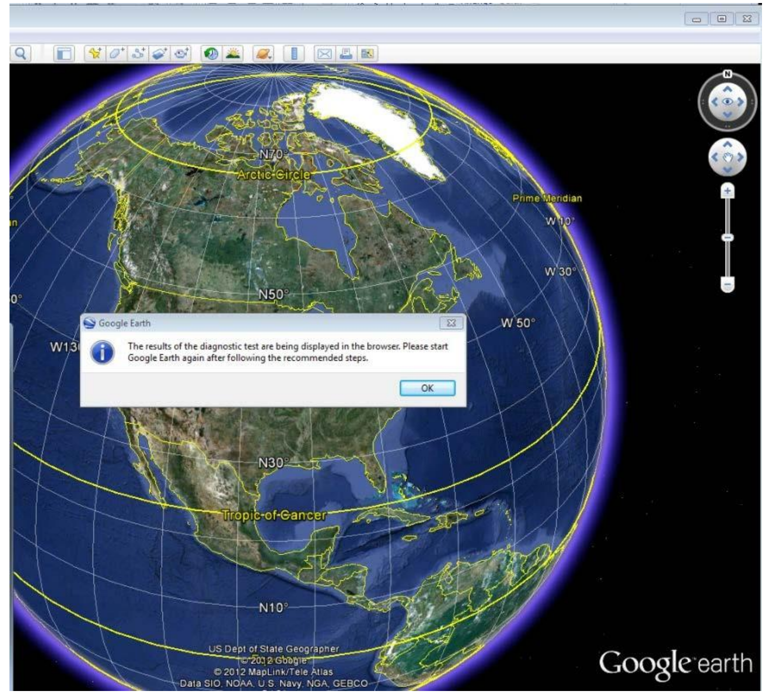


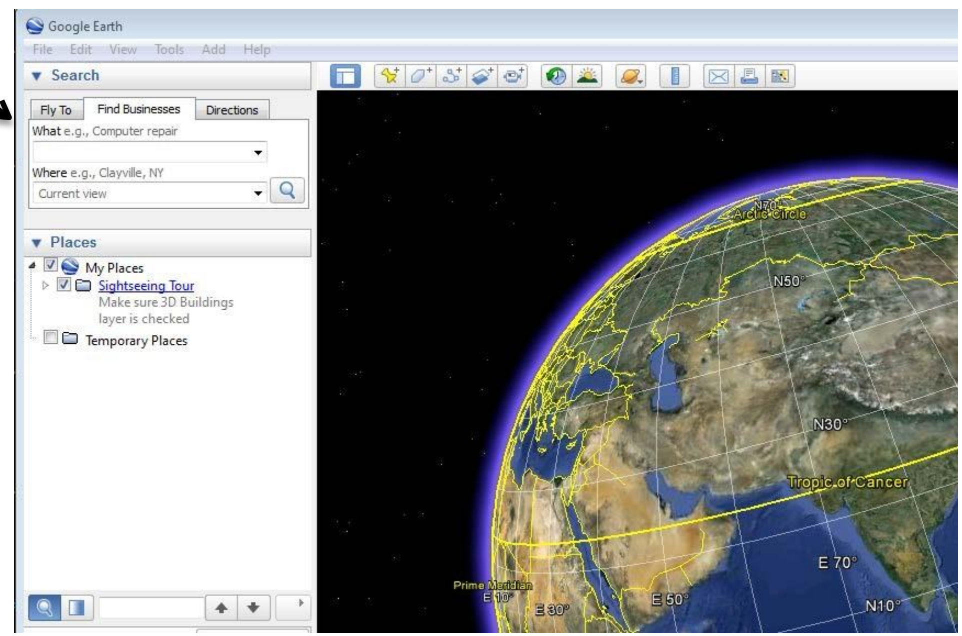
5. Error Prevention

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.





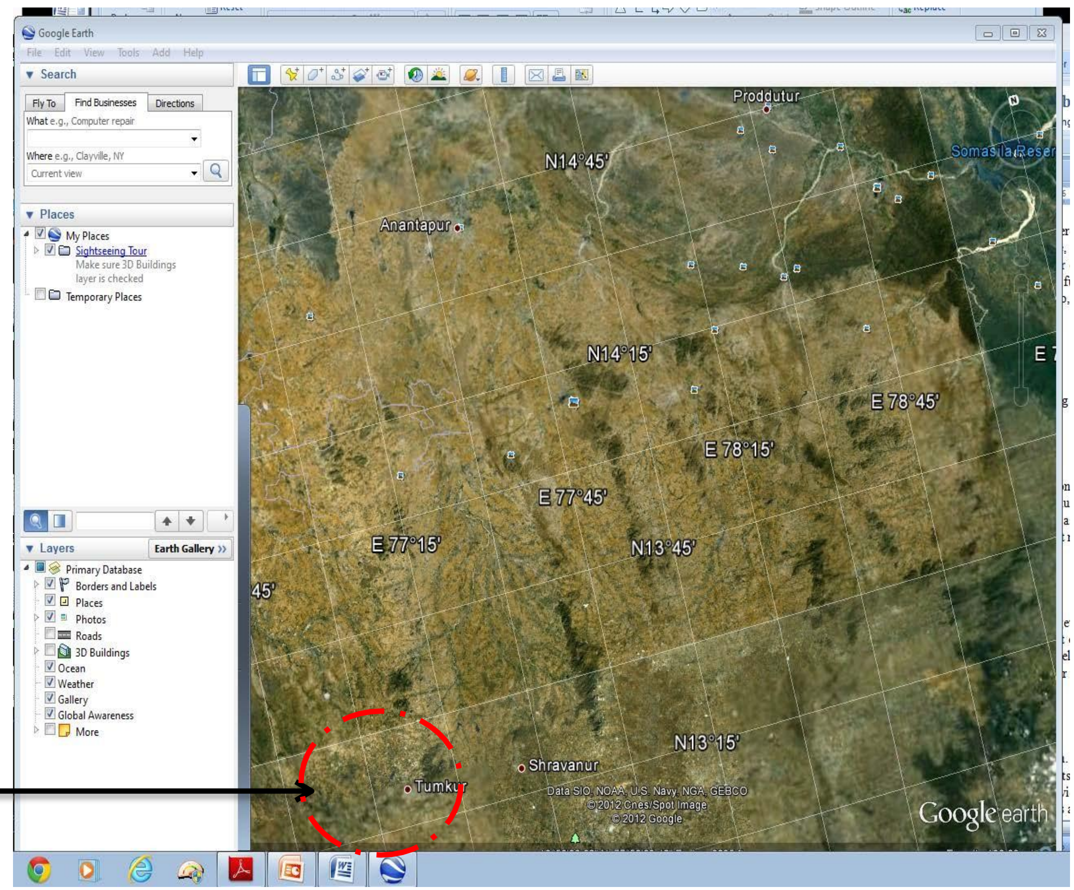
6. Recognition Rather than Recall

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 trail 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.

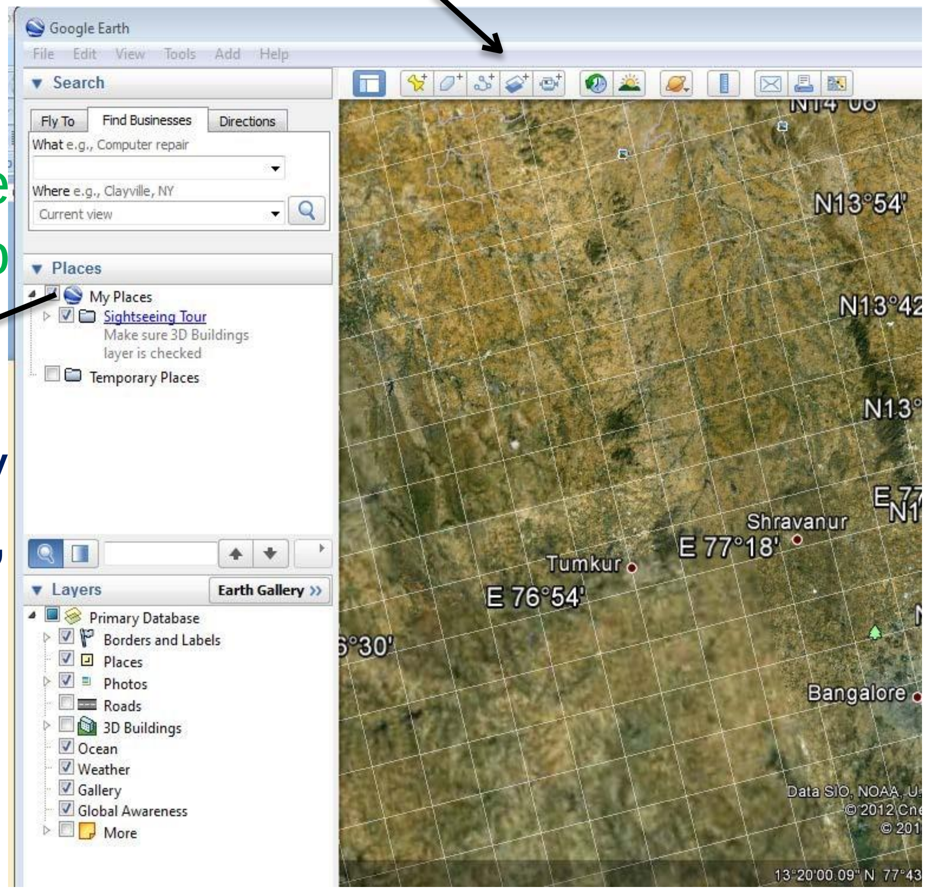
7. Flexibility and Efficiency of Use

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.

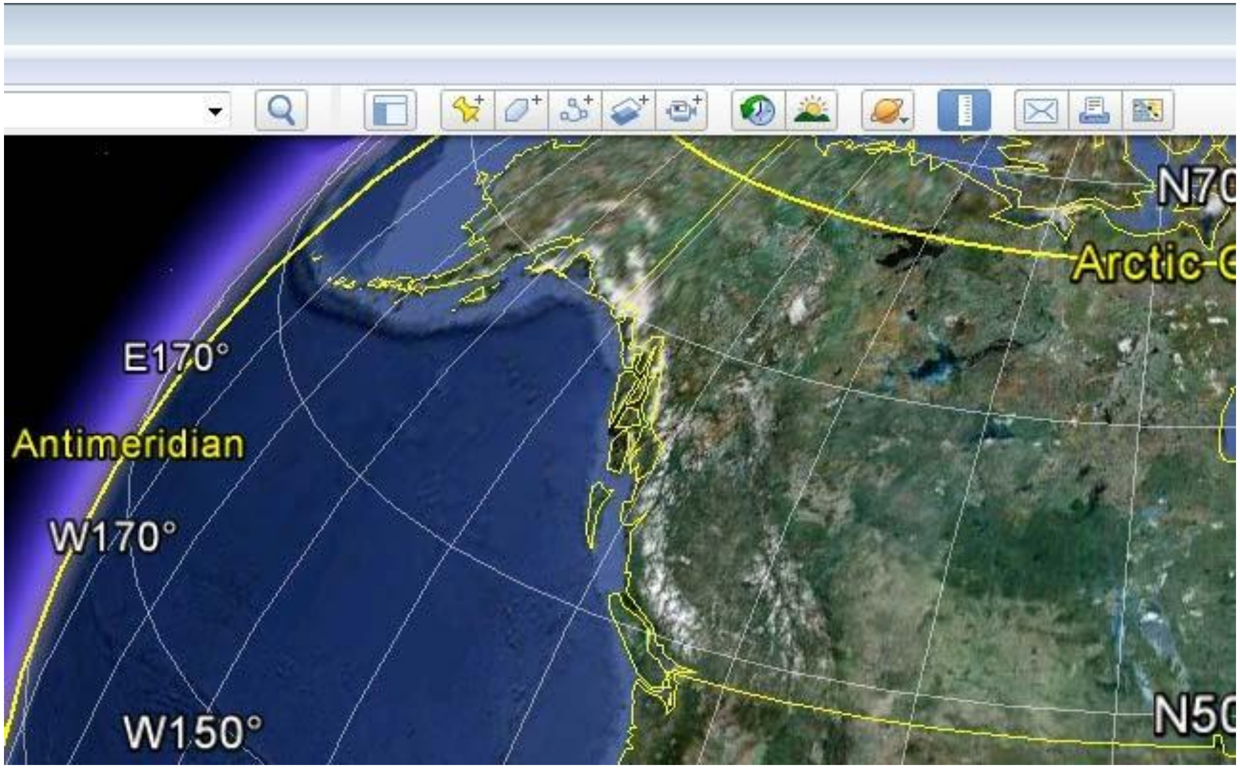


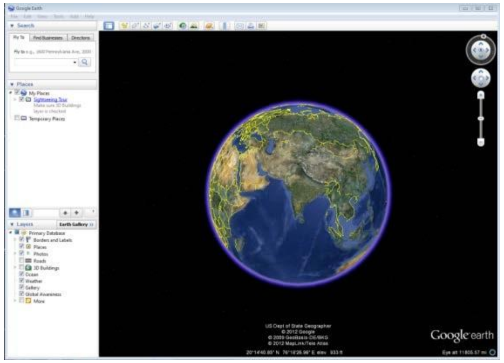
8. Aesthetic and Minimalist Design

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.

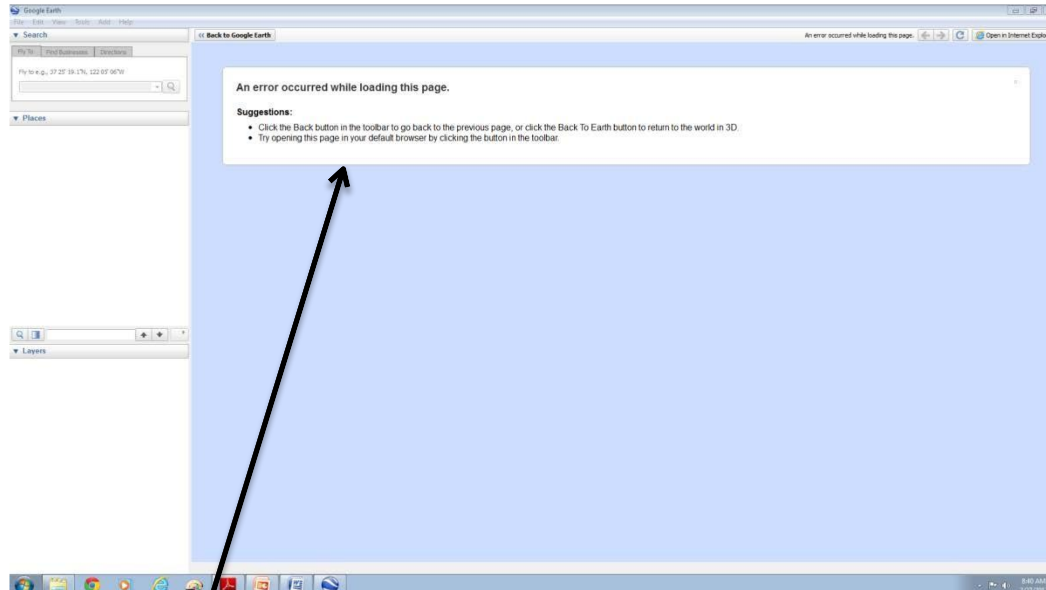
The whole graphics is tuned to make the Map window important (which is the main function of the software)





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.

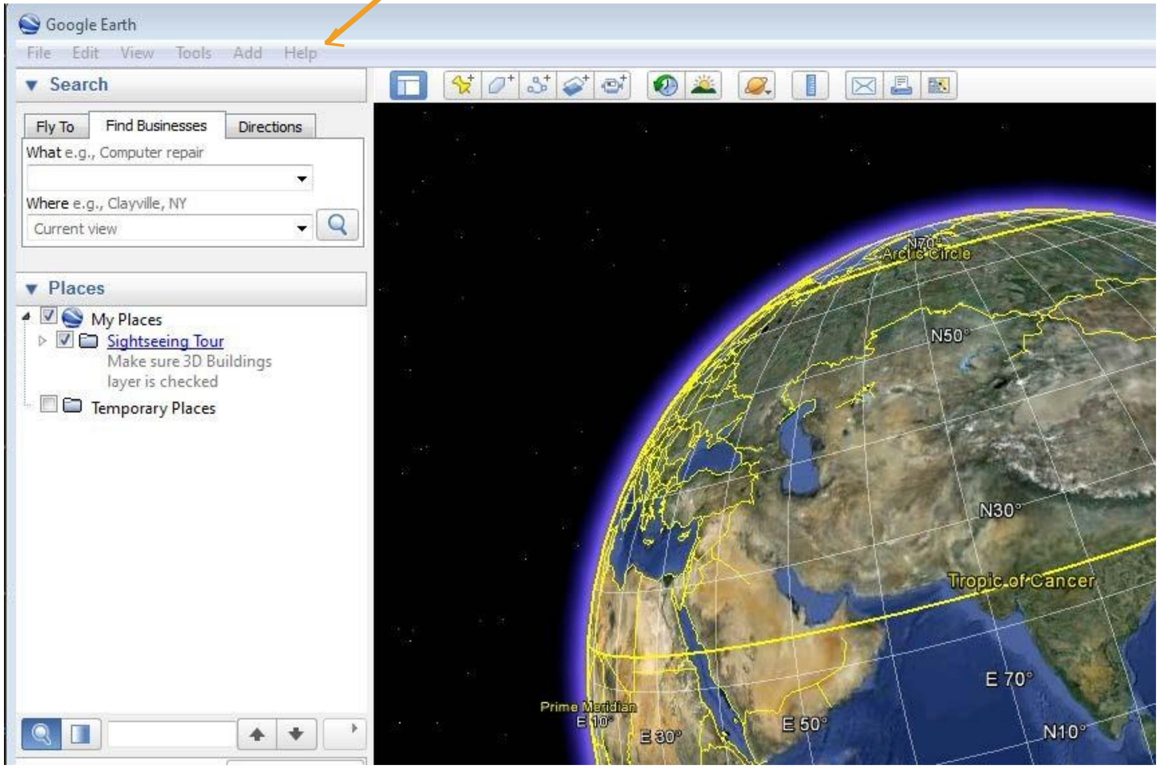


The software is user friendly.
It does not penalise the user for not knowing how to use the software.

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
1. Visibility of System Status	System status if the Network connection is lost or it is absent	Severity: Medium			
2.Match between System and the Real World	Good. No intervention required	NA			
3. User Control and the Freedom	For a novice user disappearing zoom slider bar can be confusing!	Severity: Low			
4. Consistency and the Standards	Good. No intervention required				
5.Error Prevention	Good. No intervention required				
6. Recognition Rather than Recall	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.), *Useability Inspection Methods*, John Wiley & Sons, New York, NY.

•The Ten Heuristics and description have been adapted from the authors publication.

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. 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 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.

Marketing based 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.

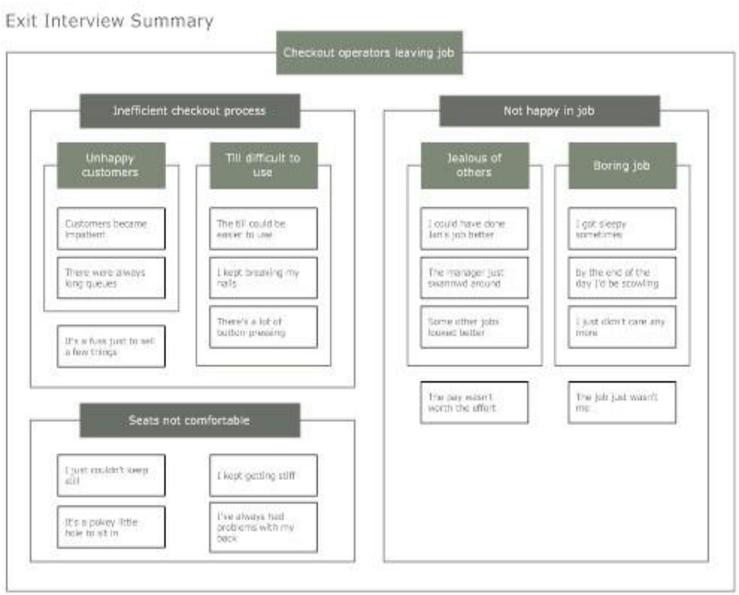
Prepare documenting mediums such as cameras, voice recorders etc.

Tools / Instruments used in Contextual Inquiry

- Open ended questioning based on observations
- Pre-prepared Questionnaire (User Survey)
- Ethnographic observation dairy with notes

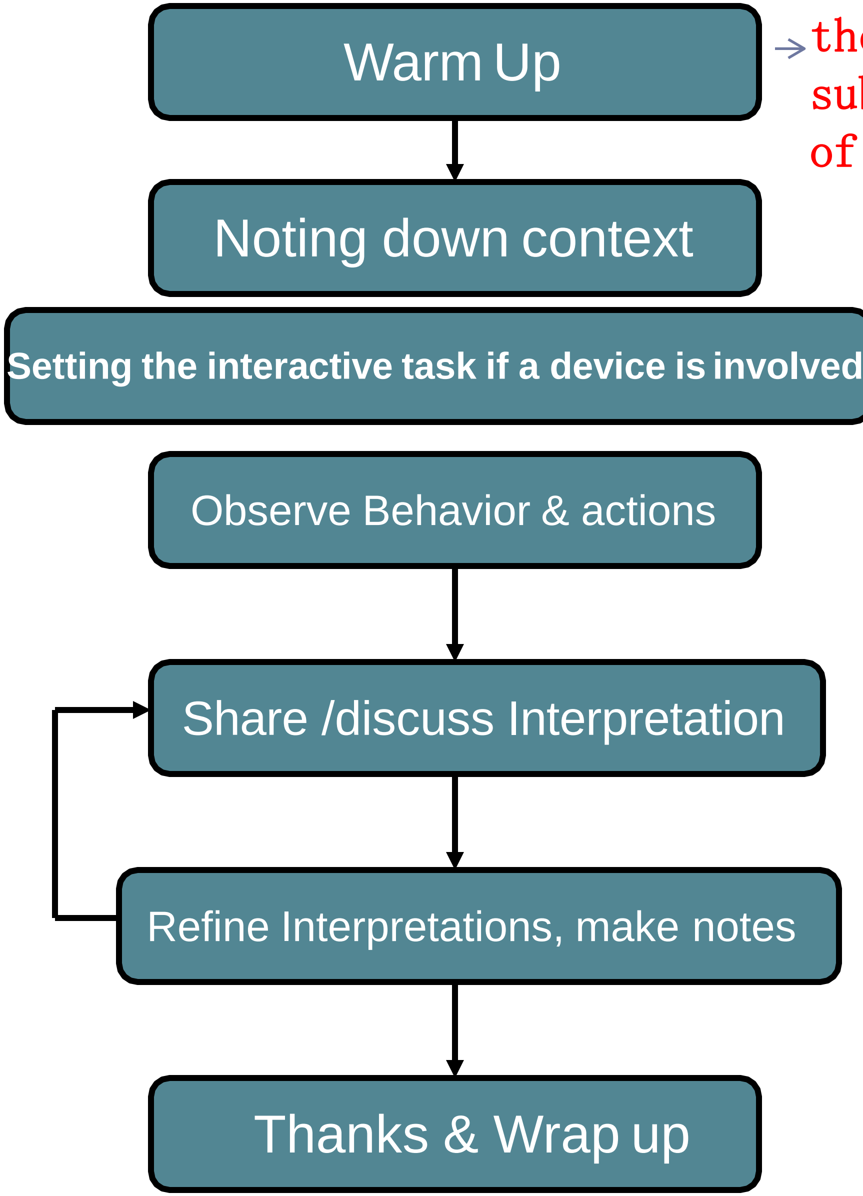
(These notes are converted into Affinity diagrams)

- Focus group interviews
- Structured discussions
- Photo / video documentation
- Hierarchy diagrams
- Story boards
- Mind maps



- Affinity diagrams: Data that have 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 'groups'. Groups are given labels.

Stages of a Contextual Interview



Greetings. Explaining the purpose. Rights of subjects for protection of personal information.

Prejudging the user. Gathering the user's usage data and the experience data.

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

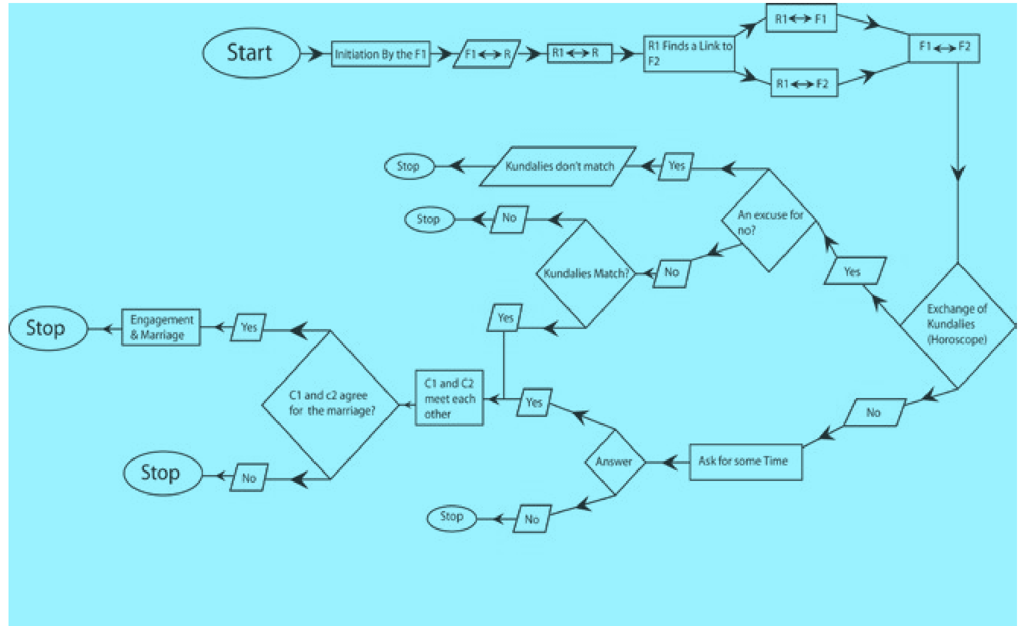
Descriptions of these models are given in the following slides.

Descriptions of Models

- **Flow Model**

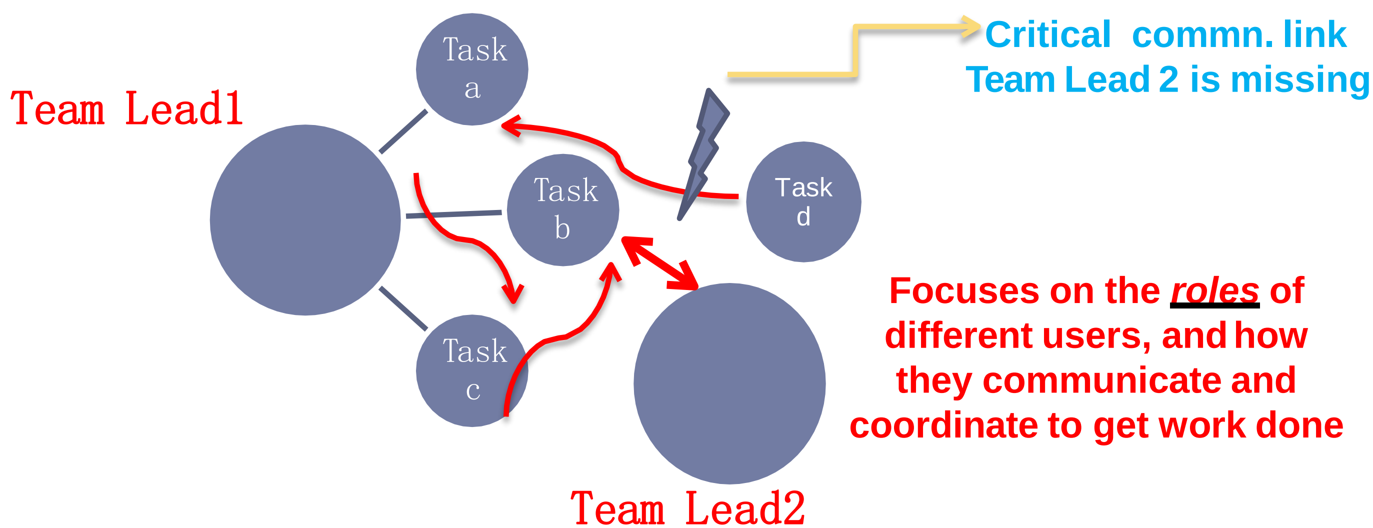
Flow model represents the coordination, communication, interaction, roles, and the responsibilities of the people in a certain work practice.

It is based on the logic of flow of information between different entities making up the system within the context.



Flow chart for the Arranged Marriage Process in Rajasthan

This model is mainly used for depicting the logic behind the flow of information



Focuses on the *roles* of different users, and how they communicate and coordinate to get work done

Sequence Model – It represents the steps users go through - to accomplish an activity.
Sequence models are 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

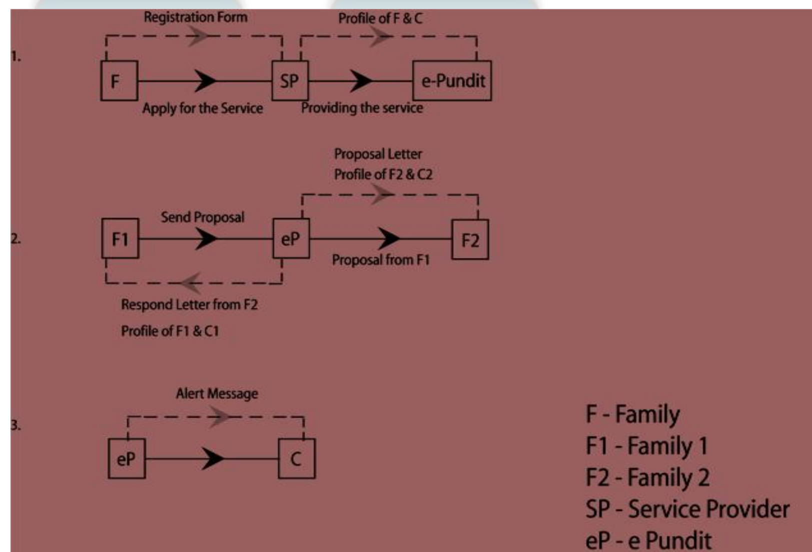
User error → HELP

Select a plane → Select the sketch tool → Under sketch tool select line tool → Draw the rectangle shape → Select Extrude tool

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.

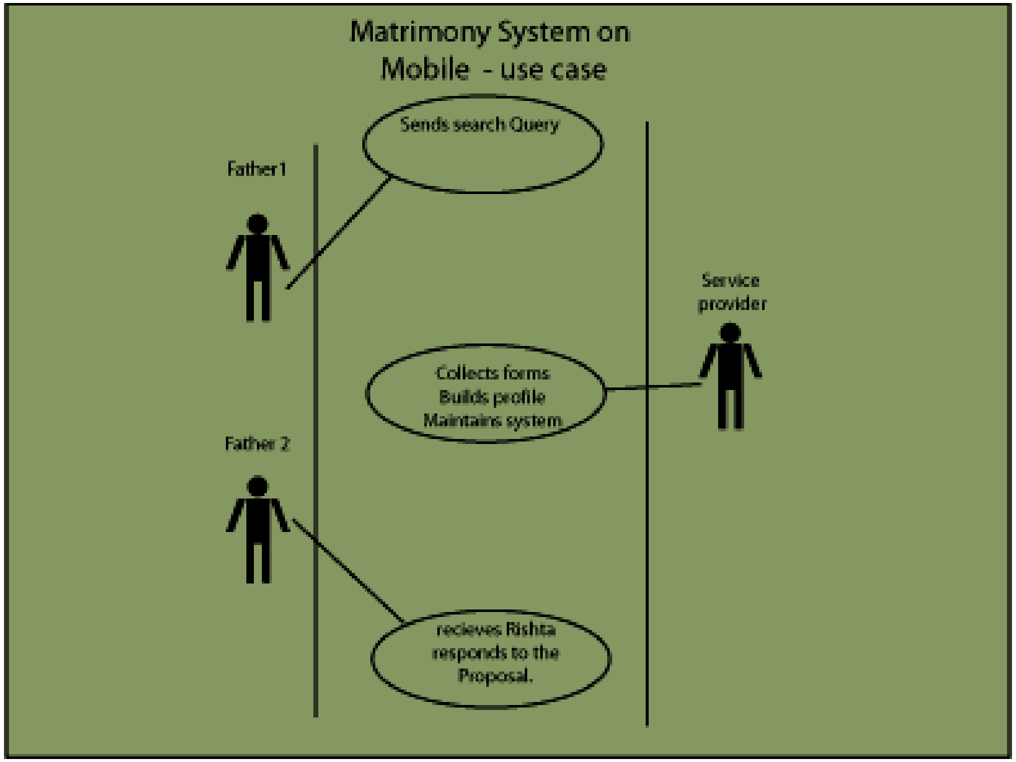
Document flow in an Indian Matrimonial pre-exchange is shown on the right.



Document Flow Diagrams

Cultural Model -

It represents the norms, influences, and the practices that are present in the work environment and which are specific to a particular region or are traditionally followed as local norms.



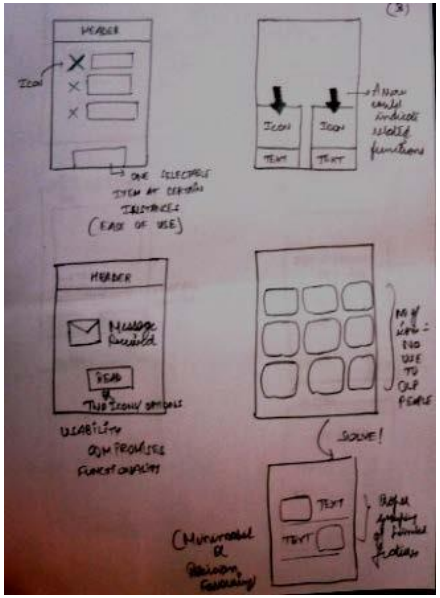
Often culture specific comments or differences are mentioned using either flow diagrams or sequence diagrams or both.

In Indian culture parents are the ones who establish the first contact and collects the information of a prospective son-in-law /daughter-in-law.

Language for example is a Culture model variable.

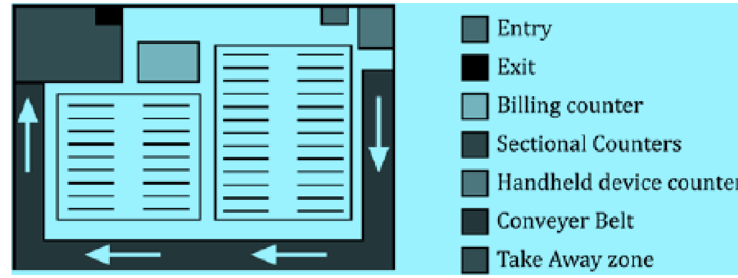
This flow model is not evident for a person from another culture.

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 are used to support the work. Example would be a Paper based voucher simultaneously filled up in a particular step of a sequential task flow.



Interviewers should inquire into the *<u>structure</u>*, *<u>content</u>*, *<u>presentation</u>* and the *<u>usage</u>* of the artifact.

Physical Model – It represents the physical lay out of the environment where the work tasks are accomplished.



Simple examples would be 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 is represented as a map. Example would be of a retail shopping.

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. Extracts from the flow models represent abstracts of communications, responsibilities and 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 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 the software **code** walk through, the sequence represents a segment of the program code that is 'walked through' by the reviewers so that they can check certain characteristics (e.g., 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", (4th 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.

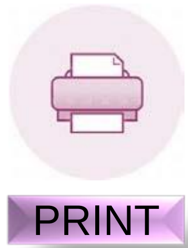
A walk through reveled both the above assumptions are out of sequence as far as the user is concerned.

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

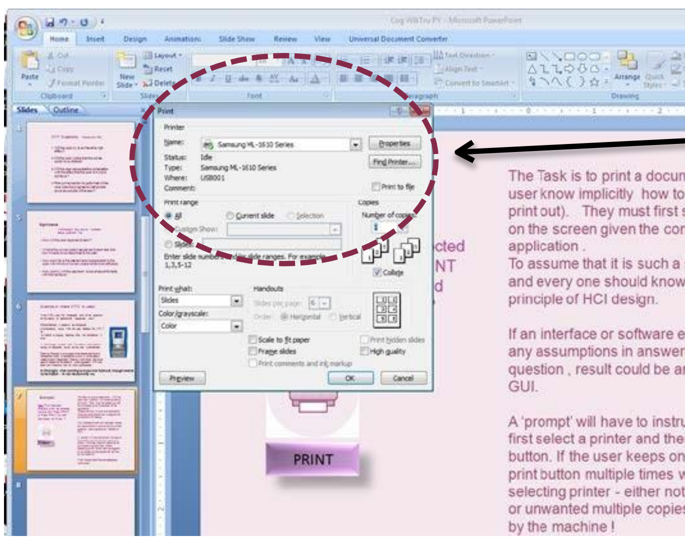
Task: Print the Document. Should a printer be selected first and then Press PRINT or Press PRINT first and then Select the Printer ?



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.
(Catagorise 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.

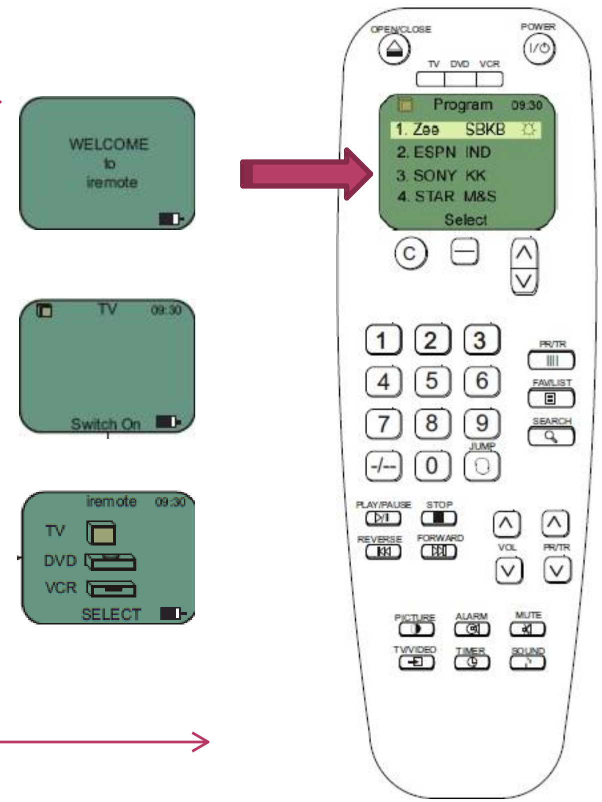
4. Conduct the Walk Through Session

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. Yes – PARTLY - No	Did the user confidently know that the choice being made by him/her is the right one? Yes No	Did the user understand the feedback after every action	Did the user Complete the Task With satisfaction Yes PARTLY No	Comments / Alternative suggestions / solutions / discussion points.
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:

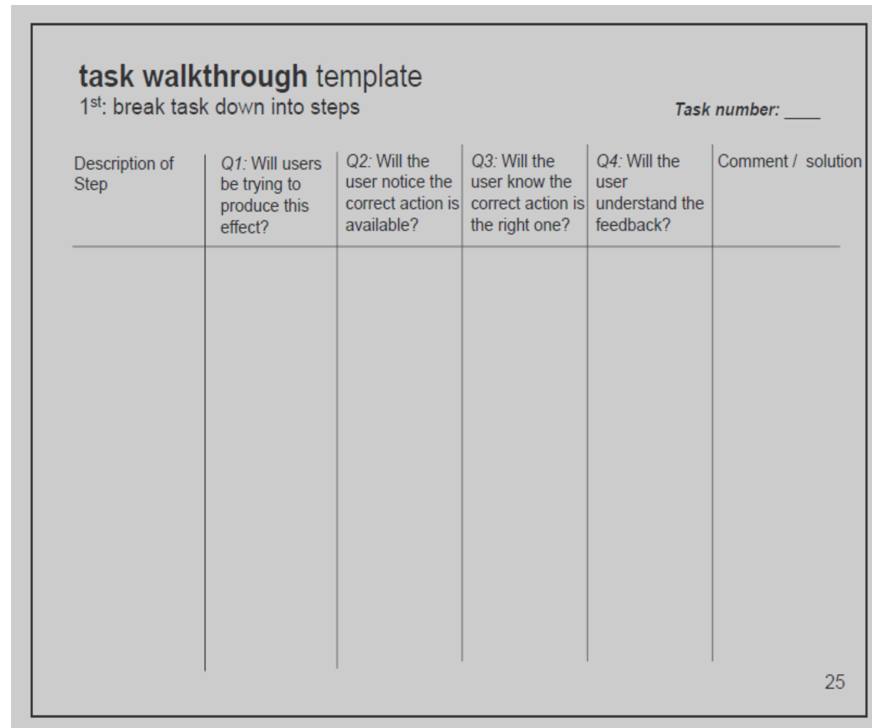
What's the difference between a Heuristic Evaluation and a Cognitive Walk Through?

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 bellow 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
1st: 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

25

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 (4th Edition) Englewood cliffs, NJ Yourdon Press.

HCI: Empirical Research Methods

Learning Objective

- In the previous lecture module, we have already learnt several evaluation methods such as heuristic evaluation, cognitive walkthroughs or cognitive models to evaluate the interactive system designs at the early phases
- As we have already mentioned, interactive system design is not complete unless it is evaluated with real users at the end

Learning Objective

- In this lecture, we shall discuss user evaluation methods
- In particular, we shall discuss the following:
  - The Key Concerns in User Evaluation
  - Data Collection Procedure
  - Data Analysis Techniques

Empirical Research

- Empirical research is broadly defined as the "observation-based investigation" seeking to discover and interpret facts, theories, or laws
- Collection and Analysis of the end user data for determining the usability of an interactive system is an “observation-based investigation”, hence it is qualified as an empirical research study

Themes of Empirical Research

- Generally, Empirical Research is based on Three Themes
  - Answer and Raise Questions (Testable Research Questions) about a new or existing UI Design or Interaction Method
  - Observe and Measure (Ratio Measurement Scale is preferable)
  - User Studies

Research Question

- It is very important in an empirical research to formulate “appropriate and relevant” research questions
- For example, consider some questions about a system
  - Is it viable?
  - Is it as good as or better than current practice?
  - Which of several design alternatives is best?
  - What are its performance limits and capabilities?
  - What are its strengths and weaknesses?
  - How much practice is required to become proficient?

Testable Research Question

- Preceding questions, while unquestionably relevant, but are not *testable research questions*
- We have to come-up with testable research questions
- Let's illustrate the idea with the following example:

Suppose we have designed a new text entry technique for mobile phones. We think that the design is good. In fact, we feel that our method is better than the most widely used current technique, multi-tap. We decide to undertake some empirical research to evaluate our invention and to compare it with multi-tap? What are our testable research questions?

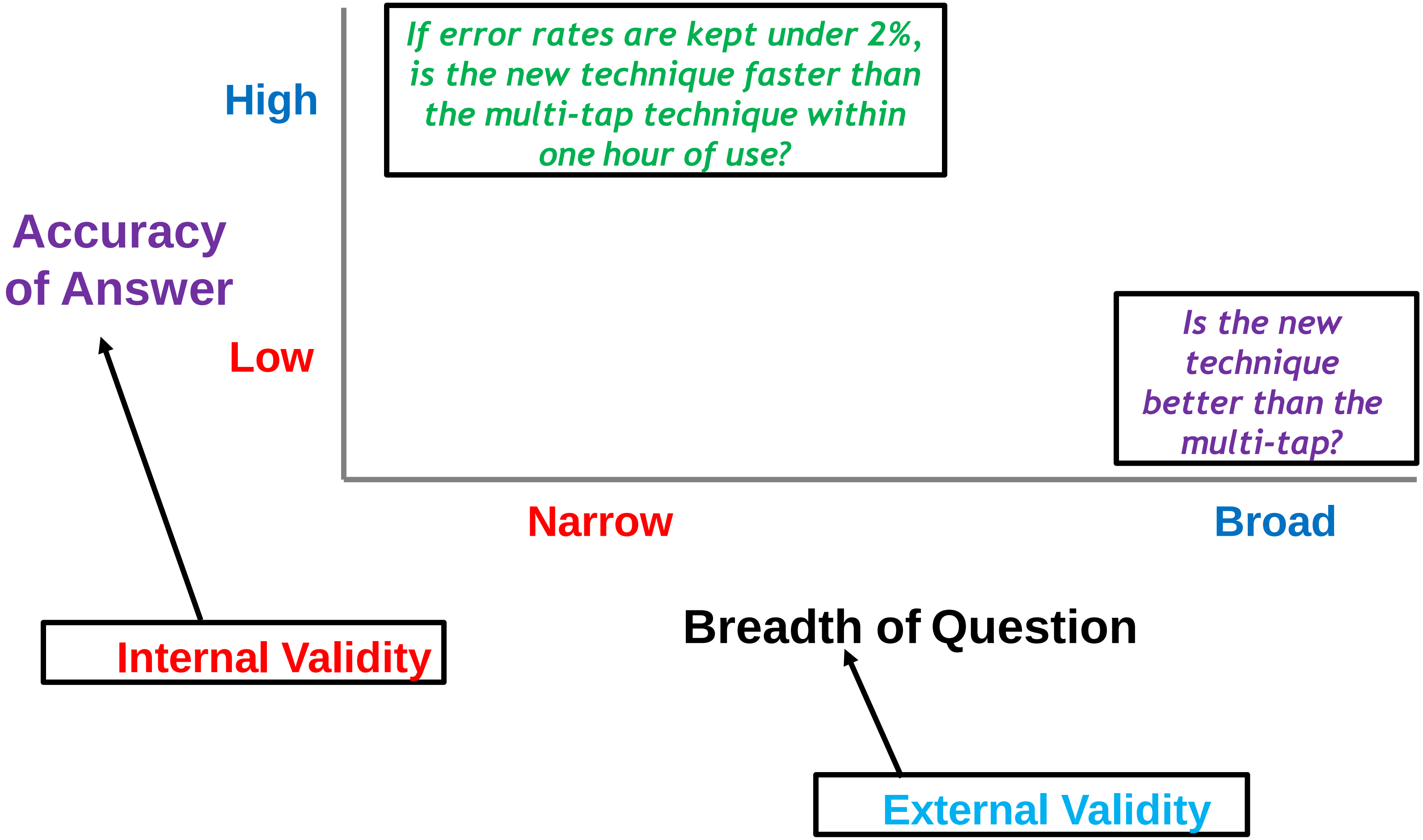
Testable Research Question

- Weak Question
  - Is the new technique better than multi-tap?
- Better
  - Is the new technique faster than multi-tap?
- Better still
  - Is the new technique faster than multi-tap within 1 hour of use?
- Even better
  - If error rates are kept under 2%, is the new technique faster than multi-tap technique within one hour of use?

Testable Research Question

- The questions are testable (we can actually conduct the experiments to test the answer to the testable questions)
- We can ask very specific questions (the last one) or relatively broader questions (the first one)
- For very specific questions, the accuracy of answers is high whereas for the broader questions, the breadth or the generalizability is very high

Testable Research Question



Internal and External Validity

- The extent to which the effects observed are due to the test conditions is called internal validity of the research question
- The extent to which the results are generalizable to other people and other situations is known as the external validity of the research question

More Examples on Validity

- Suppose we wish to compare two input devices for remote pointing (e.g., at a projection screen)
- External validity is improved if the test environment mimics expected usage
  - The test environment should use a projection screen, position participants at a significant distance from screen, have participants stand and include an audience

More Examples on Validity

- Note that creating the test environment mimicking the real usage scenario is not easy
- Instead we can go for controlled experiments where we can ask the user to sit in front of a computer in a laboratory and use the pointing devices to operate an application on the screen
  - The above setting can answer research questions with high internal validity but can not help in determining if the answers are applicable in real world scenario

More Examples on Validity

- Consider the scenario where we wish to compare two text entry techniques for the mobile devices
- To improve external validity, the test procedure should require participants to enter representative samples of text (e.g., phrases containing letters, numbers, punctuation, etc.) and correct mistakes
  - This may require compromising on internal validity

Trade-off

- There is tension between internal and external validities
  - The more the test environment and experimental procedures are "relaxed" (to mimic the real-world situations), the more the experiment is susceptible to uncontrolled sources of variation, such as pondering, distractions, or secondary tasks

Resolving the Trade-off

- Internal and external validities are increased by posing multiple narrow (testable) questions that cover the range of outcomes influencing the broader (un-testable) questions

Ex: A technique that is ***faster***, is ***more accurate***, takes ***fewer steps***, is ***easy to learn***, and is ***easy to remember***, is generally ***better***

Resolving the Trade-off

- The "good news" is that there is usually a positive correlation between the testable and un-testable questions
  - For example, participants generally find a UI ***better*** if it is ***faster***, ***more accurate***, ***takes fewer steps***, etc.
- The "good news", in fact, is not so good after all as it raises more confusions

Implication

- The “good news” actually implies we do not need empirical research!!
- We just do a user study and ask participants which technique they preferred
  - Because of the “positive correlation”, we need not take the pain in collecting and analyzing the data

Implication

- However, this is not true
- If participants are asked which technique they prefer (a broader question), they'll probably give an answer…even if they really have no particular preference!
    - There are many reasons, such as how recently they were tested on a technique, personal interaction with the experimenter, etc.

Implication

- Therefore, such preferences need not be indicative of the system performance
  – We need to scientifically ascertain the validity of the preferences expressed by the participants, which requires formulation of testable research questions
- Also, with broader questions, we may not get an idea about the feasibility or usefulness of the system
  – It is not enough to know if a system is better than another system only but we also need to know "how much better" (for example, it may not be feasible economically to develop a system that is only 5% better than the current system)

Implication

- Seeking the feedback from users on broader questions is not very helpful from another perspective
  - It does not help to identify the strengths, weaknesses, limits, capabilities of the design, thereby making it difficult to identify opportunities for improvements
- Such concerns can be addressed only with the raising of testable research questions
- A point to note is, to test the validity of empirical research questions through observations, we need **measurements**
  - This brings us towards the second theme of empirical research, i.e. to Observe and Measure.

Observe and Measure

- In empirical research, the observation is the most fundamental thing (activity) to do

- Observational (empirical) data can be gathered in two ways

  – Manual: In this case, a human observer manually records all the relevant observational data

  – Automatic: The observation can be recorded automatically, through the use of computers, software, sensor, camera etc.

Observe and Measure

- A measurement is, simply put, a recorded observation
- There are broadly four ***Scales of Measurements*** that are used (Nominal, Ordinal, Interval and Ratio)
- ***Nominal*: ** Here, we assign some (arbitrary) codes to the attributes of the observational data (for example, male = 1, female = 2 etc.)

Scales of Measurements

- ***Ordinal*****:** In this scale of measurement, the observations are ranked (for e.g., 1st, 2nd, 3rd etc.)

- ***Interval*****:** In an interval measurement, we consider equally spaced units but no absolute starting point (for e.g., 20° C, 30° C, 40° C, …)

- ***Ratio*****:** This scale of measurement has an absolute starting point (zero) and uses the ratios of two quantities (for e.g., 20 WPM, 30 CPS etc.)

Scales of Measurement

- Nominal
- Ordinal
- Interval
- **Ratio**

Crude

Sophisticated

Ratio measurements, being the most sophisticated scale of measurement, should be used as much as possible

Ratio Measurements

- As mentioned in the previous slide, ratio scales are the most preferred scale of measurement
  - This is because ratio scales make it convenient to compare or summarize observations
- If we are conducting an empirical research, then we should strive to report the "counts" as ratios wherever possible

Ratio Measurements

- For e.g., let us assume that we observed that " a 10-word phrase was entered by a participant in an empirical study in 30 seconds". What should we measure?
  - If we measure the "time to enter text" (e.g., t = 30 seconds) as an indicator of system performance, it is a bad measurement.
  - However, if we go for a ratio measurement (Entry Rate = 10/0.5 i.e. Entry Rate = 20 wpm), that is much better and gives a general indication of the performance.

Ratio Measurements

- Let us consider another example. Suppose in an empirical study, we observed that a participant has committed two errors while entering a 50 character phrase.
  - If we measure the "number of errors committed" (i.e., $n = 2$) as an indicator of system performance, it is a bad measurement.
  - However, if we go for a ratio measurement (Error Rate = 2/50, i.e. Error Rate = 0.04 = 4%), that is much better and is a more general performance indicator.

Summary

- We have discussed two of the three themes of empirical research, namely: (1) Answer and Raise (Ask) Questions about a new or existing UI Design or Interaction Method, (2) Observe and Measure
- We shall continue our discussion w.r.t. the third theme of empirical research (i.e. User Studies) in the next lecture

HCI: Empirical Research Methods

Learning Objective

- In previous lectures, we discussed the conceptual framework of empirical research (which is observation-based investigation)
  - In the context of HCI, it refers to an investigation about the usability of a system with the help of end users in a real-life scenario
- The investigation is based on testable research questions
  - We saw how to formulate the appropriate/relevant questions (the validity of questions, the trade-off and how to resolve the trade-off)
- We also discussed the process of observation and measurements
- In this lecture, we shall discuss other aspects of empirical research; In particular, we shall discuss the design of experiments.

Themes of Empirical Research

- To recap, the three themes of empirical research are
  - Answer and Raise Questions (Testable Research Questions)
  - Observe and Measure (Ratio Scale of Measurement is Preferable)
  - User Studies (Design of Experiments)
- In the previous lecture, we already discussed the first two HCI themes, namely: (1) Answer and Raise Questions, (2) Observe and Measure.
- In this lecture, let’s discuss the third theme (User Studies) which is also equally important since we need to perform the observations in such user studies (design of Experiments).

User Study

- A user study, in the context of HCI, is a scientific way of collecting and analyzing observational data from the end users on an interactive system
- Collection of data involve the real users for conducting experiments and design of experiments

Experiment Design

- Experiment design is a general term refers to the organization of variables, procedures, etc., in an experiment
- The process of designing an experiment is the process of deciding on which variables to use, what procedure to use, how many participants (users) to use, how to solicit them etc.

Terminology

- Terms to know
  - Participant (User)
  - Independent Variable (Test Conditions)
  - Dependent Variable
  - Control Variable
  - Random Variable
  - Confounding Variable
  - Within Subjects vs. Between Subjects
  - Counterbalancing and Latin Square

Participant (User)

- The people participating in an experiment are referred to as participants (users)
  - When referring specifically to the experiment, use the term participants (e.g., “all participants exhibited a high error rate…”)
  - General comments on the problem or conclusions drawn from the results may use other terms (e.g., “these results suggest that users are less likely to…”

Independent Variable

- An independent variable is a variable that is selected or controlled through the design of the experiment
  - Examples including the device, feedback mode, the button layout, visual layout, gender, age, expertise, etc.
- The terms: independent variable and the factor are synonymous

Test Conditions

- The levels, values, or settings for an independent variable are the test conditions

- These conditions provide names for both an independent variable (factor) and the test conditions (levels) for the controlled variable (ex: table details)

Factor	**Levels (Test Conditions)**
Device	Mouse, Trackball, Joystick
Feedback Mode	Audio, Tactile, Visual
Task	Pointing, Dragging
Visualization	2D, 3D, Animated

Dependent Variable

- A variable representing the measurements or observations on a independent variable
- It is required to provide a name for both the dependent variable and its unit
  - Examples: Task completion time (ms), speed (WPM (word per minute)), selections per minute, etc.), error rate (%), throughput (bits/s (bps))

Control Variable

- Circumstances or factors that might influence a dependent variable, but are not under investigation need to be accommodated in some manner

- One way is to control them or to treat them as the control variables (e.g., room lighting, background noise, temperature)

Random Variable

- Instead of controlling all circumstances or factors, some might be allowed to vary randomly

- Such circumstances are random variables

Confounding Variable

- Any variable that varies systematically with an independent variable is a confounding variable
  - For example, if three devices are always administered in the same order, participant performance might improve due to practice; i.e., from the 1^{st} to the 2^{nd} to the 3^{rd} condition; thus "practice" is a confounding variable (because it varies systematically with "device")

Within Subjects, Between Subjects

- The administering of levels of a factor is either within subjects or between subjects
  - If each participant is tested on each level, then the factor is said to be within subjects
  - If each participant is tested on only one level, then the factor is said to be between subjects. In this case a separate group of participants is used for each level
- The terms *repeated measures* and within subjects are synonymous
- A relevant question is, which of the two approaches (within subject and between subject) should be chosen in designing an experiment

Within Subjects, Between Subjects

- Answer: It depends!
  - Sometimes a factor must be between subjects (e.g., gender, age)
  - Sometimes a factor must be within subjects (e.g., session, block)
  - Sometimes there is a choice. In this case there is a trade-off
- The advantage of within subject design is, the variance due to participants' pre-dispositions should be the same across test conditions
- Between subjects design, on the other hand, has the advantage of avoiding interference effects (e.g., the practice effect while typing on two different layouts of keyboards)

Counterbalancing

- For repeated measures designs, participants' performance may tend to improve with practice as they progress from one level to the next level
  - Thus, users may perform better on the second level simply because they benefited from the practice on the first (this is undesirable)
- To compensate this, the order of presenting conditions must be counterbalanced

Latin Square

- Participants are divided into groups, and a different order of administration is used for each group
- The order is best governed by a Latin Square (the defining characteristic of a Latin Square is that each condition occurs only once in each row and column)

Latin Square

- Example: Suppose that we want to administer 4 levels (denoted by A, B, C and D) of a factor to the 4 participants (represented by P1, P2, P3 and P4)
  - We can construct a 4×4 Latin square arrangement to depict the order of administering the levels to each participant

P1	A	B	C	D
P2	B	C	D	A
P3	C	D	A	B
P4	D	A	B	C

Latin Square

- In a *balanced* Latin Square, each condition both precedes and follows each other condition an equal number of times
  - We can construct a balanced 4×4 Latin square arrangement for the previous example

P1	A	B	C	D
P2	B	D	A	C
P3	D	C	B	A
P4	C	A	D	B

Expressing Experiment Design

- Consider the statement “3 x 2 repeated-measures design”
  - It refers to an experiment with two factors, having three levels on the first, and two levels on the second. There are six test conditions in total. Both factors are repeated measures, meaning all participants were tested on all test conditions
- Any type of experiment is expressed similarly

Summary

- In this and the previous lectures, we discussed the fundamental ideas associated with empirical research (namely: question formulation, observation and measurement and experiment design)
- In the next lecture, we shall see another important aspect of empirical research, namely: the case for statistical analysis of empirical data

HCI: Analysis of Empirical Data

Learning Objective

- In the previous lectures, we have already discussed the basics of HCI - Empirical Research Methods
  - We discussed the three important themes, namely: Research Question Formulation, Observation and Measurement, and User Study (Experiment Design)
- In the present lecture, we shall mainly focus on the Analysis of Empirical Data

Learning Objective

- In particular, we shall learn the following:
  - The case for Statistical Analysis of Observed Data
  - Discussion on one of the commonly and widely used Statistical-based Empirical Analysis Techniques, namely: One-way ANalysis Of VAriance (ANOVA) Test

Answering Empirical Questions

- Suppose, we want to determine if the text entry speed of a proposed text input system is more than an existing system
- We know how to design an experiment and further we know how to observe and measure
- So, let us do the following:

- We conduct a User Study and measure the performance on each of the given test conditions (our proposed system and the existing system) over a group of participants

- For each test condition, let us compute the mean score (text entry speed) over the group of participants (users)

Answering Empirical Questions

- We now have the observed (empirical) data. What next?
- Now, we are faced with the following three questions:
  - Is there any difference?

    This is true as we are most likely to see some differences. However, can we conclude anything from this difference? This brings us to the second question.

  - Is the difference too large or too small?

    This is more difficult to answer. If we observe a difference of, say, 30%, we can definitely say this difference is too large. But, we can't say anything definite about, i.e., 5% difference. Clearly, the difference itself can't help us to draw any definite conclusion. This brings us to the third question.

Answering Empirical Questions

- We are faced with the following three questions (contd…):
  - Is the difference significant or is it due to chance?

    Even if the observed difference is "small", it can still lead us to conclude about our design if we can find the nature of difference. If the difference is found to be "significant" (not occurred due to any chance), then we can say something about our design.
- Pl. note that the term "significance" is a statistical term
- The test of (statistical) significance is an important aspect of empirical data analysis
- We can use statistical techniques for this purpose
  - The basic technique is ANOVA or ANalysis Of VAriance

Statistical Hypothesis Testing

In statistics, the Hypothesis is a claim or the statement about a property of a population.

Hypothesis Test (or Test of Significance) is considered as a standard procedure for testing a claim about a property of a population.

Basics of Hypothesis Testing

The following components of a statistical hypothesis test are widely used for carrying out the comprehensive procedures.

- **Null and Alternative Hypotheses**
- **Test Statistics**
- **Critical Region and Critical Values**
- **Significance Levels**
- ***P*-values**
- **Decision Criteria**
- **Type I and II Errors**
- **Power of a Hypothesis Test**

Learning Objectives

- **Given a claim, How to identify both the null and the alternative hypotheses, and then express them in symbolic form.**
- **Given a claim and sample data, how to calculate the value of the test statistic.**
- **Given a significance level, How to identify the critical value(s).**
- **Given a value of the test statistic, How to identify the *P*-value.**
- **State the conclusion of a hypothesis test in simple, non-technical terms.**

Example: Let's refer to the Gender Choice product that was once distributed by the ProCare Industries. ProCare claimed that couples using pink packages of Gender Choice would have girls at a rate greater than 50% or 0.5. Let's again consider an experiment whereby 100 couples use the Gender Choice in an attempt to have a baby girl; Let's assume that the 100 babies include exactly 52 girls, and let's formalize some of the analysis. Under normal circumstances the proportion of girls is 0.5, so a claim that Gender Choice is effective, can be expressed as $p > 0.5$. Using a normal distribution as an approximation to the binomial distribution, we find that P(52 girls or more in 100 births) = 0.3821. Figure 1 shows that with a probability of 0.5, the outcome of 52 girls in 100 births is not unusual.

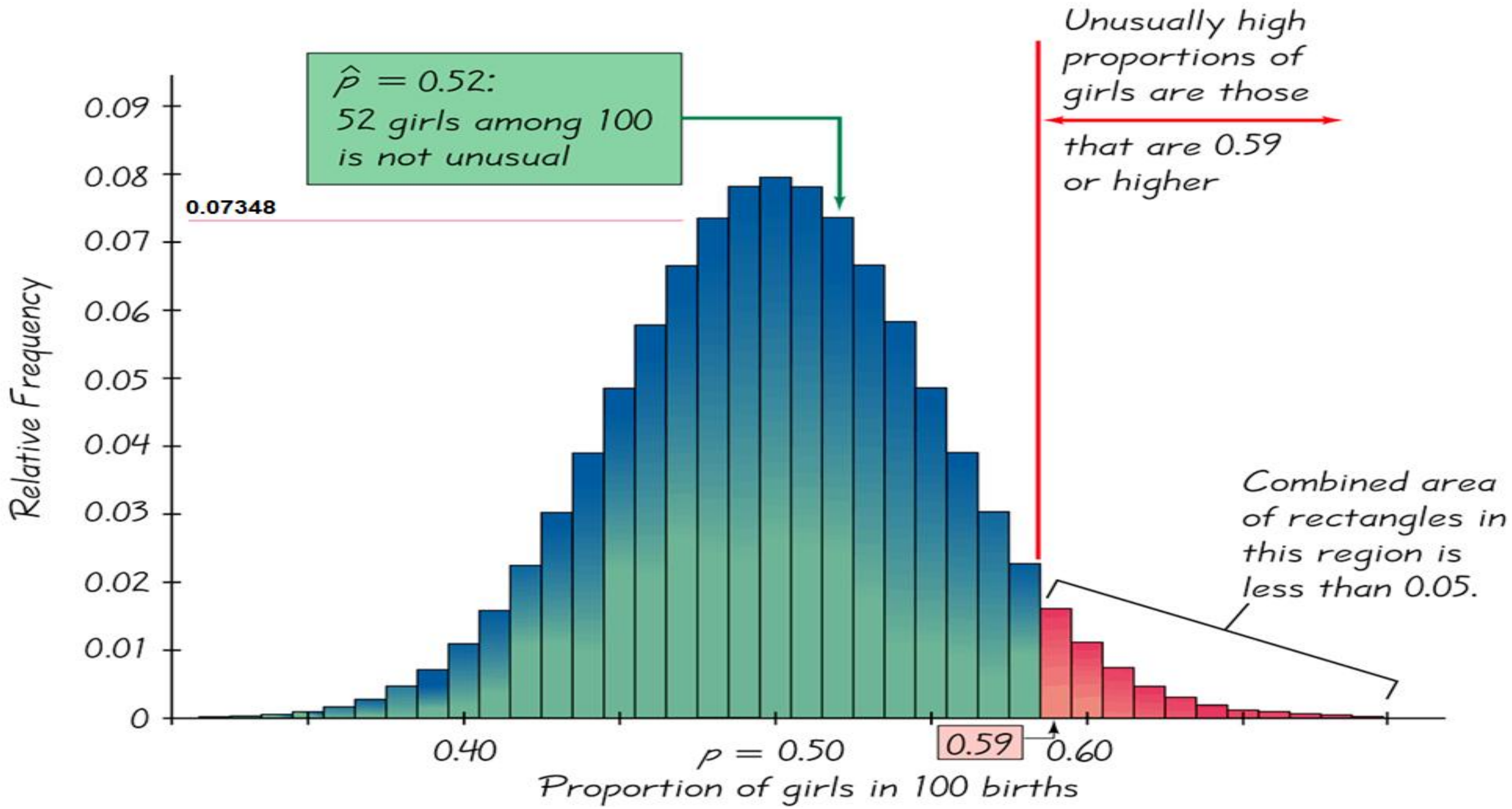


Figure 1

We do not reject random chance as a reasonable explanation. We conclude that the proportion of girls born to couples using Gender Choice is not significantly greater than the number that we would expect by random chance.

Observations

- **Claim: For couples using the Gender Choice product, the proportion of girls is $p > 0.5$.**
- **Working assumption: The proportion of girls is $p = 0.5$ (with no effect from the Gender Choice).**
- The sample resulted in 52 girls among 100 births, so the sample proportion is $\hat{p} = 52/100 = 0.52$.
- **Assuming that $p = 0.5$, we use a normal distribution as an approximation to the binomial distribution to find that P (at least 52 girls in 100 births) = 0.3821.**
- There are two possible explanations for the result of 52 girls out of 100 children births: Either a random chance event (with probability 0.3821) has occurred, or the proportion of girls born to couples using Gender Choice is greater than 0.5.
- There isn't sufficient evidence to support Gender Choice's claim.

Components of a Formal Statistical Hypothesis Test

Null Hypothesis: H_0

- The Null Hypothesis (H_0) is a statement that the value of a population parameter (such as proportion, mean, or the standard deviation) is equal to some claimed value.
- We test the Null Hypothesis directly.
- Either reject H_0 or fail to reject (accept) H_0.

Alternative Hypothesis: (H_1 or H_a or H_A)

- **The Alternative Hypothesis (H_1 or H_a or H_A) is the statement that the parameter has a value that somehow differs from The Null Hypothesis.**

- **The symbolic form of the alternative hypothesis must use one of these symbols: $\neq$, $<$, $>$.**

Note about Forming Our Own Claims (Hypotheses)

If we are conducting a study and want to use a hypothesis test to support our claim, then the claim must be worded so that it becomes the alternative hypothesis.

Note about Identifying H_0 and H_1

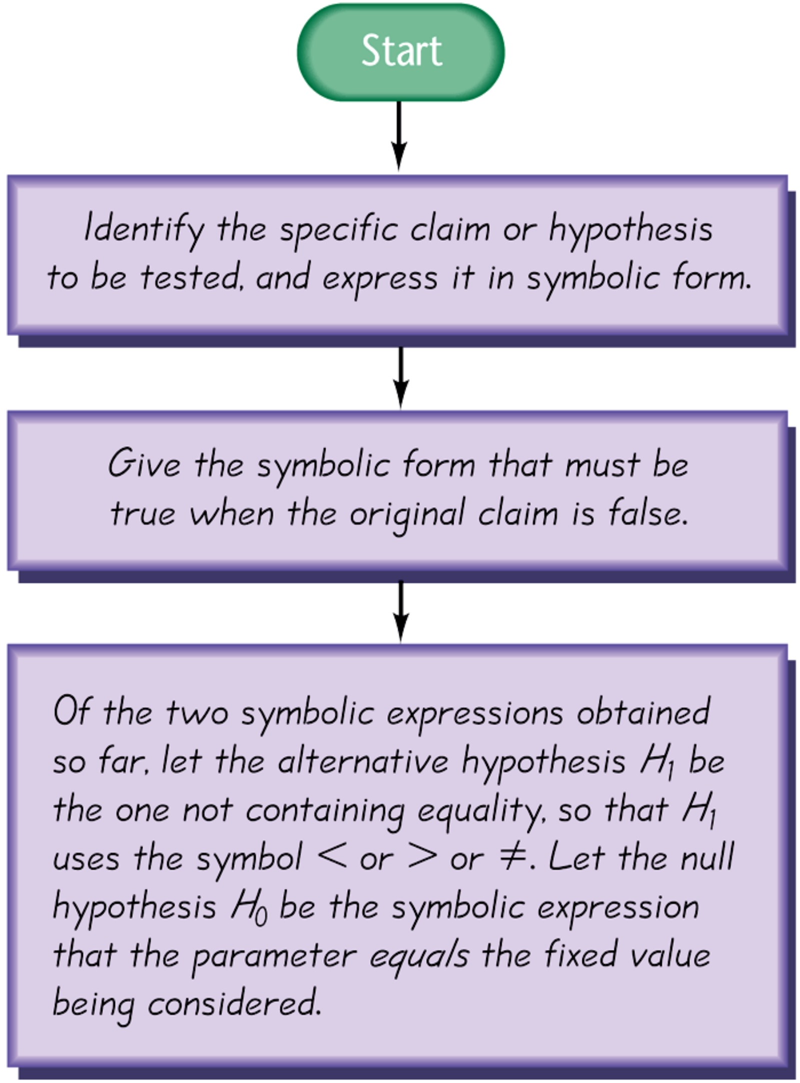


Figure 2

Example: Identify the Null and Alternative Hypotheses. Refer to Fig. 2 and use given claims to express corresponding null and alternative hypotheses in symbolic form.

a) The proportion of drivers who admit to running red lights is > 0.5.

b) The mean height of professional basketball players is at most 7 ft.

c) The standard deviation of IQ scores of actors is equal to 15.

Example: Identify the Null and Alternative Hypotheses. Refer to the Figure 2 and use the given claims to express the corresponding null and alternative hypotheses in symbolic form.

a) The proportion of drivers who admit to running red lights is greater than 0.5. In Step 1 of Figure 2, we express the given claim as $p > 0.5$. In Step 2, we see that if $p > 0.5$ is false, then $p \leq 0.5$ must be true. In Step 3, we see that the expression $p > 0.5$ does not contain equality, so we let the alternative hypothesis H_1 be $p > 0.5$, and we let H_0 be $p = 0.5$.

Example: Identify the Null and Alternative Hypotheses. Refer to the Figure 2 and use the given claims to express the corresponding null and alternative hypotheses in symbolic form.

b) The mean height of professional basketball players is at most 7 ft. In Step 1 of Figure 2, we can express that "The mean of at most 7 ft" it symbols as $\mu \leq 7$. In Step 2, we see that if $\mu \leq 7$ is false, then $\mu > 7$ must be true. In Step 3, we see that the expression $\mu > 7$ does not contain equality, so we let the alternative hypothesis H_1 be $\mu > 7$, and we let H_0 be $\mu = 7$.

Example: Identify the Null and Alternative Hypotheses. Refer to Figure 2 and use the given claims to express the corresponding null and alternative hypotheses in symbolic form.

c) The standard deviation of IQ scores of actors is equal to 15. In Step 1 of Figure 2, we express the given claim as $\sigma = 15$. In Step 2, we see that if $\sigma = 15$ is false, then $\sigma \neq 15$ must be true. In Step 3, we let the alternative hypothesis H_1 be $\sigma \neq 15$, and we let H_0 be $\sigma = 15$.

Test Statistic

The Test Statistic is a value used in making a decision about the null hypothesis, and is found by converting the sample statistic to a score with the assumption that the null hypothesis is true.

Test Statistic - Formulas

$$z = \frac{\hat{p} - p}{\sqrt{\frac{pq}{n}}}$$

Test Statistic for Proportions

$$z = \frac{\bar{x} - \mu_{\bar{x}}}{\frac{\sigma}{\sqrt{n}}}$$

Test Statistic for Mean

$$\chi^2 = \frac{(n-1)s^2}{\sigma^2}$$

Test Statistic for Standard Deviation

Example: A survey of n = 880 randomly selected adult drivers showed that 56% (or p = 0.56) of those respondents admitted to running the red lights. Find the value of test statistic for the claim that the majority of adult drivers admit to running red lights. (Assume that the required assumptions are satisfied and the focus must be on finding the indicated test statistic).

Solution: The example of the previous slide showed that the given claim results in the following null and alternative hypotheses: H_0: $p = 0.5$ and H_1: $p > 0.5$. Because we work under the assumption that the null hypothesis is true for a value of $p = 0.5$, we get the following test statistic:

$$z = \frac{\hat{p} - p}{\sqrt{\frac{pq}{n}}} = \frac{0.56 - 0.5}{\sqrt{\frac{(0.5)(0.5)}{880}}} = 3.56$$

Critical Region

The Critical Region (or Rejection Region) is the set of all values of the test statistic that cause us to reject the null hypothesis. For example, see the red-shaded region as shown in Figure 3.

Significance Level

The Significance Level (α) is the probability that the test statistic will fall in the critical region when the null hypothesis is actually true. Common choices for α are 0.05, 0.01, and 0.10.

Critical Value

A Critical Value is any value that separates the critical region (where we reject null hypothesis) from the values of the test statistic that do not lead to rejection of the null hypothesis. The critical values depend on the nature of the null hypothesis, sampling distribution that applies, and the significance level (α). Pl. See Figure 3 where a critical value of z = 1.645 corresponds to a significance level of $\alpha = 0.05$.

Test Statistic

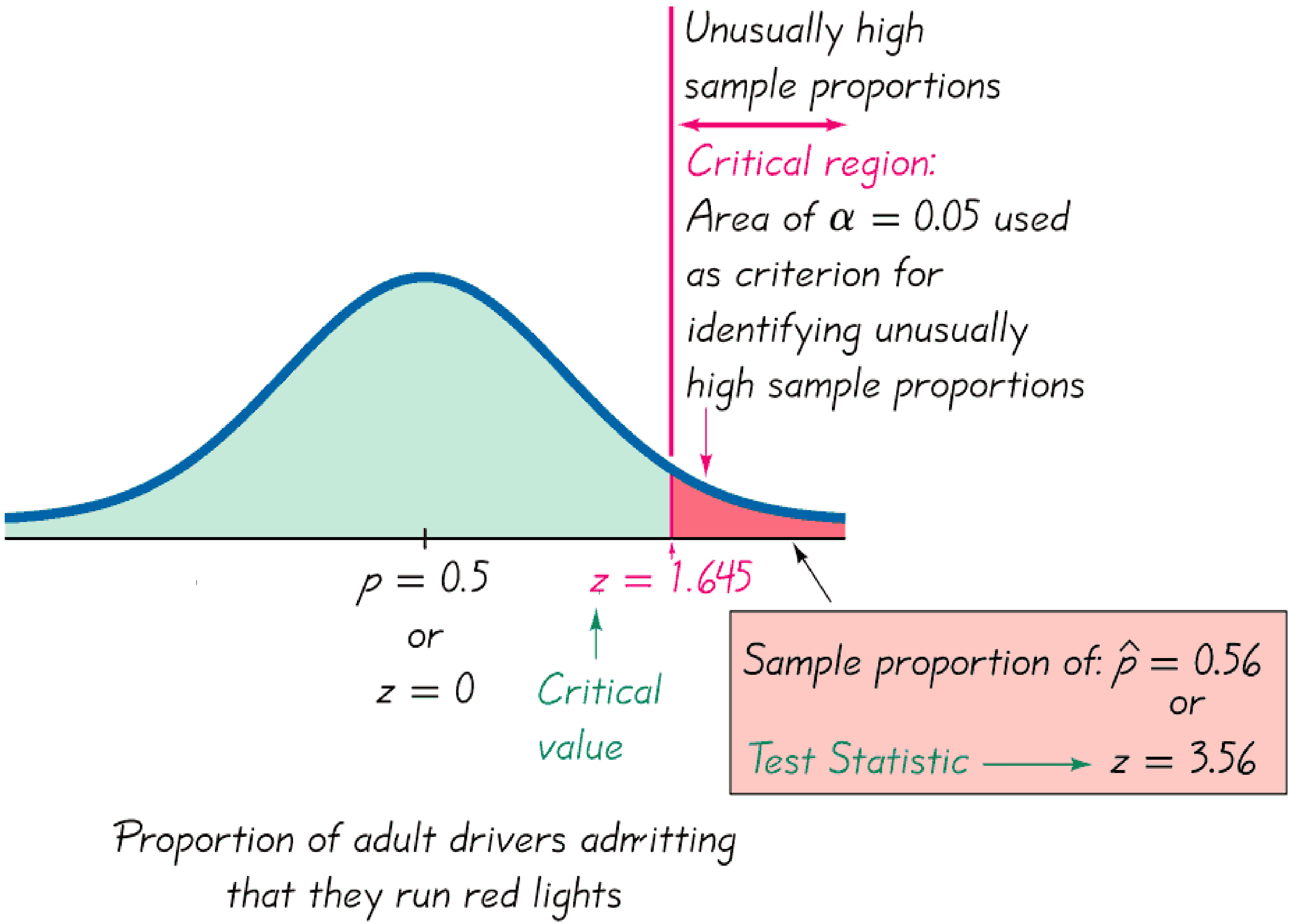


Figure 3

Two-tailed Test

H_0: =

H_1: ≠

The Tails in a distribution are the extreme regions bounded by critical values.

Significance level (α) is divided equally between the two tails of the critical region

Here ≠ means Less than or Greater than

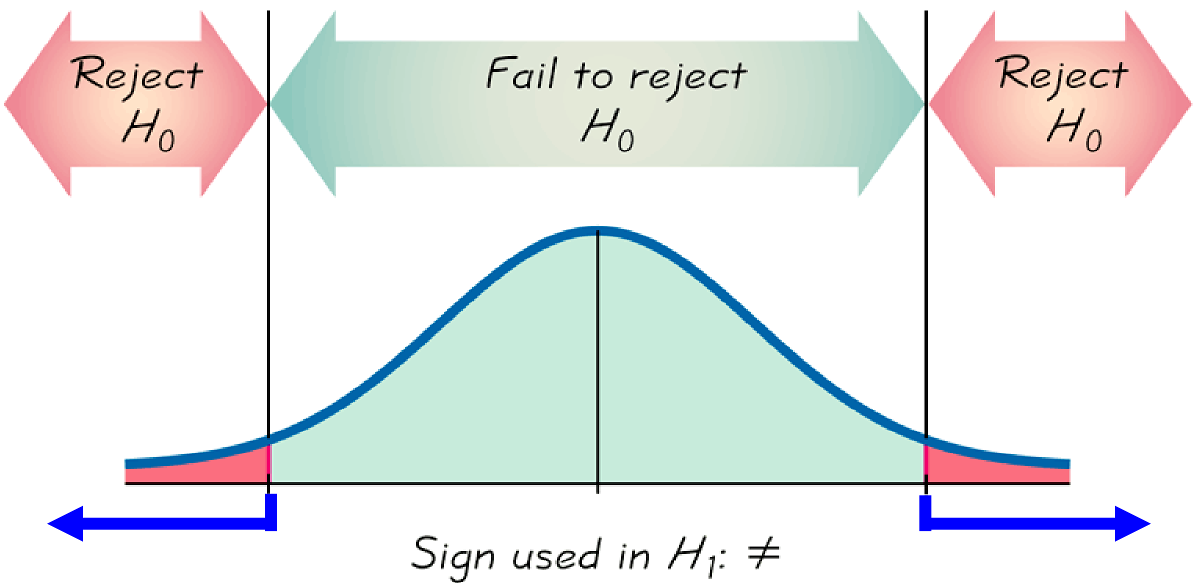


Figure 4

Right-tailed Test

H_0: =

H_1: >

Points Right

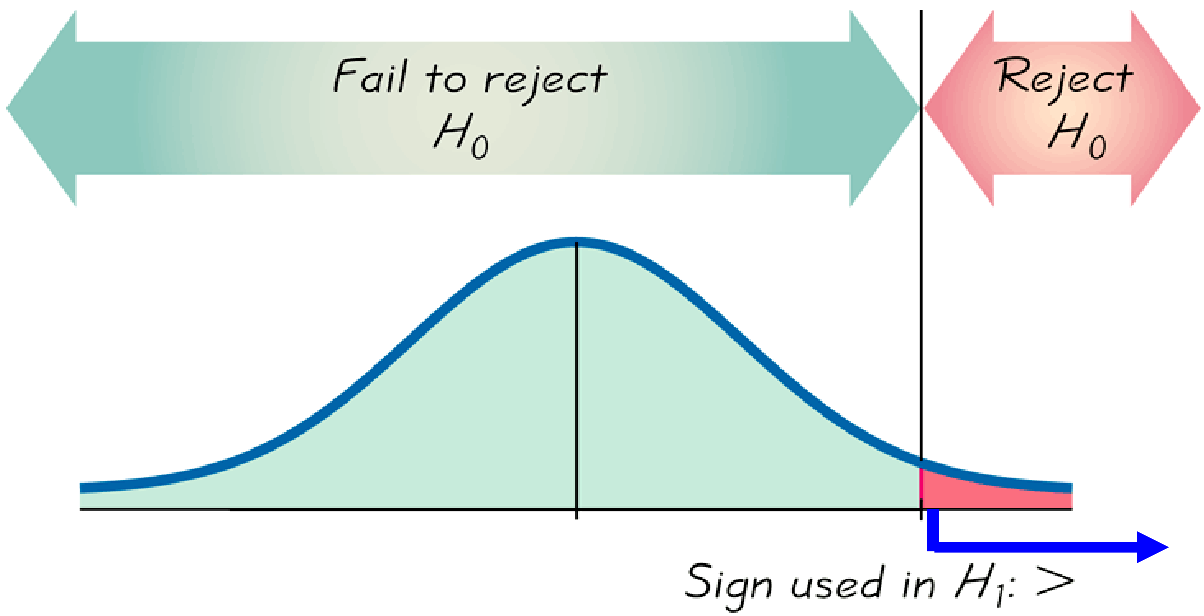


Figure 5

Left-tailed Test

H_0: =

H_1: <

Points Left

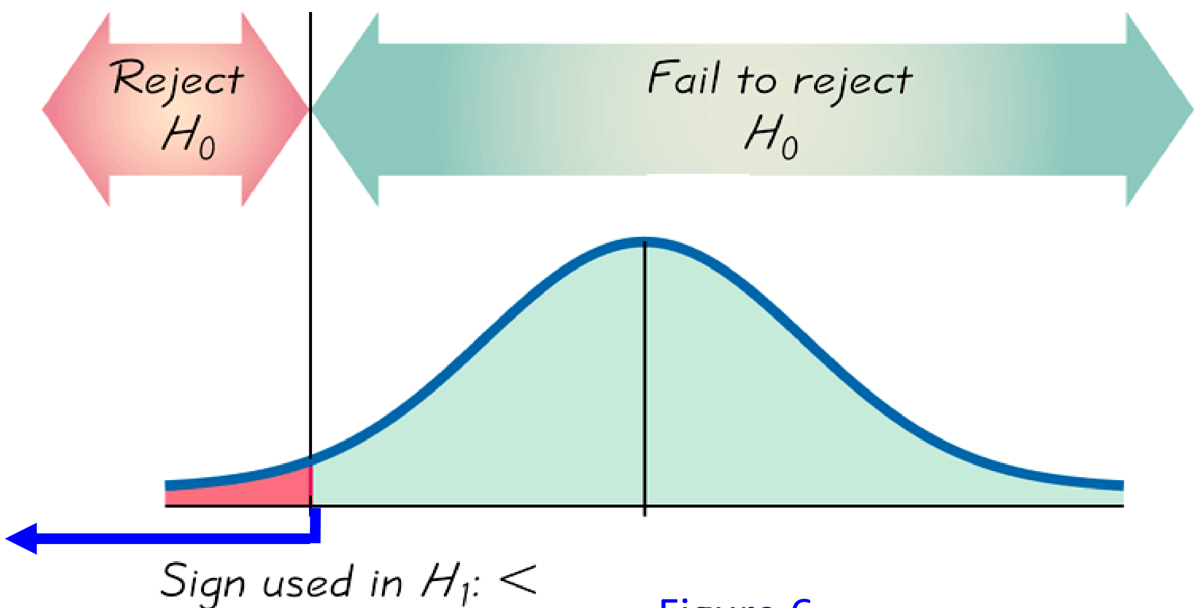


Figure 6

P-Value

The *P*-value (or) *p*-value (or) probability value is the probability of getting a value of the test statistic that is at least as extreme as the one representing the sample data, assuming that null hypothesis is true, and it is rejected if the *P*-value is very small, such as 0.05 or less.

Initial Conclusions in Hypothesis Testing

We always test the null hypothesis. The inference will be one of the following:

1. **Reject the Null Hypothesis.**

2. **Fail to Reject the Null Hypothesis.**

Decision Criterion

Traditional Method:

Reject the *Null Hypothesis* (H_0) if the test statistic falls within the critical region.

Fail to Reject the H_0 if the test statistic does not fall within the critical region.

Decision Criterion (Contd…)

P-value Method:

Reject H_0 **if the *P*-value $\leq \alpha$** (where α is the significance level, such as 0.05).

Fail to Reject H_0 if the *P*-value $> \alpha$.

Decision Criterion (Contd…)

Another Option:

Instead of using a significance level (α) such as 0.05, simply identify the *P*-value and leave the decision to the reader.

Decision Criterion (Contd…)

Confidence Intervals:

Because a confidence interval estimate of a population parameter contains likely values of that parameter, reject a claim that the population parameter has a value that is not included in the confidence interval.

Procedure for Finding *P*-Values

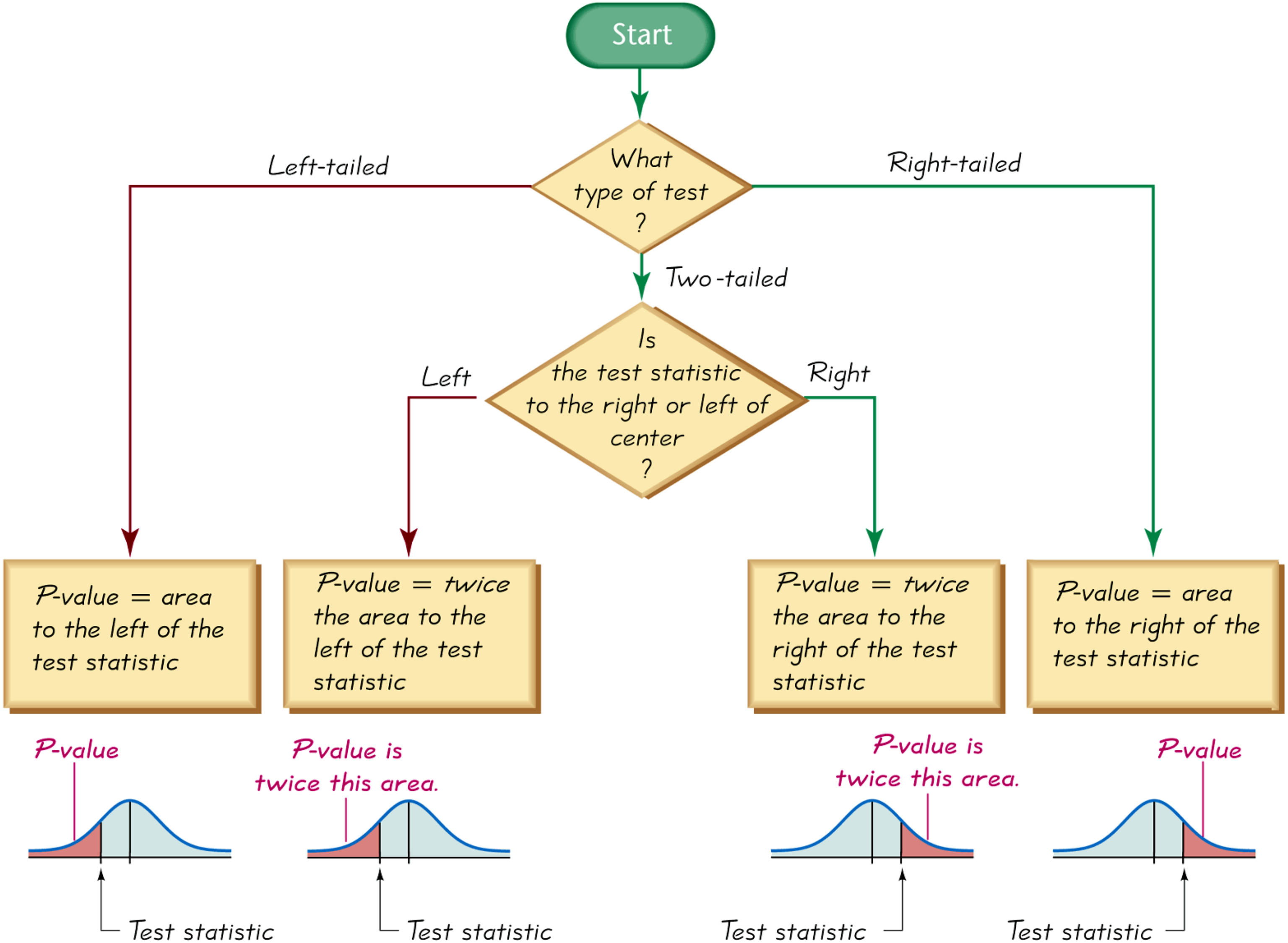


Figure 7

Wording of Final Conclusion

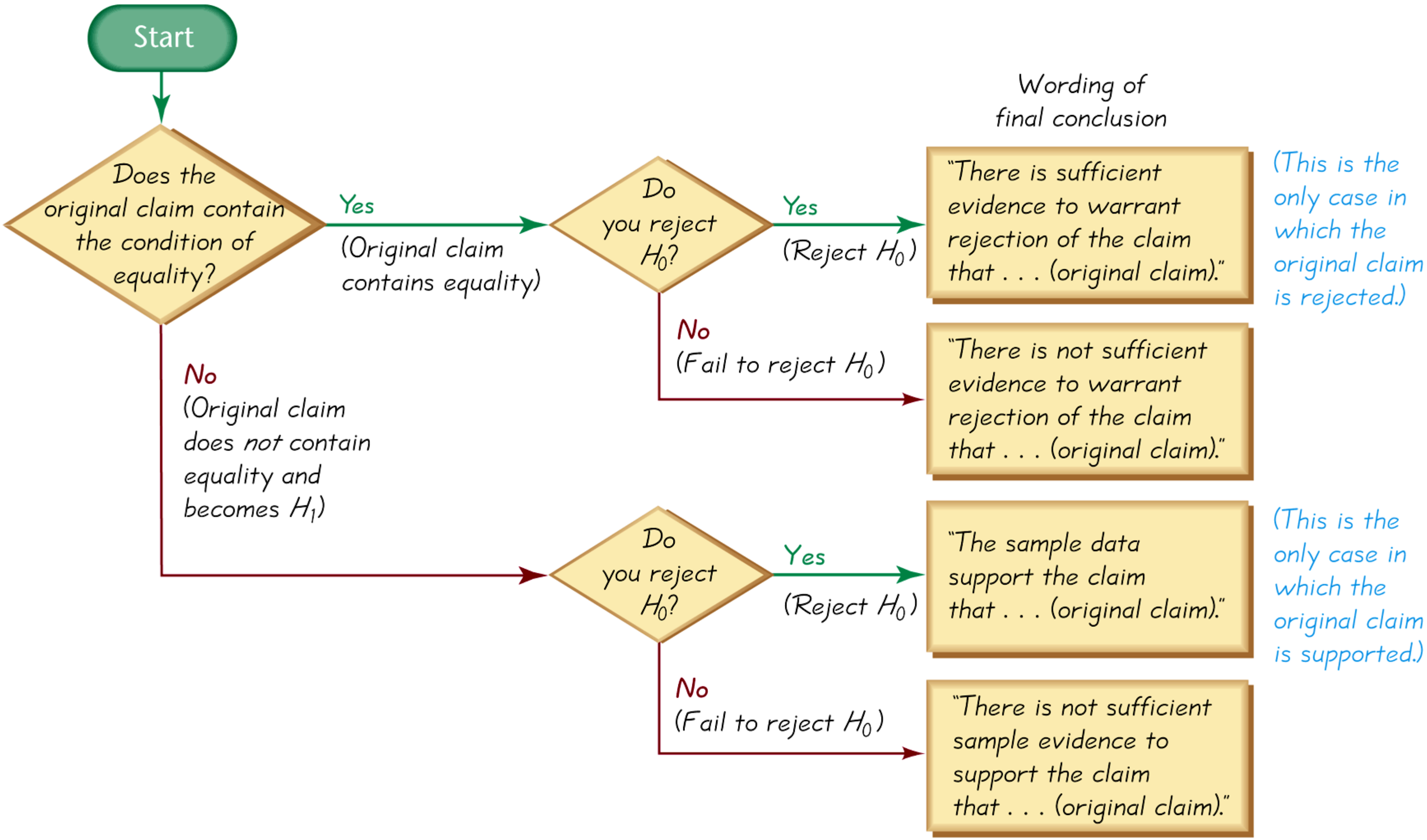


Figure 8

Accept Versus Fail to Reject

- Some texts use "accept the null hypothesis".
- We are not proving the null hypothesis.
- The sample evidence is not strong enough to warrant the rejection (such that there is not enough evidence to convict a suspect).

Type I Error

- **A Type I Error is the mistake of rejecting the null hypothesis when it is true.**

- **The symbol α (alpha) is used to represent the probability of a Type I error.**

Type II Error

- **A Type II Error is the mistake of failing to reject the null hypothesis when it is false.**

- **The symbol β (beta) is used to represent the probability of a Type II error.**

Type I and Type II Errors

Table: Type I and Type II Errors			
		True State of Nature	
		The null hypothesis is true	The null hypothesis is false
Decision	We decide to reject the null hypothesis	**Type I error** (rejecting a true null hypothesis) α	Correct decision
	We fall to reject the null hypothesis	Correct decision	**Type II error** (failing to reject a false null hypothesis) β

Controlling Type I and Type II Errors

- **For any fixed value of α, an increase in the sample size *n* will cause a decrease in β.**

- **For any fixed sample size *n*, a decrease in α will cause an increase in β. Conversely, an increase in α will cause a decrease in β.**

- **To decrease both α and β, we need to increase the sample size.**

Power of a Hypothesis Test

The Power of a Hypothesis Test is the probability (1 - β) of rejecting a false null hypothesis, which is computed by using a particular significance level α and a particular value of population parameter that is an alternative to the value which is assumed to be true in the null hypothesis. That is, the power of the hypothesis test is known as the probability of supporting an alternative hypothesis that is true.

Comprehensive Hypothesis Test: *P*-Value Method

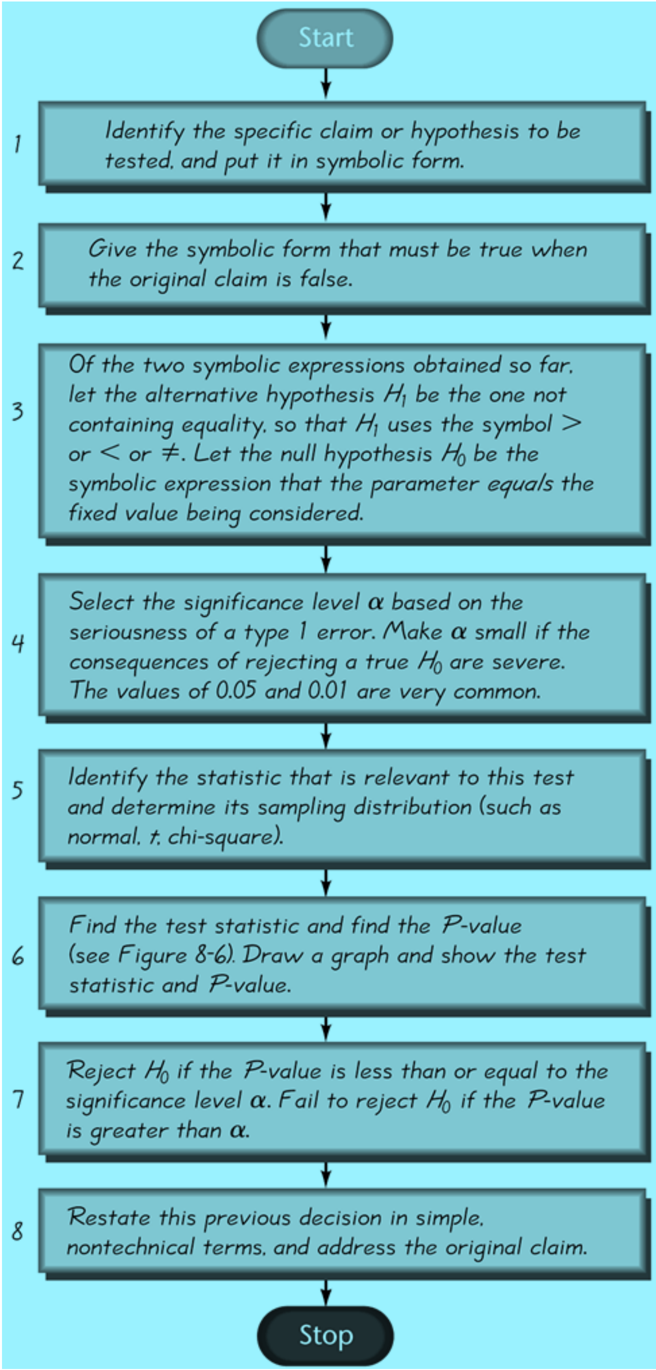


Figure 9

Comprehensive Hypothesis Test: Traditional Method

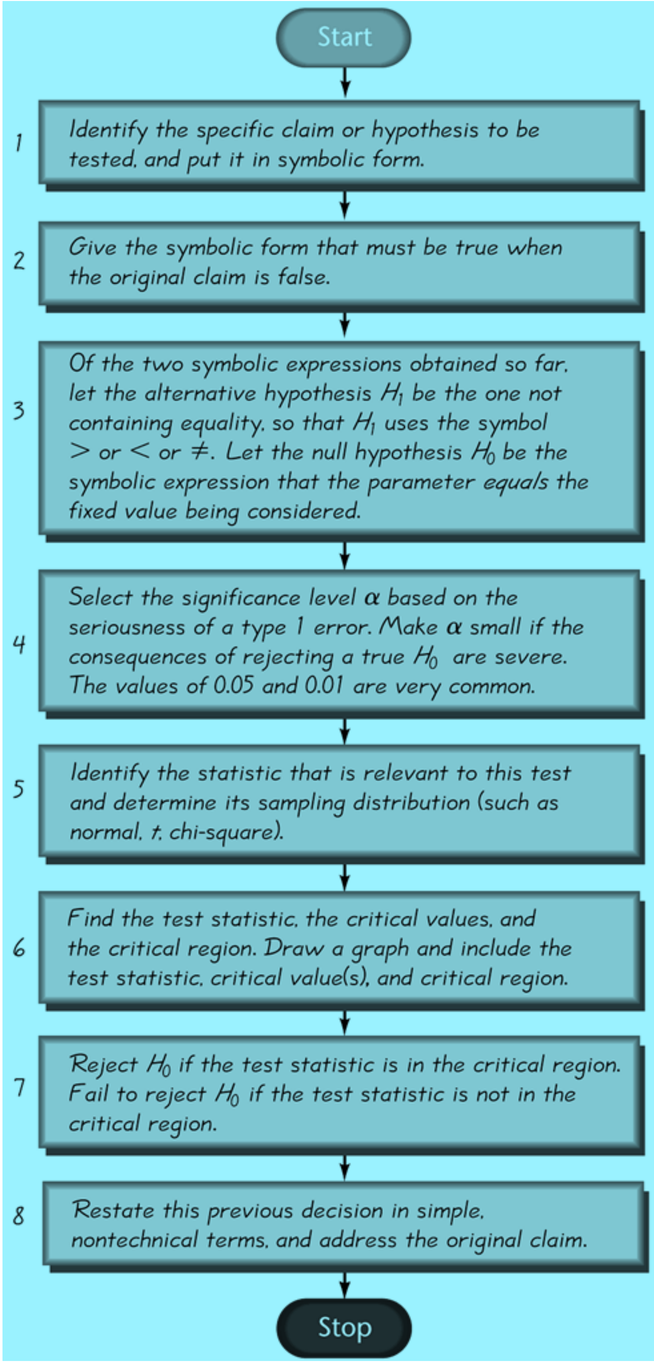


Figure 10

Comprehensive Hypothesis Test

A confidence interval estimate of a population parameter contains the values of that parameter. **We should therefore reject a claim that the population parameter has a value that is not included in the confidence interval.**

Table Confidence Level for Confidence Interval

		Two-Tailed Test	One-Tailed Test
Significance Level for Hypothesis Test	0.01	99%	98%
	0.05	95%	90%
	0.10	90%	80%

Summary of Hypothesis Testing

An **objective** method of making decisions or **inferences** from sample data (evidence)

Sample data used to choose between two choices i.e. **Hypotheses** or Statements about a population

We do this by comparing what we have observed to what we expected if one of the statements (**Null Hypothesis**) was true

The cat is faced with a decision to go out into the snow or not. It has two hypotheses; Null Hypothesis: Cat won't get its paws wet and be cold, Alternative Hypothesis: It will get its paws wet and will be cold. Cat could base its decision on data that is collected in its mind in the past; when it went out in the snow in the past its paws did get cold and wet and so the cat may make the decision not to go out based on that evidence.

Summary of Hypothesis Testing

Always we have Two Hypotheses:

H_1: Alternative (Research) Hypothesis

What we aim to gather evidence of.

Typically that there **is** a difference/effect/relationship etc.

H_0: Null Hypothesis

What we assume is true to begin with.

Typically that there is **no** difference/effect/relationship etc.

Summary of Hypothesis Testing

Define Empirical Research Question

↓

Set Null and Alternative Hypotheses

↓

Calculate a Test Statistic

↓

Calculate a p-value

↓

Make a Decision and Interpret Our Conclusions

Choose a Suitable Test

Summary of Hypothesis Testing

We can use statistical software to undertake a hypothesis test e.g. SPSS (Statistical Package for Social Sciences)

One part of the output is the p-value (P)

If P < **0.05 reject** H_0 (Reject Null Hypothesis) => **Evidence** of H_1 being true (i.e. **IS** association (Alternative Hypothesis))

If P > **0.05 do not** reject H_0 (Accept the Null Hypothesis) (i.e. **NO** association (Null Hypothesis))

What if P = 0.049 or 0.051? Discuss the fact that hypothesis testing involves the weight of evidence and "shades of grey" rather than being a clear cut decision making process.

Summary of Hypothesis Testing (Choosing the Right Test)

1) A clearly defined research question
2) What is the dependent variable and what type of variable is it?
3) How many independent variables are there and what data types are they?
4) Are you interested in comparing means or investigating relationships?
5) Do you have repeated measurements of the same variable for each subject?

Summary of Hypothesis Testing (Choosing the Right Test)

- Clarity of empirical research questions with measurable quantities
- Which variables will help answer these empirical research questions
- Think about what test is needed before carrying out a study so that the right type of variables are collected

Summary of Hypothesis Testing (Choosing the Right Test)

- How many variable are involved?
- Two – interested in the relationship
- One dependent and one independent
- One dependent and several independent variables: some may be controls
- Relationships between more than two: multivariate techniques (not covered here)

Summary of Hypothesis Testing (Choosing the Right Test)

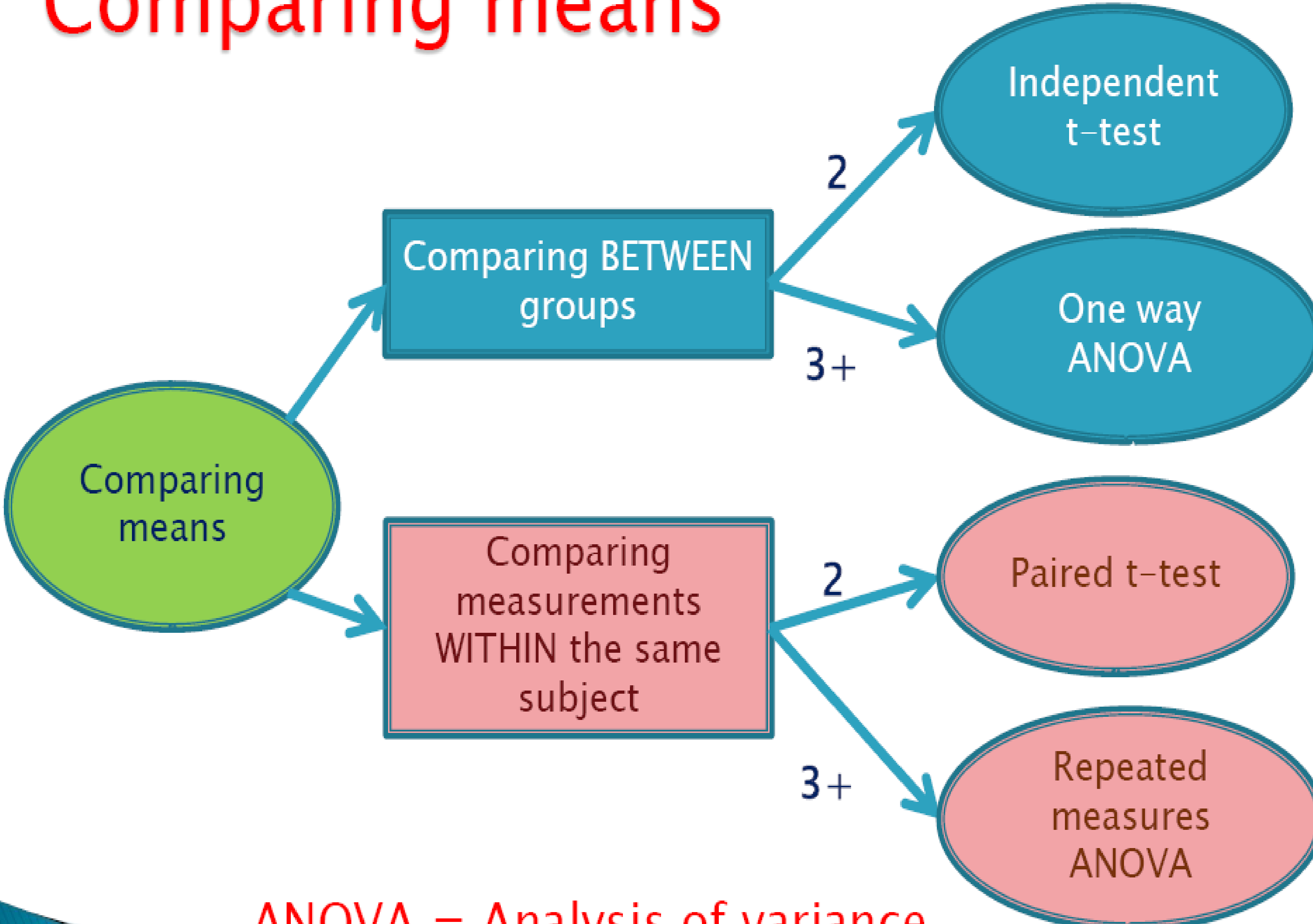
Comparing the Means

- Dependent = Scale
- Independent = Categorical

- How many means are we comparing?

- Do we have independent groups or repeated measurements on each person?

Comparing means



ANOVA = Analysis of variance

Exercise – Comparing the Means

Research Question	Dependent variable	Independent variable	Test
Do women do more housework than men?	Housework (hrs per week) (Scale)	Gender (Nominal)	Independent t-test
Does Margarine X reduce cholesterol? Everyone has cholesterol measured on 3 occasions	Cholesterol (Scale)	Occasion (Nominal)	Repeated measures ANOVA
Which of 3 diets is best for losing weight?	Weight lost on diet (Scale)	Diet (Nominal)	One-way ANOVA

Parametric or non-parametric?

Statistical tests fall into two types:

Parametric tests

Assume data follows a particular distribution e.g. normal

Non-parametric

Nonparametric techniques are usually based on ranks/ signs rather than actual data

Non-parametric Tests

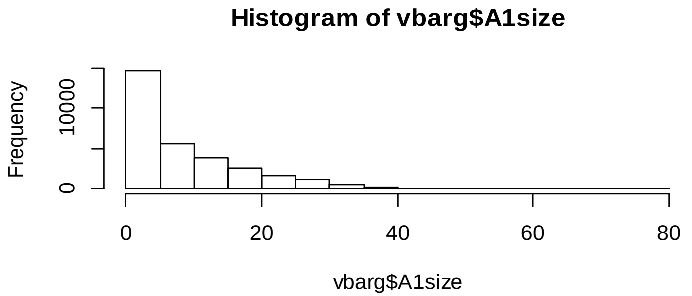
- Non-parametric methods are used when:
  - Data is ordinal
  - Data does not seem to follow any particular shape or distribution (e.g. Normal)
  - Assumptions underlying parametric test not met
  - A plot of the data appears to be very skewed
  - There are potential influential outliers in the dataset
  - Sample size is small

Note: Parametric tests are fairly robust to non-normality.
Data has to be very skewed to be a problem

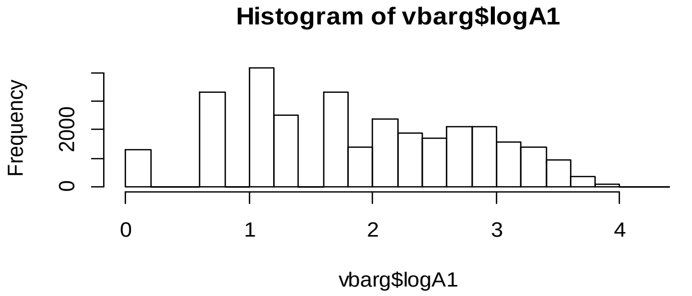
What can be done about non-normality?

If the data are not normally distributed, there are two options:

1. **Use a non-parametric test**
2. **Transform the dependent variable**



For positively skewed data, taking the log of the dependent variable often produces normally distributed values

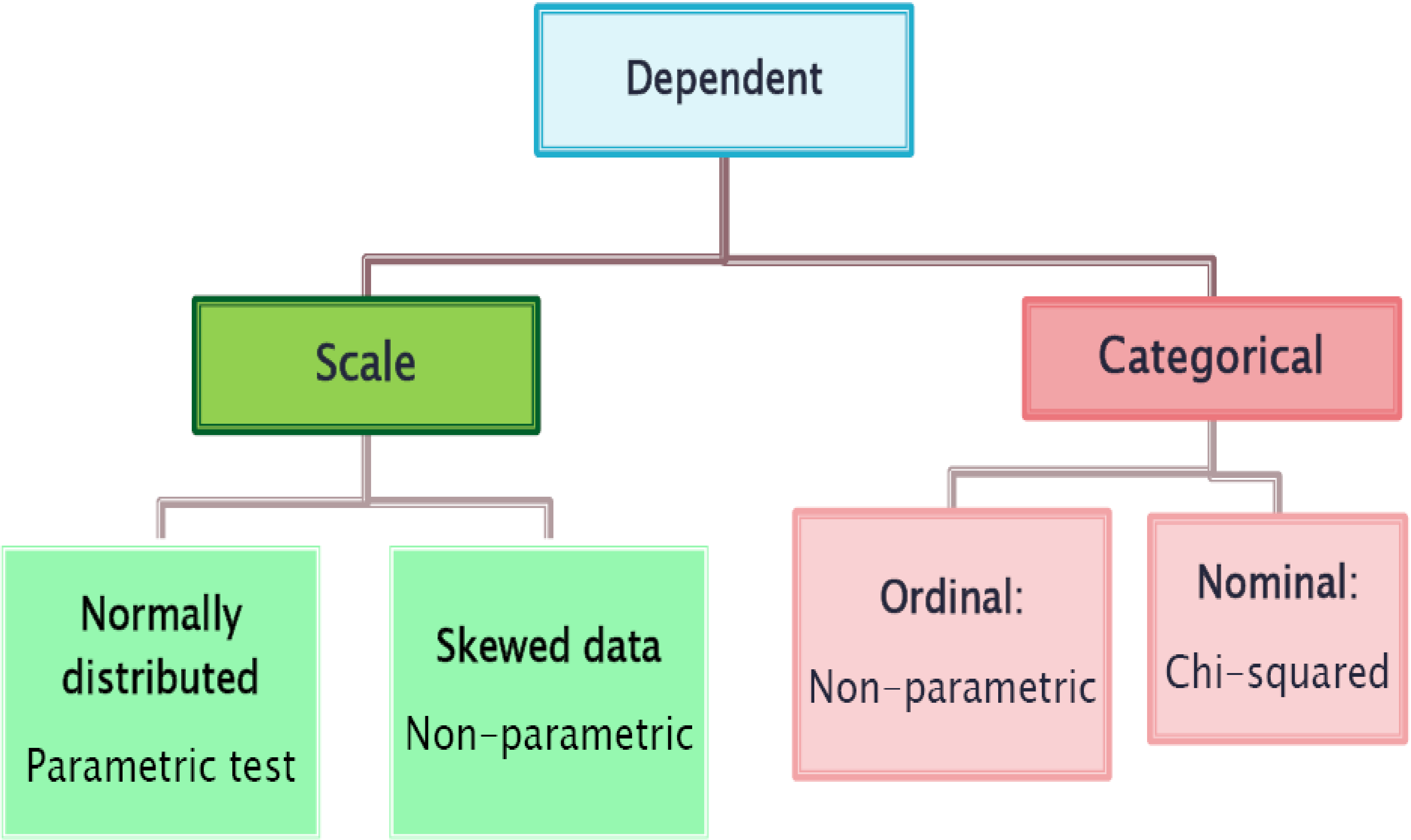


Non-parametric tests

Parametric test	What to check for normality	Non-parametric test
Independent t-test	Dependent variable by group	Mann-Whitney test
Paired t-test	Paired differences	Wilcoxon signed rank test
One-way ANOVA	Residuals/Dependent	Kruskal-Wallis test
Repeated measures ANOVA	Residuals	Friedman test
Pearson's Correlation Co-efficient	At least one of the variables should be normal	Spearman's Correlation Co-efficient
Linear Regression	Residuals	None - transform the data

Notes: The residuals are the differences between the observed and expected values.

Summary


Dependent
Scale
Categorical
Normally distributed
Parametric test
Skewed data
Non-parametric
Ordinal:
Non-parametric
Nominal:
Chi-squared

Statistical Hypothesis Testing: T-tests

(Paired or Independent (Unpaired) Data?)

We are often interested in comparing two sets of data, prior to analysis we must determine whether this data is paired or not.

T-tests are used to compare two population means

- **Paired Data:** Same individuals studied at two different times or under two conditions PAIRED T-TEST
- **Independent (Unpaired) Data:** Data is collected from two separate groups INDEPENDENT SAMPLES T-TEST (UNPAIRED T-TEST)

What is the t-distribution?

- The t-distribution is similar to the standard normal distribution but has an additional parameter called degrees of freedom (*df*). The df is calculated as the number of observations – 1.

  - For a paired t-test, *df* (or *v*) = number of pairs – 1

  - For an independent t-test, $v = n_{group1} + n_{group2} - 2$

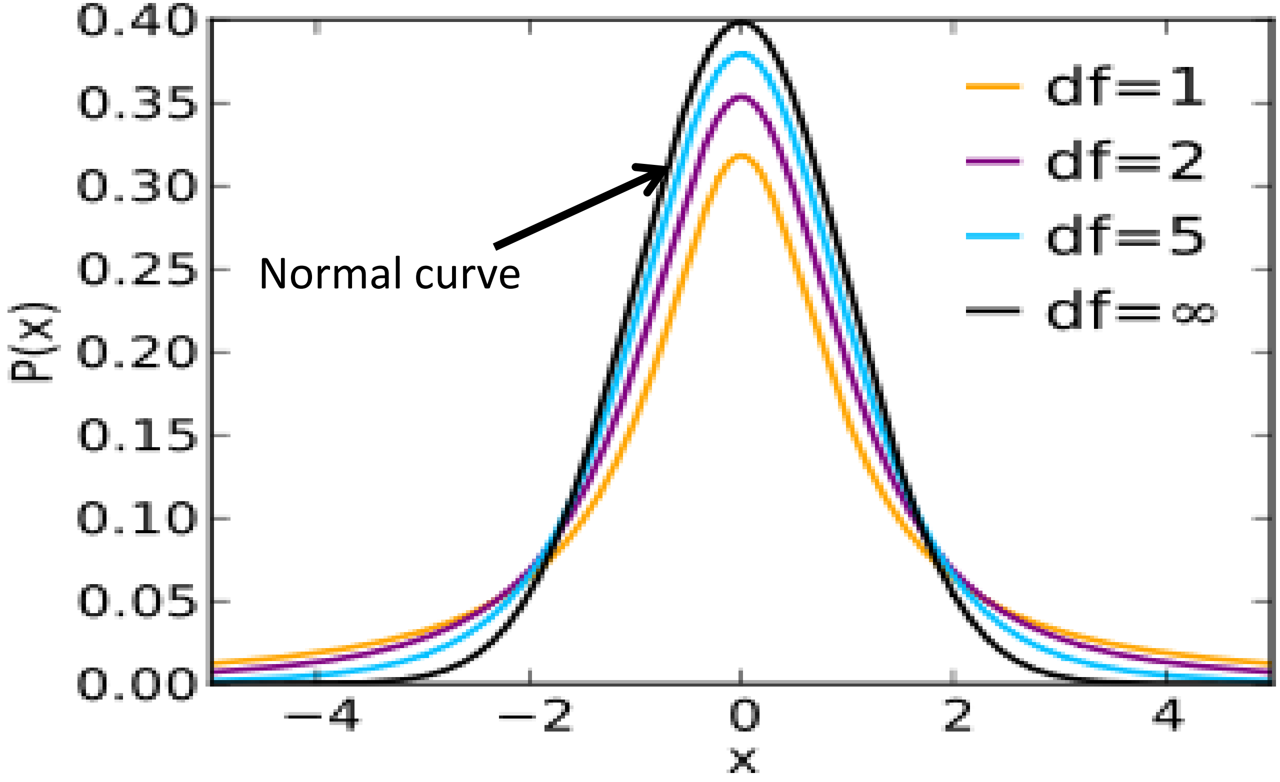
- **Used for small samples and when the population standard deviation is unknown**

- **Small sample sizes have heavier tails**

When we have small sample sizes (n < 30), we replace the normal distribution with ***Student's t distribution***, which has slightly less probability of being close to the mean and a somewhat larger probability of being in the tails. As the sample size increases, the critical values tend towards those of the normal distribution e.g. for a two tailed test with 5% significance, the critical value gets closer to 1.96, as the sample size increases.

Relationship to Normal Distrbution

As the sample size increases then the df increases, thus the t-distribution becomes more like the Normal Distribution.



Assumptions in t-Tests

Normality: Plot Histograms

One plot of the paired differences for any paired data.

Two (One for each group) for independent samples.

Don't have to be perfect, just roughly symmetric.

Equal Population Variances: Compare sample standard deviations

As a rough estimate, one should be no more than twice the other.

Do an F-test (Levene's in SPSS) to formally test for differences.

However the *t*-test is very robust to violations of the assumptions of Normality and equal variances, particularly for moderate (i.e. >30) and larger sample sizes.

Suggests ways of checking the assumptions in a t-test. For paired data it is the single column of differences that are assumed to be normal not each set of data from the two time points. The equal variances assumption does not apply to the paired t-test since there is only one sample (and population!)

What if the assumptions are not met?

There are alternative tests which do not have these assumptions

Test	Check	Equivalent non-parametric test
Independent t-test	Histograms of data by group	Mann-Whitney
Paired t-test	Histogram of paired differences	Wilcoxon signed rank

ANOVA Test

- Let us go through the procedure for one-way ANOVA
  - That means, one independent variable
- Multi-way ANOVA computations are cumbersome and very time consuming to do manually
  - So it is better to do computations using statistical packages

ANOVA Test

Compares the means of several groups.

- Which diet is the best?
  Dependent: Weight lost (Scale)
  Independent: Diet 1, 2 or 3 (Nominal)
- **Null Hypothesis: The mean weight lost on diets 1, 2 and 3 is the same.**
- **Alternative Hypothesis: The mean weight lost on diets 1, 2 and 3 are not the same.**

Summary Statistics

	Overall	Diet 1	Diet 2	Diet 3
Mean	3.85	3.3	3.03	5.15
Standard deviation	2.55	2.24	2.52	2.4
Number in group	78	24	27	27

- **Which diet was the best?**
- **Are the standard deviations similar?**

ANOVA Test

ANOVA = ANalysis Of VAriance

We compare variation **between** groups relative to variation **within** groups

Population variance estimated in two ways:

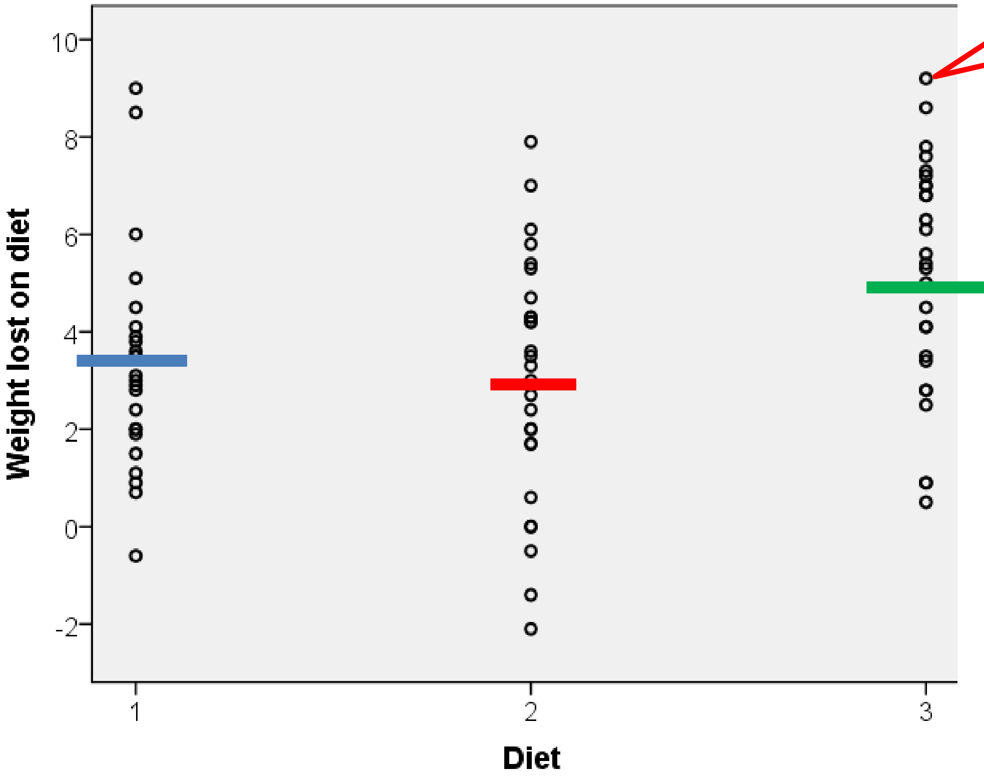
- One based on variation **between** groups we call the **M**ean **S**quare due to **T**reatments/ **MST/ MS$_{between}$**
- Other based on variation **within** groups we call the **M**ean **S**quare due to **E**rror/ **MSE/ MS$_{within}$**

Within the Group Variation

Residual =difference between an individual and the group mean

SS_{within}=sum of squared residuals

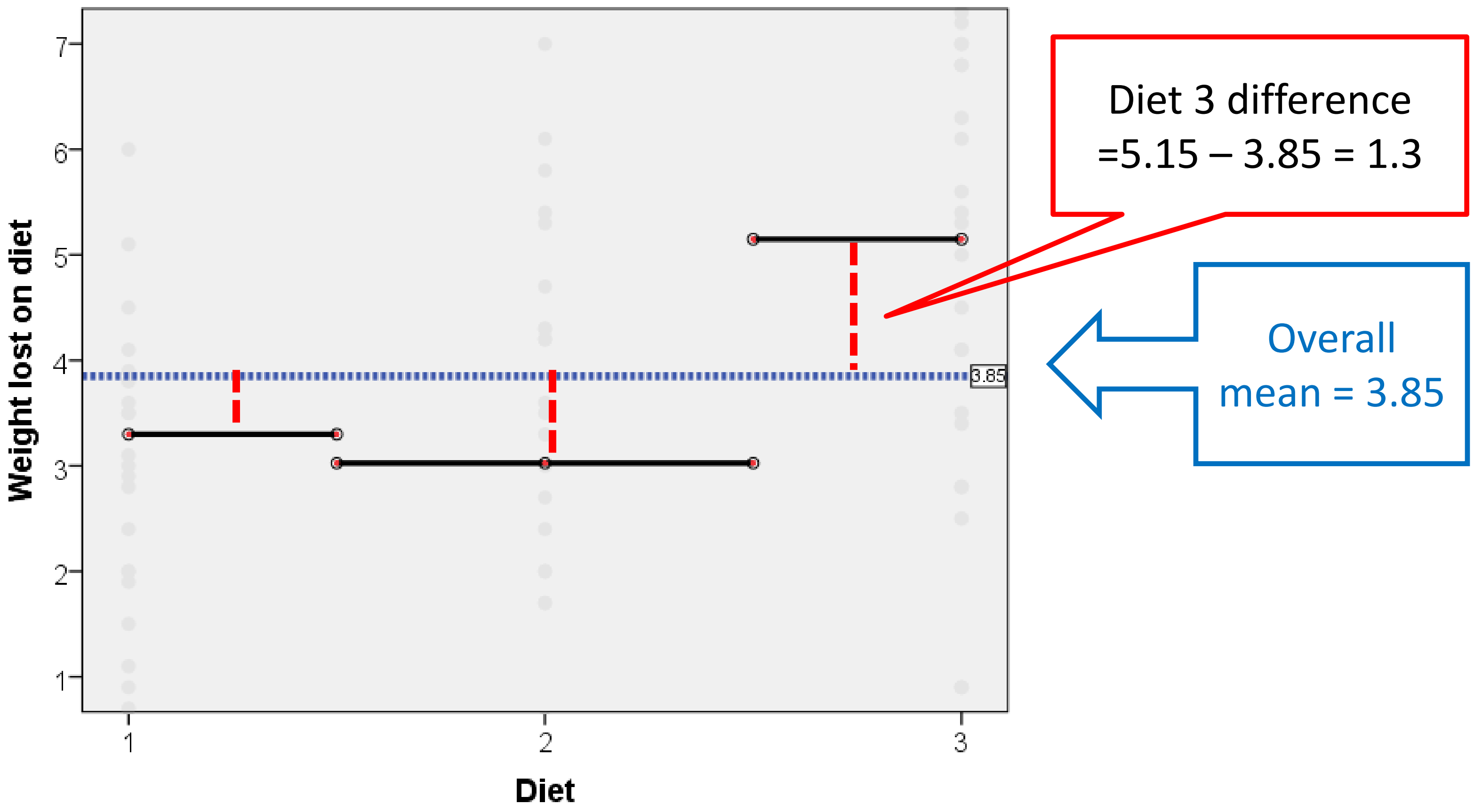
Person lost 9.2kg kg, hence residual =9.2 – 5.15 = 4.05



Mean weight lost on diet 3 = 5.15kg

Between the Group Variation

Differences between each group mean and the overall mean



Sum of Squares Calculations

K = Number of Groups

$$SS_{within} = \sum_{j=1}^{k}\sum_{i=1}^{n}\left(x_{ij} - \overline{x_j}\right)^2$$

$$= \sum_{i=1}^{24}\left(x_i - 3.3\right)^2 + \sum_{i=1}^{27}\left(x_i - 3.03\right)^2 + \sum_{i=1}^{27}\left(x_i - 5.15\right)^2 = 430.179$$

$$SS_{Between} = \sum_{j=1}^{k} n_j\left(\overline{x_j} - \overline{x_T}\right)^2$$

$$= 24\left(3.3 - 3.85\right)^2 + 27\left(3.03 - 3.85\right)^2 + 27\left(5.15 - 3.85\right)^2 = 71.094$$

ANOVA Test Statistics

Summary ANOVA

Source	Sum of Squares	Degrees of Freedom	Variance Estimate (Mean Square)	F Ratio
Between	SS_B	$K-1$	$MS_B = \frac{SS_B}{K-1}$	$\frac{MS_B}{MS_W}$
Within	SS_W	$N-K$	$MS_W = \frac{SS_W}{N-K}$	
Total	$SS_T = SS_B + SS_W$	$N-1$		

Test Statistic
(usually reported)

N = Total observations in all groups,

K = Number of groups

Test Statistic (by hand)

Filling in the boxes

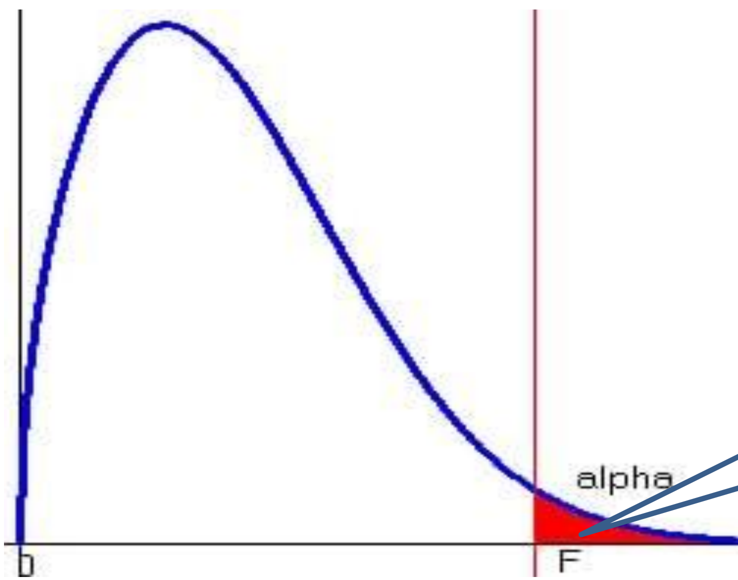
	Sum of Squares	Degrees of Freedom	Mean Square	F-ratio (Test Statistic)
$SS_{between}$	71.045	2	35.522	**6.193**
SS_{within}	430.180	75	5.736	
SS_{total}	501.275	77		

$$\text{F-ratio} = \frac{\text{Mean between group sum of squared differences}}{\text{Mean within group sum of squared differences}}$$

If F-ratio > 1, then there is a bigger difference between the groups than within the groups

P-value

- The p-value for ANOVA is calculated using the F-distribution
- If we repeated the experiment several times, then we would get a variety of test statistics



p-value = probability of getting a test statistic at least as extreme as ours if the null is true

Test Statistic

One Way ANOVA

$$\text{Test Statistic} = \frac{\text{between group variation}}{\text{within group variation}} = \frac{MS_{Diet}}{MS_{Error}} = 6.197$$

Tests of Between-Subjects Effects

Dependent Variable: Weight lost on diet (kg)

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	71.094[a]	2	35.547	6.197	.003
Intercept	1137.494	1	1137.494	198.317	.000
Diet	71.094	2	35.547	6.197	.003
Error	430.179	75	5.736		
Total	1654.350	78			
Corrected Total	501.273	77			

a. R Squared = .142 (Adjusted R Squared = .119)

$MS_{between}$

MS_{within}

There was a significant difference in weight lost between the diets (p=0.003)

Post-hoc Tests

If there is a significant ANOVA result, then Pairwise comparisons are made

These are t-tests with adjustments to keep the type 1 error to a minimum

- Tukey's and Scheffe's tests are the most commonly used post-hoc tests.
- Hochberg's GT2 is better where the sample sizes for the groups are very different.

Post-hoc Tests

Which diets are significantly different?

Multiple Comparisons

Dependent Variable: Weight lost on diet (kg)

	(I) Diet	(J) Diet	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Tukey HSD	1	2	.2741	.67188	.912	-1.3325	1.8806
		3	-1.8481*	.67188	.020	-3.4547	-.2416
	2	1	-.2741	.67188	.912	-1.8806	1.3325
		3	-2.1222*	.65182	.005	-3.6808	-.5636
	3	1	1.8481*	.67188	.020	.2416	3.4547
		2	2.1222*	.65182	.005	.5636	3.6808

Write up the results and conclude with which diet is the best.

Pairwise Comparisons

Test	p-value
Diet 1 vs Diet 2	P = 0.912
Diet 1 vs Diet 3	P = 0.02
Diet 2 vs Diet 3	P = 0.005

There is no significant difference between Diets 1 and 2 but there is between diet 3 and diet 1 (p = 0.02) and diet 2 and diet 3 (p = 0.005).

The mean weight lost on Diets 1 (3.3kg) and 2 (3kg) are less than the mean weight lost on diet 3 (5.15kg).

Assumptions for ANOVA Test

Assumption	How to check	What to do if assumption not met
Normality: The residuals (difference between observed and expected values) should be normally distributed	Histograms/ QQ plots/ normality tests of residuals	Do a Kruskall-Wallis test which is non-parametric (does not assume the normality)
Homogeneity of variance (each group should have a similar standard deviation)	Levene's test	Welch test instead of ANOVA and Games-Howell for post-hoc or Kruskall-Wallis

ANOVA Illustrated

- Let's illustrate the idea with the following example:

Suppose we have designed a new text entry technique for mobile phones. We think the design is good. In fact, we feel that our method is *better* than the most widely used state-of-the-art techniques: multi-tap and T9. We decide to undertake some empirical research to evaluate our design invention and to compare it with these current techniques?

Suppose "better" is defined in terms of error rate

Empirical Data

- In order to ascertain the validity of our claim, we conducted the experiments and thus collected the following empirical data (error rate of participants under different test conditions)

Participants	Our Method	Multi-tap	T9
1	3	5	7
2	2	2	4
3	1	4	5
4	1	2	3
5	4	3	6

ANOVA Steps - 1

- Now Let's compute means, standard deviations (SD) and variances for each test condition (over all participants)

	Our Method	Multi-tap	T9
Mean	2.20	3.20	5.00
SD	1.30	1.30	1.58
Variance	1.70	1.70	2.50

ANOVA Steps - 1

- **Also calculate "grands" – values involving all irrespective of groups**
  - **Grand Mean (mean of means) = 3.467**
  - **Grand SD (w.r.t. grand mean) = 1.767**
  - **Grand Variance (w.r.t. grand mean) = 3.124**

ANOVA Steps - 2

- **Calculate "total sum of squares (SS_T)"**

$$SS\_T = \sum (x\_i - mean\_grand)^2$$
$$= 43.74$$

Where, *x_i* is the error rate value of the i-th participant (among all)

ANOVA Steps - 2

- **An associated concept is the degrees of freedom (DoF (df)), which is the number of observations that are free to vary**
- **DoF (df) can be calculated simply as the (number of things used to calculate – 1)**
  - **For SS_T calculation, DoF (df) = N-1**

ANOVA Steps - 3

- **Next calculate the "model sum of square (SS_M)"**
  - **Calculate (mean_group_*i*-mean_grand) for the i-th group**
  - **Square the above**
  - **Multiply by *n_i*, the number of participants in the *i*-th group**
  - **Sum for all groups**

ANOVA Steps - 3

- **In the example,**

$$SS\_M = 5(2.200 - 3.467)^2 + 5(3.200 - 3.467)^2 + 5(5.000 - 3.467)^2$$

$$= 20.135$$

- **DoF (df) = Number of group means – 1**

= 3 - 1 = 2 (in our example)

ANOVA Steps - 4

- **Calculate the "residual sum of square (SS_R)" and the corresponding DoF**

  SS_R = SS_T – SS_M

  DoF (SS_R) = DoF (SS_T) – DoF (SS_M)

- **Thus, in this example,**

  SS_R = 43.74 – 20.14 = 23.60

  DoF (SS_R) = 14 – 2 = 12

ANOVA Steps - 5

- **Calculate two "average sum of squares" or "mean squares (MS)"**

- **Model MS (MS_M) = SS_M/DoF(SS_M)**

  = 20.135/2 = 10.067 (for our example)

- **Residue MS (MS_R) = SS_R/DOF(SS_R)**

  = 23.60/12 = 1.967 (for our example)

ANOVA Steps - 6

- Calculate the "F-ratio" (simply divide MS_M by MS_R)
  - F-ratio (F) = 10.067/1.967 = 5.12 (for our example)
- DoFs associated with the F-ratio are the DoFs used to calculate the DoF(SS_M) and DoF(SS_R)
  - In our case, these are 2, 12 respectively
- Hence, in our case, the F-ratio would be written as F(2, 12) i.e. F-ratio (F) = 5.12

ANOVA Steps - 6

- Look up the critical value of F-ratio (F)
  - The critical values for different “significance levels” / thresholds (α) are available in a tabular form
  - The critical values signifies the value of F that we would expect to get by chance for α% of tests

ANOVA Steps - 6

- Example (for more details, please see the next slide)
  - To find the critical value of $F(2, 12)$ from the Table for α=.05, look at 2nd column, 12th row for .05
  - Which is 3.89
  - That means, 3.89 is the F-value we would expect to get by chance for 5% of the tests.

df (Denominator)	p	df (Numerator)																
		1	2	3	4	5	6	7	8	9	10	15	20	25	30	40	50	1000
1	.05	161.45	199.50	215.71	224.58	230.16	233.99	236.77	238.88	240.54	241.88	245.95	248.01	249.26	250.10	251.14	251.77	254.19
	.01	4052.18	4999.50	5403.35	5624.58	5763.65	5858.99	5928.36	5981.07	6022.47	6055.85	6157.31	6208.74	6239.83	6260.65	6286.79	6302.52	6362.70
2	.05	18.51	19.00	19.16	19.25	19.30	19.33	19.35	19.37	19.38	19.40	19.43	19.45	19.46	19.46	19.47	19.48	19.49
	.01	98.50	99.00	99.17	99.25	99.30	99.33	99.36	99.37	99.39	99.40	99.43	99.45	99.46	99.47	99.47	99.48	99.50
3	.05	10.13	9.55	9.28	9.12	9.01	8.94	8.89	8.85	8.81	8.79	8.70	8.66	8.63	8.62	8.59	8.58	8.53
	.01	34.12	30.82	29.46	28.71	28.24	27.91	27.67	27.49	27.35	27.23	26.87	26.69	26.58	26.50	26.41	26.35	26.14
4	.05	7.71	6.94	6.59	6.39	6.26	6.16	6.09	6.04	6.00	5.96	5.86	5.80	5.77	5.75	5.72	5.70	5.63
	.01	21.20	18.00	16.69	15.98	15.52	15.21	14.98	14.80	14.66	14.55	14.20	14.02	13.91	13.84	13.75	13.69	13.47
5	.05	6.61	5.79	5.41	5.19	5.05	4.95	4.88	4.82	4.77	4.74	4.62	4.56	4.52	4.50	4.46	4.44	4.37
	.01	16.26	13.27	12.06	11.39	10.97	10.67	10.46	10.29	10.16	10.05	9.72	9.55	9.45	9.38	9.29	9.24	9.03
6	.05	5.99	5.14	4.76	4.53	4.39	4.28	4.21	4.15	4.10	4.06	3.94	3.87	3.83	3.81	3.77	3.75	3.67
	.01	13.75	10.92	9.78	9.15	8.75	8.47	8.26	8.10	7.98	7.87	7.56	7.40	7.30	7.23	7.14	7.09	6.89
7	.05	5.59	4.74	4.35	4.12	3.97	3.87	3.79	3.73	3.68	3.64	3.51	3.44	3.40	3.38	3.34	3.32	3.23
	.01	12.25	9.55	8.45	7.85	7.46	7.19	6.99	6.84	6.72	6.62	6.31	6.16	6.06	5.99	5.91	5.86	5.66
8	.05	5.32	4.46	4.07	3.84	3.69	3.58	3.50	3.44	3.39	3.35	3.22	3.15	3.11	3.08	3.04	3.02	2.93
	.01	11.26	8.65	7.59	7.01	6.63	6.37	6.18	6.03	5.91	5.81	5.52	5.36	5.26	5.20	5.12	5.07	4.87
9	.05	5.12	4.26	3.86	3.63	3.48	3.37	3.29	3.23	3.18	3.14	3.01	2.94	2.89	2.86	2.83	2.80	2.71
	.01	10.56	8.02	6.99	6.42	6.06	5.80	5.61	5.47	5.35	5.26	4.96	4.81	4.71	4.65	4.57	4.52	4.32
10	.05	4.96	4.10	3.71	3.48	3.33	3.22	3.14	3.07	3.02	2.98	2.85	2.77	2.73	2.70	2.66	2.64	2.54
	.01	10.04	7.56	6.55	5.99	5.64	5.39	5.20	5.06	4.94	4.85	4.56	4.41	4.31	4.25	4.17	4.12	3.92
11	.05	4.84	3.98	3.59	3.36	3.20	3.09	3.01	2.95	2.90	2.85	2.72	2.65	2.60	2.57	2.53	2.51	2.41
	.01	9.65	7.21	6.22	5.67	5.32	5.07	4.89	4.74	4.63	4.54	4.25	4.10	4.01	3.94	3.86	3.81	3.61
12	.05	4.75	3.89	3.49	3.26	3.11	3.00	2.91	2.85	2.80	2.75	2.62	2.54	2.50	2.47	2.43	2.40	2.30
	.01	9.33	6.93	5.95	5.41	5.06	4.82	4.64	4.50	4.39	4.30	4.01	3.86	3.76	3.70	3.62	3.57	3.37
13	.05	4.67	3.81	3.41	3.18	3.03	2.92	2.83	2.77	2.71	2.67	2.53	2.46	2.41	2.38	2.34	2.31	2.21
	.01	9.07	6.70	5.74	5.21	4.86	4.62	4.44	4.30	4.19	4.10	3.82	3.66	3.57	3.51	3.43	3.38	3.18
14	.05	4.60	3.74	3.34	3.11	2.96	2.85	2.76	2.70	2.65	2.60	2.46	2.39	2.34	2.31	2.27	2.24	2.14
	.01	8.86	6.51	5.56	5.04	4.69	4.46	4.28	4.14	4.03	3.94	3.66	3.51	3.41	3.35	3.27	3.22	3.02
15	.05	4.54	3.68	3.29	3.06	2.90	2.79	2.71	2.64	2.59	2.54	2.40	2.33	2.28	2.25	2.20	2.18	2.07
	.01	8.68	6.36	5.42	4.89	4.56	4.32	4.14	4.00	3.89	3.80	3.52	3.37	3.28	3.21	3.13	3.08	2.88
16	.05	4.49	3.63	3.24	3.01	2.85	2.74	2.66	2.59	2.54	2.49	2.35	2.28	2.23	2.19	2.15	2.12	2.02
	.01	8.53	6.23	5.29	4.77	4.44	4.20	4.03	3.89	3.78	3.69	3.41	3.26	3.16	3.10	3.02	2.97	2.76
17	.05	4.45	3.59	3.20	2.96	2.81	2.70	2.61	2.55	2.49	2.45	2.31	2.23	2.18	2.15	2.10	2.08	1.97
	.01	8.40	6.11	5.18	4.67	4.34	4.10	3.93	3.79	3.68	3.59	3.31	3.16	3.07	3.00	2.92	2.87	2.66
18	.05	4.41	3.55	3.16	2.93	2.77	2.66	2.58	2.51	2.46	2.41	2.27	2.19	2.14	2.11	2.06	2.04	1.92
	.01	8.29	6.01	5.09	4.58	4.25	4.01	3.84	3.71	3.60	3.51	3.23	3.08	2.98	2.92	2.84	2.78	2.58
19	.05	4.38	3.52	3.13	2.90	2.74	2.63	2.54	2.48	2.42	2.38	2.23	2.16	2.11	2.07	2.03	2.00	1.88
	.01	8.18	5.93	5.01	4.50	4.17	3.94	3.77	3.63	3.52	3.43	3.15	3.00	2.91	2.84	2.76	2.71	2.50
20	.05	4.35	3.49	3.10	2.87	2.71	2.60	2.51	2.45	2.39	2.35	2.20	2.12	2.07	2.04	1.99	1.97	1.85
	.01	8.10	5.85	4.94	4.43	4.10	3.87	3.70	3.56	3.46	3.37	3.09	2.94	2.84	2.78	2.69	2.64	2.43
22	.05	4.30	3.44	3.05	2.82	2.66	2.55	2.46	2.40	2.34	2.30	2.15	2.07	2.02	1.98	1.94	1.91	1.79
	.01	7.95	5.72	4.82	4.31	3.99	3.76	3.59	3.45	3.35	3.26	2.98	2.83	2.73	2.67	2.58	2.53	2.32

Implication

- Thus, we get the critical value = 3.89 for $F(2,12)$, $\alpha = 0.05$

- Note that $F(2, 12)=5.12$ > the critical value
  – Implies that the effect of test conditions has a significant effect on the outcome w.r.t. $\alpha=.05$

Reporting F-Statistic

- We can report the result as “our proposed method has a significant effect on reducing user errors [$F(2,12)=5.12$, $p < 0.05$] as compared to the other methods”.

- If it is found that the effect is not significant, it is reported as “our method has no significant effect on reducing user errors [$F(1,9)=0.634$, ns] as compared to the other methods”.

A Note of Caution

- **ANOVA requires that**
  - Empirical Data should have normally distributed sampling distribution and from a normally distributed population
  - Variances in each experimental condition are fairly similar
  - Observations should be independent
  - Dependent variables are measured on at least an interval scale
- The first two may be ignored if group sizes are equal
  - Otherwise, ALL conditions MUST have to be met

HCI: Empirical Research Case Study

Learning Objective

- In the previous lectures, we discussed different empirical research methods involved in HCI
- We introduced several concepts such as testable empirical question formulation, experiment design, data collection and statistical analysis of data
- In this lecture, we shall consider a case study

Case Study

- Suppose, we want to study the application of eye tracking technology for the text entry task (i.e., typing through eye gaze). Let us initiate an empirical inquiry to explore performance limits and the capabilities of various feedback modalities for keys in on-screen keyboards used with eye gaze based typing.
  - Suppose four feedback modalities are considered by us, namely: Audio only [A], Click+Visual [C], Speech+Visual [S], Visual only [V].

Steps in Empirical Research (Classical View)

Hypothesis = Research Question

Phenomenon = Interaction between the Human and the Computer (Technology)

- **Four steps…**

1. Observe and describe a **phenomenon**
2. Formulate an **hypothesis** to explain it
3. Use the hypothesis test to **predict** or **describe** **other phenomena**

Other Interactions

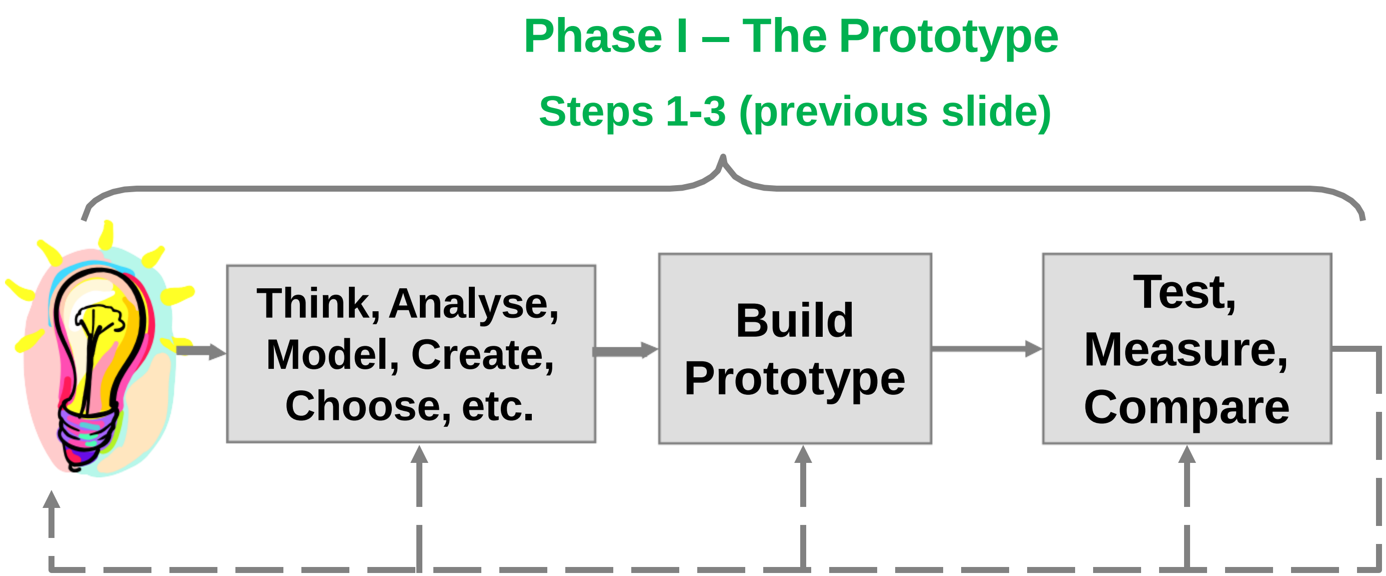
Hypothesis Testing

4. Perform **experiment**

Experiment = User Study

Predict = Predictive Model
Describe = Descriptive Model

Steps in Empirical Research (Practical View)

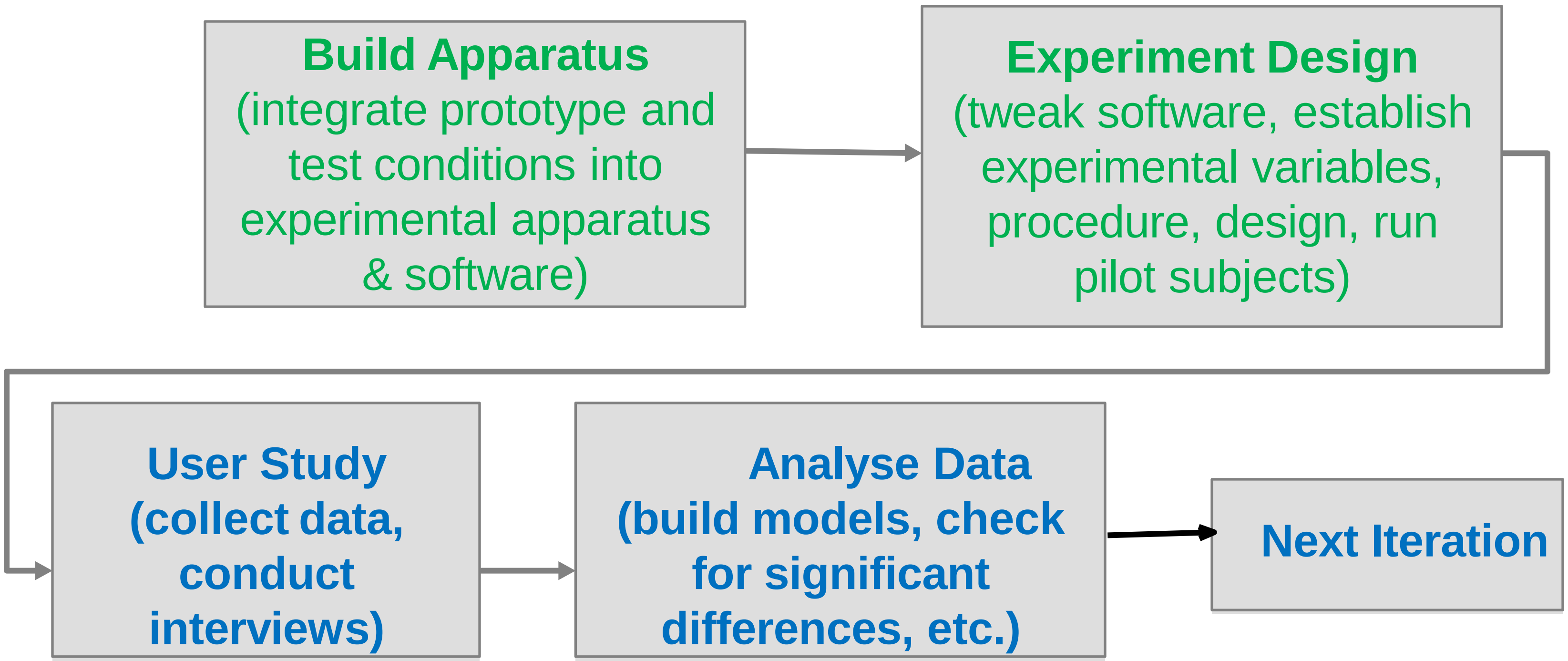


Iterations are frequent, unstructured, intuitive, informed, …

Empirical Research Questions "take shape" (i.e., certain measurable aspects of the interaction suggest "test conditions", and "tasks" for empirical inquiry

Steps in Empirical Research (Practical View)

Phase II – The User Study



The User Study

- Describe the participants employed for our study
  - Thirteen, volunteers, recruited from university campus, age, gender, computer experience, eye tracking/typing experience
- Apparatus
  - Describe hardware and software, etc.

The User Study

- Experiment Design
  - We decided to have a 4 x 4 repeated measures design
  - There are two independent variables (factors) with four levels each
    - Feedback modality (with the levels A, C, S, V)
    - Users were asked to enter blocks of text at a time and four such blocks were there for each user. So, “block” is a factor with four levels 1, 2, 3, 4

Note: Audio only [A], Click+Visual [C], Speech+Visual [S], Visual only [V].

The User Study

- Experiment Design
  - We have identified dependent variables (measures)
    - Speed of text entry (in “words per minutes”)
    - Accuracy of text entry (in “percentage of characters in error”)
    - Key selection activity (in “keystrokes per character”)
    - Also… responses to “broader” questions
  - Order of Conditions
    - Feedback modality order differed for each user (Latin Square)

The User Study

- Procedure for Data Collection
  - We first explained to the participants the general objectives of the experiment
  - Then the eye tracking apparatus was calibrated
  - The participants were put through some practice trials for familiarization
  - Afterwards, let us begin data collection

The User Study

- Procedure for Data Collection
  - Phrases of text presented to the participants by experimental software
  - Participants instructed to enter phrases “as quickly and accurately as possible”
  - Five phrases were entered by the participants per block
  - Total number of phrases entered in this experiment is found to be 13 x 4 x 4 x 5 = 1040

Experiment Replication

- The description of the experimental methodology (i.e., participants, participant selection, apparatus, design, procedure) must be sufficient to allow the experiment to be replicated by other researchers
  - This is necessary to allow the possibility for the results to be verified or refuted as part of performance evaluation
  - **An experiment that cannot be replicated is useless**

Data Tables

- Next slide contains example data on text entry speed, recorded in this empirical study (user study)
  - Create a Table to arrange data i.e. Data Table
  - From the Data Table, let us calculate other quantities such as the grand mean = 6.96 WPM
  - The Data Table also allows us to make salient observations (for example, 4th block speed for best condition was…)

Data Tables

Note: Audio only [A], Click+Visual [C], Speech+Visual [S], Visual only [V].

Factors and Levels

Speed																
	A	A	A	A	C	C	C	C	S	S	S	S	V	V	V	V	
Participant	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	Mean
1	6.17	7.19	7.04	7.09	6.76	7.40	7.54	7.94	6.44	6.17	7.84	6.81	5.20	6.29	7.39	7.63	6.93
2	6.71	7.25	7.05	7.15	7.73	7.57	8.04	7.26	7.00	6.75	7.68	7.46	7.50	7.07	7.32	7.06	7.29
3	6.80	6.65	7.62	7.98	6.61	7.18	7.34	8.19	6.65	7.53	7.09	7.90	5.73	7.24	6.94	7.13	7.16
5	6.30	6.31	7.59	7.38	6.85	7.64	7.58	7.88	7.07	6.43	7.26	7.65	6.75	6.59	6.97	7.72	7.12
7	6.68	6.89	7.32	7.51	7.00	7.81	7.64	7.2[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]72	7.57	7.20	7.11
8	6.08	6.55	6.83	5.92	7.44	6.93	7.56	6.41	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	7.9[illegible]3	7.45	7.16	6.98
9	7.62	7.01	6.60	7.07	6.91	6.81	6.91	7.73	6.50	7.57	7.59	7.80	6.62	7.06	7.16	7.41	7.15
10	5.88	5.71	7.33	7.11	6.66	7.97	7.64	8.15	6.35	7.21	6.56	7.33	5.00	6.97	6.54	6.36	6.80
12	6.89	7.61	7.42	7.88	7.79	8.28	8.20	8.39	6.62	6.87	7.99	8.23	**9.57**	8.17	7.91	7.09	7.81
13	6.85	6.57	8.14	6.00	5.92	7.89	7.49	6.98	6.05	7.45	5.34	7.46	7.21	6.81	6.80	8.24	6.95
14	5.37	5.56	**6.04**	6.86	6.20	6.82	7.71	7.76	5.85	6.37	6.74	6.69	5.98	6.43	6.38	5.87	6.41
15	5.51	6.12	6.32	7.00	6.16	6.49	7.21	7.19	5.65	6.52	6.49	7.10	5.31	6.88	6.36	6.93	6.45
16	5.88	7.18	5.95	6.00	4.85	6.98	7.37	6.98	6.88	6.21	4.96	5.34	6.72	7.14	4.96	6.80	6.26
																	6.96

Outlier

Each cell is the mean for five phrases of input

Statistical Analysis of Data

- The data recorded in the Data Table is analyzed statistically to identify the significant effects
- For example, we may have the following findings:
  - Significant effect for Feedback mode [$F(3,36)=8.77$, $p<.0005$]
  - Insignificant Effect for Feedback mode by the block interaction [$F(9,108)=0.767$, ns]

Data Tables

- Apart from the main tables, other tables are also created, which helps in making more useful observations
- The next slide shows an example of a summary table created from the data on text entry speed

Data Tables

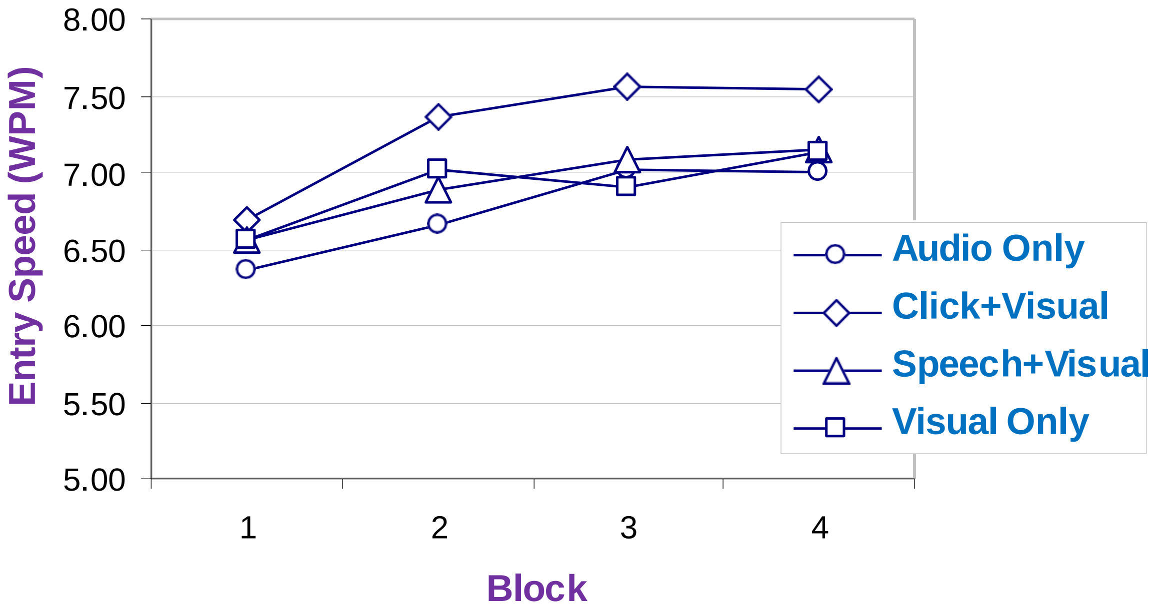
Speed (WPM)					
	Feedback Mode				
Block	Audio Only	Click+Visual	Speech+Visual	Visual Only	Mean
1	6.36	6.68	6.56	6.55	**6.54**
2	6.66	7.37	6.88	7.02	**6.98**
3	7.02	7.56	7.09	6.90	**7.14**
4	**7.00**	**7.55**	**7.14**	**7.12**	**7.20**
Mean	**6.76**	**7.29**	**6.92**	**6.90**	**6.97**

5.7% faster on 4th block

Each cell is the mean for 13 participants

Charts/Graphs

- Also let us create graphs/charts to visualize findings



The Broader Questions

- Along with the data analysis, an empirical study typically collects the direct feedback from all the participants on “broader” questions
  - For example, all participants can be asked about their preferences, satisfaction levels or even their suggestions for further improvements

The Broader Questions

- In the study, we asked the participants to rank (between 1 to 4) the feedback mode based on their personal preferences
- We obtained the following results:
  - Six of thirteen participants gave a 1st place ranking to the fastest feedback modality

The Broader Questions

- The results obtained are not strong enough to come to any conclusions
  - A reason may be that the differences just weren't large enough for the participants to really tell the difference in overall performance
- However, we have also made another observation, namely ten of the thirteen participants gave a 1st or 2nd place ranking to the fastest feedback modality
  - This can be treated as a strong indication that the better performance yields a better preference rating

What's Missing?

- **The case study just described show that the user study involves collection and analysis of usage data as well as participants' feedback**
- However, that's not all (it misses an important aspect of empirical research)
  - There is no theoretical account of the phenomena

What’s Missing?

- There is no delineation, description, categorization of known and observed behaviors (…that can form such a theoretical account)
- It is not sufficient to simply observe and conclude, it is also necessary to theorize about the observations (e.g., why the text entry speed is the least in a particular feedback mode)

Empirical Research in HCI

- The direct conclusions from observations help us to decide an interaction method; a theory about observed behavior can help us do much more
  - Such theories can eliminate the need for further investigations as well as can suggest the suitable ways for further improvement
- Such theories, if found, are another motivation for conducting empirical research in HCI (in fact, many models in HCI have been derived empirically)

Case Study: The Case for a Model

- Is there a “model of interaction” suggested by the observations in this case study?

- Perhaps. Here’s one possibility

  – All gaze point changes were logged as “events”.
  What was the total number of such events?
  Are there categories of such events?

- The identification, labelling, and tabulation of such could form the basis of a model of interaction for eye typing

Analysis of Variance (ANOVA)

Analysis of variance (ANOVA)

- is a statistical technique that is used to check if the means of two or more groups are significantly different from each other.
- ANOVA checks the impact of one or more factors by comparing the means of different samples.
- One-way or two-way refers to the number of independent variables in the analysis of variance test.

Steps:

1. Define Hypothesis (Null and alternative)
2. Calculate the sum of squares
3. Determine degrees of freedom

4. Find the mean for each of the groups.
5. Find the **overall mean** (the mean of the groups combined).
6. Find the **Within Group Variation**; the total deviation of each member's score from the Group Mean.
7. Find the **Between Group Variation**: the deviation of each Group Mean from the Overall Mean.
8. Find the F statistic: the ratio of Between Group Variation to Within Group Variation.

Formulas Used

One-Way ANOVA Table					
Source	Degrees of Freedom DF	Sum of Squares SS	Mean Square MS	F-Stat	P-Value
Between Groups	$k - 1$	SS_B	$MS_B = SS_B / (k - 1)$	$F = MS_B / MS_W$	Right tail of $F(k\text{-}1, N\text{-}k)$
Within Groups	$N - k$	SS_W	$MS_W = SS_W / (N - k)$		
Total:	$N - 1$	$SS_T = SS_B + SS_W$			

Between Groups Degrees of Freedom: $\mathbf{DF = k - 1}$, where $\mathbf{k}$ is the number of groups

Within Groups Degrees of Freedom: $\mathbf{DF = N - k}$, where $\mathbf{N}$ is the total number of subjects

Total Degrees of Freedom: $\mathbf{DF = N - 1}$

Sum of Squares Between Groups: $\mathbf{SS_B = \sum_{i=1}^{k} n_i (\bar{x}_i - \bar{x})^2}$, where $\mathbf{n_i}$ is the number of subjects in the i-th group

Sum of Squares Within Groups: $\mathbf{SS_W = \sum_{i=1}^{k} (n_i - 1) S_i^2}$, where $\mathbf{S_i}$ is the standard deviation of the i-th group

Total Sum of Squares: $\mathbf{SS_T = SS_B + SS_W}$

Mean Square Between Groups: $\mathbf{MS_B = SS_B / (k - 1)}$

Mean Square Within Groups: $\mathbf{MS_W = SS_W / (N - k)}$

F-Statistic (or F-ratio): $\mathbf{F = MS_B / MS_W}$

$$SS_{within} = \Sigma\,(X_i - \overline{X}_j)^2$$

where

- $\overline{X}_j$ denotes a group mean;
- X_i denotes an individual observation (“data point”).

URLs for ANOVA test

- https://goodcalculators.com/one-way-anova-calculator/

- ANOVA Calculator | AAT Bioquest (https://www.aatbio.com/tools/anova-analysis-of-variance-one-two-way-calculator)

Example 1:

- Suppose we want to know whether or not three different exam prep programs lead to different mean scores on a certain exam. To test this, we recruit 30 students to participate in a study and split them into three groups. The students in each group are randomly assigned to use one of the three exam prep programs for the next three weeks to prepare for an exam. At the end of the three weeks, all of the students take the same exam. The exam scores for each group are shown below:

Group 1	Group 2	Group 3
85	91	79
86	92	78
88	93	88
75	85	94
78	87	92
94	84	85
98	82	83
79	88	85
71	95	82
80	96	81

Group	Degrees of Freedom (DF)	Sum of Squares (SS)	Mean Square (MS)	F-Statistic	P-Value
Between Groups	2	192.2	96.1	2.3575	0.1138
Within Groups	27	1100.6	40.763		
Total	29	1292.8			

- α (significance level) = 0.05
- DF1 (numerator degrees of freedom) = df treatment = 2
- DF2 (denominator degrees of freedom) = df error = 27
- We find that the F critical value is **3.3541**.
- F test statistic in the ANOVA table is less than the F critical value in the F distribution table, we fail to reject the null hypothesis. This means we don't have sufficient evidence to say that there is a statistically significant difference between the mean exam scores of the three groups.

- To find P-value please refer

https://www.statology.org/here-is-how-to-find-the-p-value-from-the-f-distribution-table/

https://www.statology.org/f-distribution-calculator/

Example2:

Participant	Test Condition			
	A	B	C	D
1	11	11	21	16
2	18	11	22	15
3	17	10	18	13
4	19	15	21	20
5	13	17	23	10
6	10	15	15	20
7	14	14	15	13
8	13	14	19	18
9	19	18	16	12
10	10	17	21	18
11	10	19	22	13
12	16	14	18	20
13	10	20	17	19
14	10	13	21	18
15	20	17	14	18
16	18	17	17	14
Mean	14.25	15.13	18.75	16.06
SD	3.84	2.94	2.89	3.23

Key Parameters

Source	F-statistic	P-value
Between Groups	5.7587	0.0016

ANOVA Summary

Group	Degrees of Freedom (DF)	Sum of Squares (SS)	Mean Square (MS)	F-Statistic	P-Value
Between Groups	3	182.1719	60.724	5.7587	0.0016
Within Groups	60	632.6875	10.5448		
Total	63	814.8594			

HCI: Dialog Design

Learning Objective

- One key aspect of HCI is the dialog (communication), which is the interaction that takes place between a human user and the computer (machine)
- In this lecture, we shall learn about the representation, modelling and analysis of dialogs

Dialog

- A dialog refers to the *structure* of the interaction
- Dialog in HCI can be analyzed at three levels:
  - Lexical - At this level, the details such as the shape of icons, actual keys pressed etc. are dealt with
  - Syntactic - The order of inputs and outputs in an interaction are described at this level
  - Semantic - The effect a dialog has on the internal application/data is the subject matter at this level

Dialog Representation

- We need (formal) techniques to represent dialogs, which serves two purposes
  - It helps to understand the proposed design better
  - Formal representation makes it possible to analyze dialogs to identify usability problems (e.g., we can answer questions such as "does the design *actually* support *undo*"?)

Dialog Representation

- There are several formalisms that we can use to represent dialogs
- We shall discuss three of these formalisms in this lecture:
  - The state transition networks (STN)
  - The state charts (Finite State Machine (FSM))
  - The (classical) Petri-Nets

State Transition Network (STN)

- STNs are the most intuitive among all formalisms
- It assumes that a dialog essentially refers to a progression from one state of the system to the next in the system state space (in fact this assumption holds for all formalisms that we shall discuss)

State Transition Network (STN)

- The syntax of an STN is simple and consists of the following two entities
  - Circles: a circle in a STN refers to a state of the system, which is labeled (by giving a name to the state)
  - Arcs: the circles are connected with arcs, each of which refers to the action/event (represented by arc labels) that results in the system making a transition from the state where the arc originates to the state where it terminates

State Transition Network (STN)

Let’s illustrate the idea with an example. Suppose, we are using a drawing interface that allows us to draw lines and circles, by choosing appropriate menu item. To draw a circle, we need to select a center and circumference. A line can be drawn by selecting points on the line. How can we model this dialog using an STN?

State Transition Network (STN)

- So, what are states and transitions here?
  - We shall have a “start” state
  - From this “start” state, we shall go to a “menu” state, where we are shown the menu options. If we select the circle option, we go to a “circle” state. Otherwise, we select the “line” option and go to the “line” state

State Transition Network (STN)

- So, what are states and transitions here?
  - While at the “circle” state, we select a point as the circle center (through mouse click, say), which takes us to the “center” state
  - In the “center” state, we select the circle periphery (through mouse movement, say) and double click to indicate the end of input (the “finish” state). At this stage, the circle is displayed

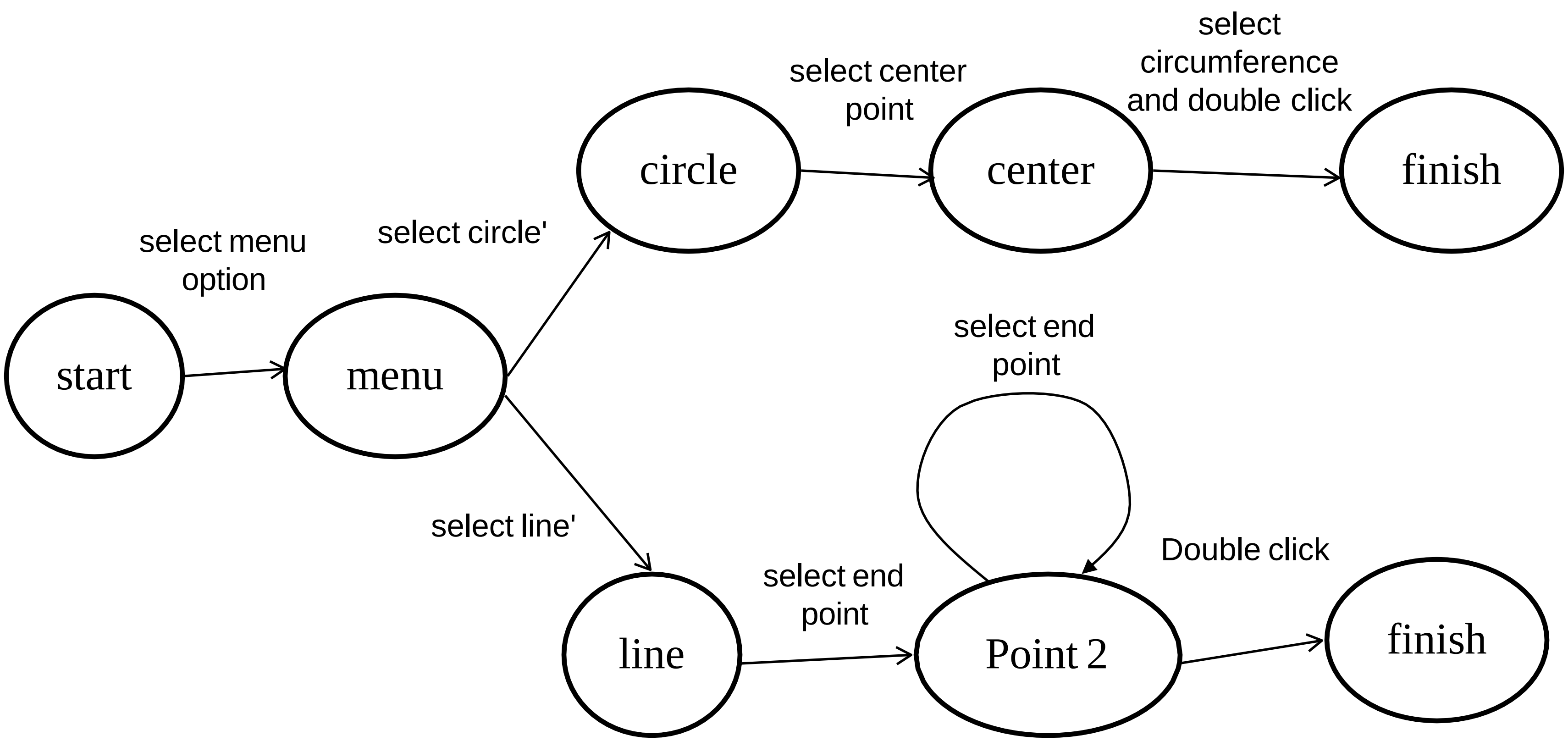
State Transition Network (STN)

- So, what are states and transitions here?
  - While at the “line” state, we select a point as the beginning of the line (through mouse click, say)
  - Then, we select another point to denote the last point on the line and transit to “point 2”. At this stage, a line is displayed between the two points

State Transition Network (STN)

- So, what are states and transitions here?
  - We can select another point, while at “point 2” to draw another line segment between this point and the point last selected. We can actually repeat this as many times as we want, to draw line of arbitrary shape and size
  - When we perform a double click, it indicates the end of input and the dialog comes to the “finish” stage

State Transition Network (STN)



STN – Pros and Cons

- Pros
  - Intuitive
  - Easy to understand
- Cons
  - Good for simple systems
  - Quickly becomes messy as the number of states/arcs grow

How to Model Complex Dialogs

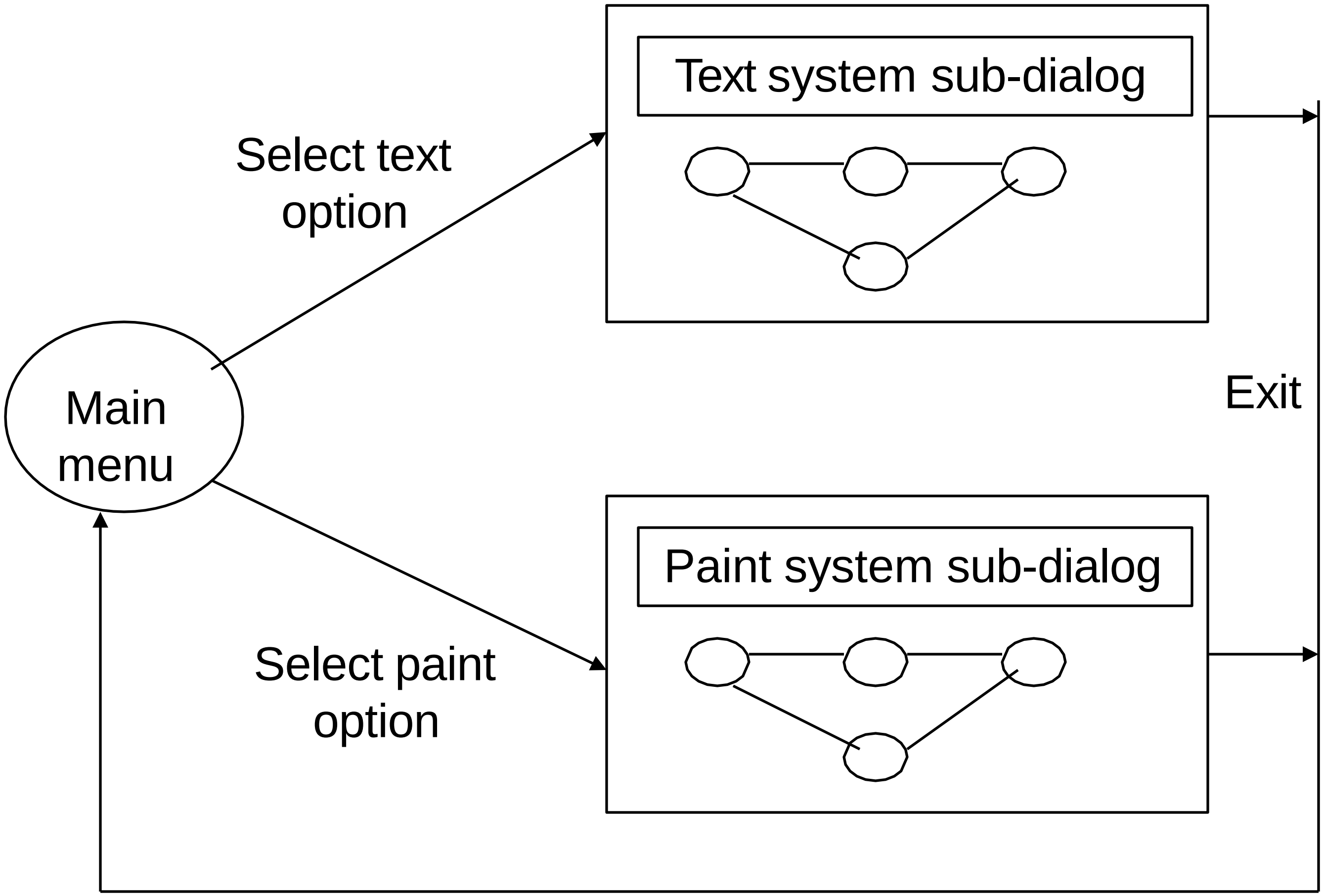
- Hierarchical STNs provide a way to manage complex dialogs
- Here, we divide the dialog into sub-dialogs
- Each sub-dialog is modeled with STNs
- Upper level STNs are designed to connect sub-dialogs

Hierarchical STN - Example

Suppose we want to model the dialog for a menu-based system. There are two menu items, one for a text system and the other for a paint-like system.

Each of these systems has its own dialog. For example, the paint system may have the dialog shown in the previous example. We can model the overall dialog as a hierarchical STN, as shown in the next slide

Hierarchical STN - Example

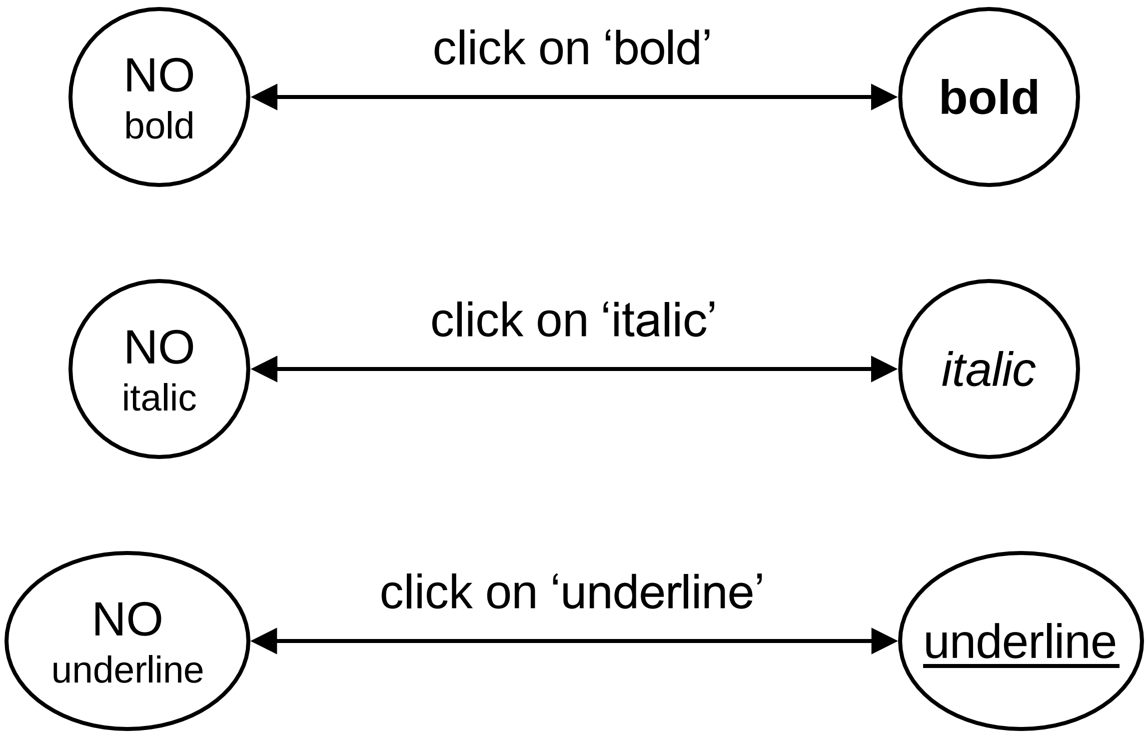


How to Model Complex Dialogs

- However, even hierarchical STNs are inadequate to model many “common” dialogs

  For example, consider a text editor that supports three operations: <u>underline</u>, **bold** and *italic*. Let us try to model this dialog with an STN, assuming first that we can perform (only) one operation on a piece of text

Modelling Complex Dialogs

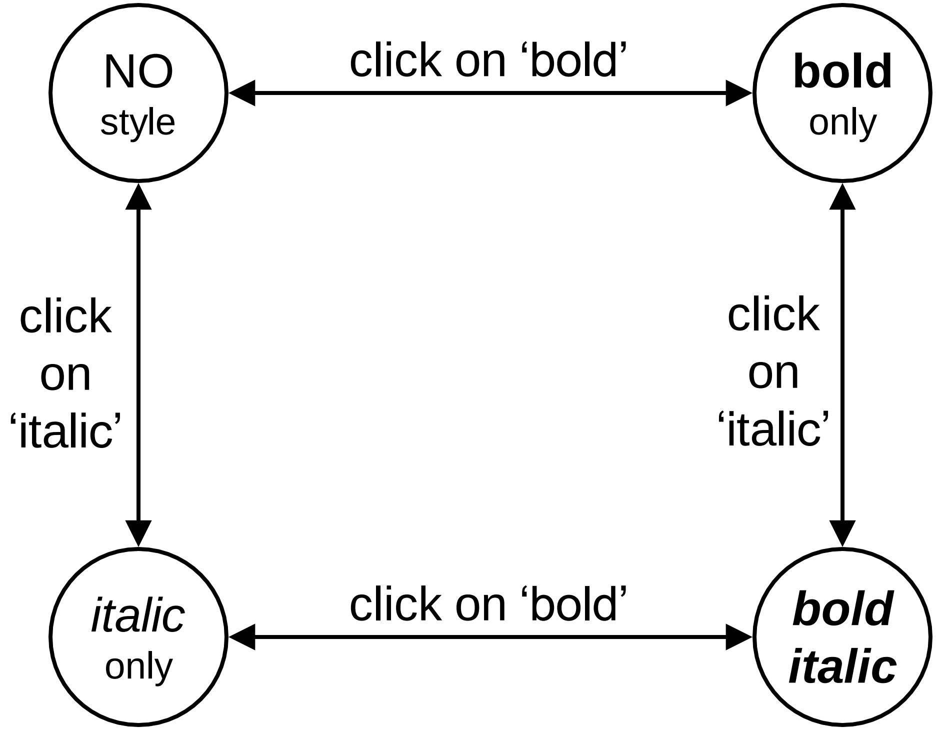


How to Model Complex Dialogs

Now suppose we relax the condition "we can perform (only) one operation on a piece of text". Now we can perform two operations together on the same piece of text (e.g., **bold** followed by *italic*).
How the STN for this new system looks?

(let us construct the STN for only the dialog involving **bold** and *italic*. STN for other pairs will be similar)

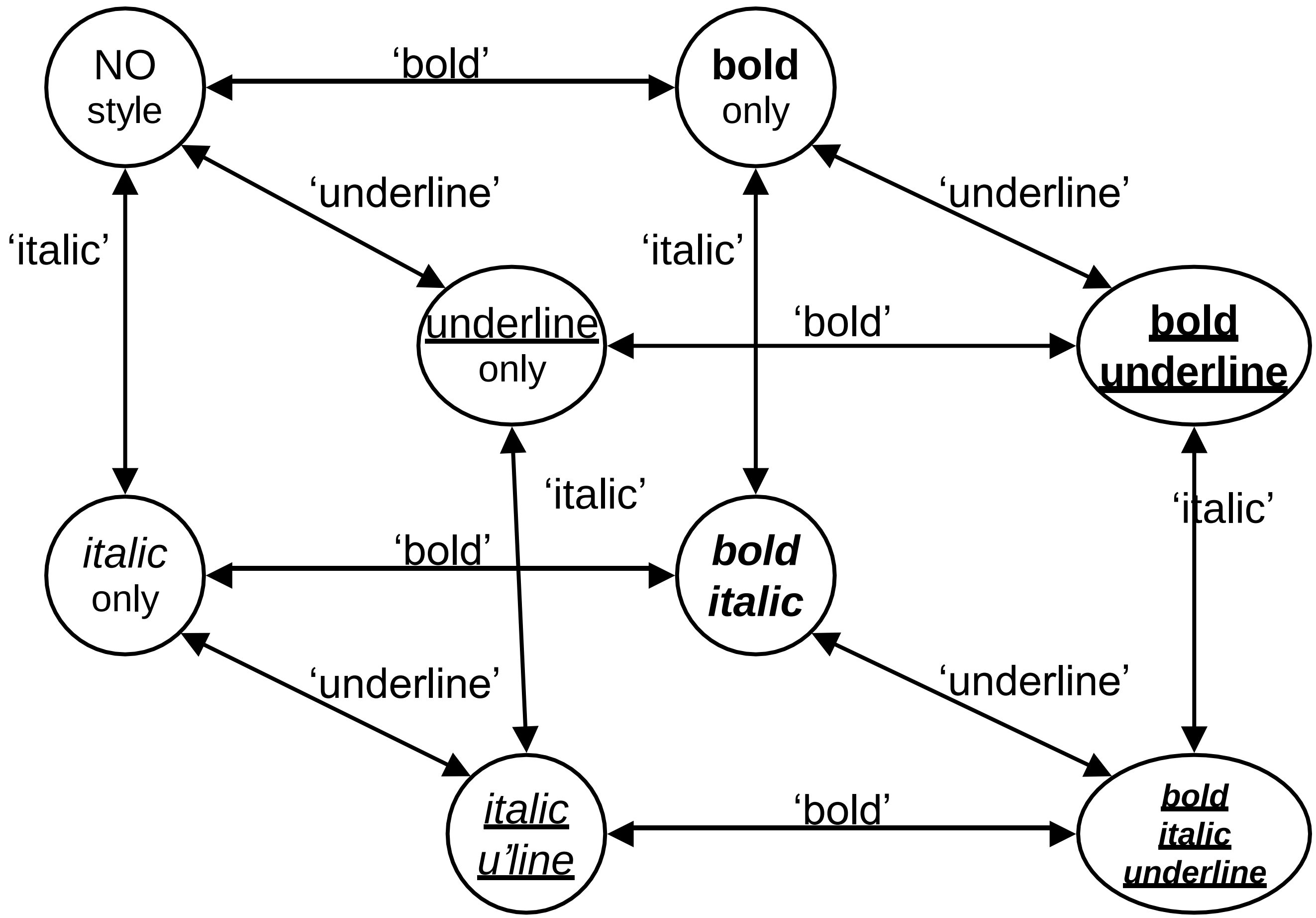
Modelling Complex Dialogs



How to Model Complex Dialogs

Now suppose we relax the condition further. Now we can perform all the three operations together on the same piece of text. This is a fairly common scenario and supported by all text editors. Let us see how the STN for this new system looks.

Modelling Complex Dialogs



How to Model Complex Dialogs

- As we can see, the STN has become very complex with too many states and arcs
- This is because we are trying to model activities that occur on the same object or at the same time. Such behaviors are known as “concurrent behaviors”
- STNs are not very good at modeling concurrent behaviors, which are fairly common in dialogs that we encounter in HCI

Summary

- To better model “concurrent” dialogs, other formalisms are used
- We shall discuss two of those formalisms, namely the State-Charts and the Petri Nets, in the following lectures

HCI: Dialog Design (State-Charts)

Learning Objective

- In the previous lecture, we introduced the need for dialog design
- We also learned about the advantages about formal modeling of dialogs
- We discussed how to use STNs for the purpose

Learning Objective

- As we mentioned, STNs are good for modeling simple systems; for complex systems as well as systems having concurrency, STNs fail
- In this lecture, we shall learn about the State-Chart formalism that can overcome the problems with STNs

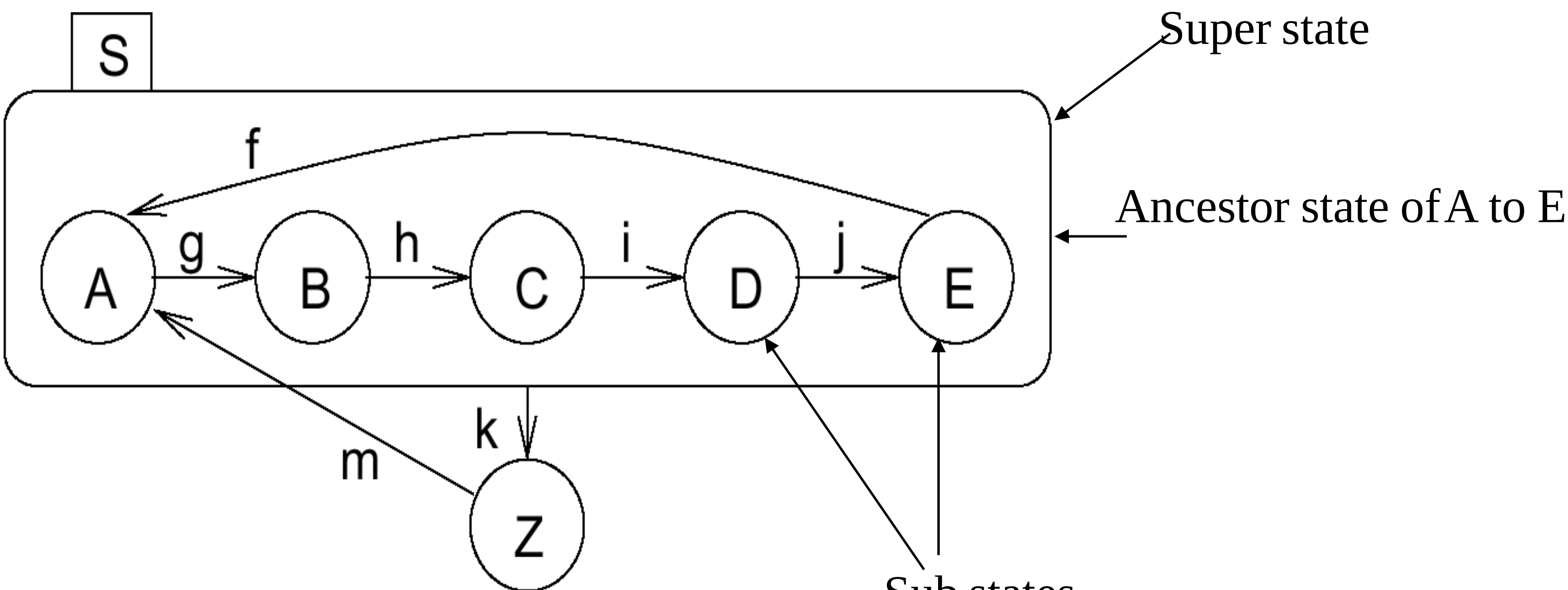
State-Charts

- Proposed by David Harel (1987) to represent complex *reactive* systems
- Extends finite state machines (FSM)
  - Better handle concurrency
  - Adds memory, conditional statements to FSM
- Simplifies complex system representation (states and arcs) to a great extent

Definitions

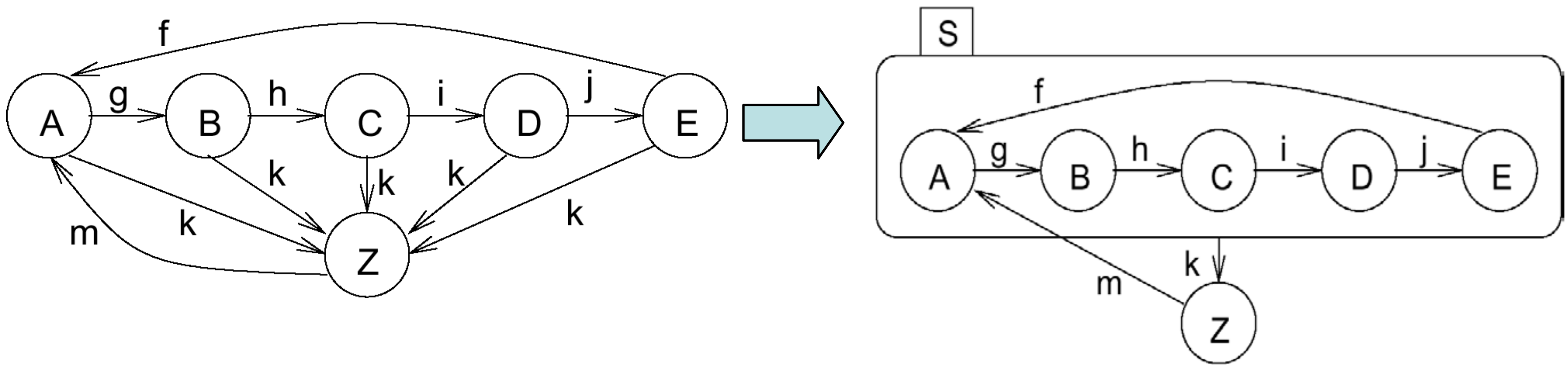
- **Active state**: the current state of the underlying FSM
- **Basic states**: states that are not composed of other states
- **Super states**: states that are composed of other states
  - For each basic state *b*, the super state containing *b* is called the *ancestor* state
  - A super state is called OR super state if exactly one of its sub states is active, whenever it is active

Definitions



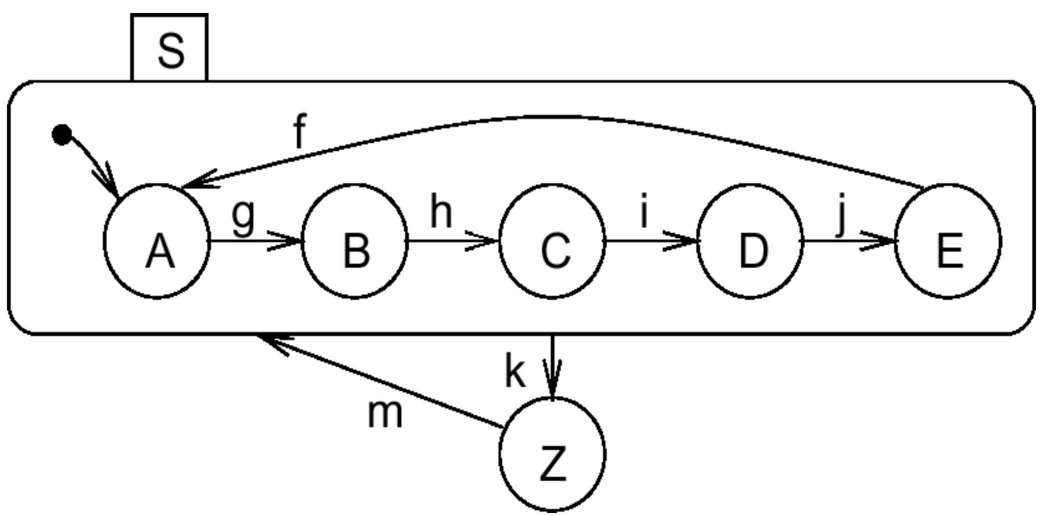
Super State Advantage

- It allows us to represent complex FSM in a nice way, by clustering states



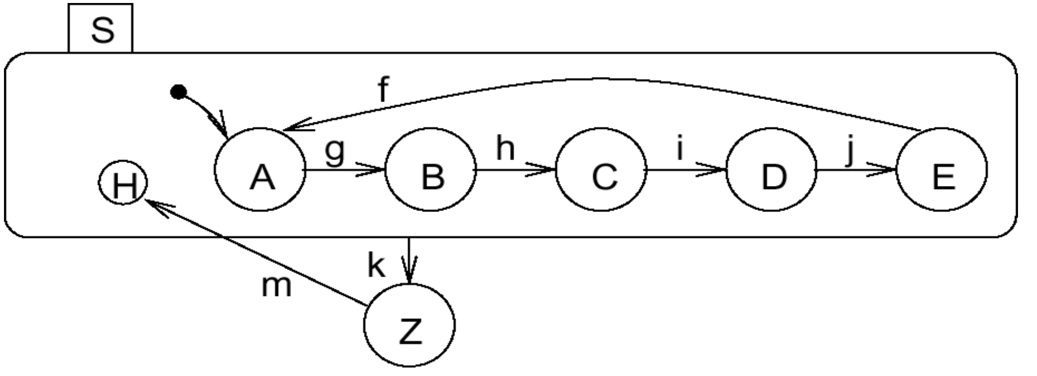
Default State Mechanism

- Indicates the sub state entered whenever super state is entered – represented using a filled circle
  - Not a state by itself

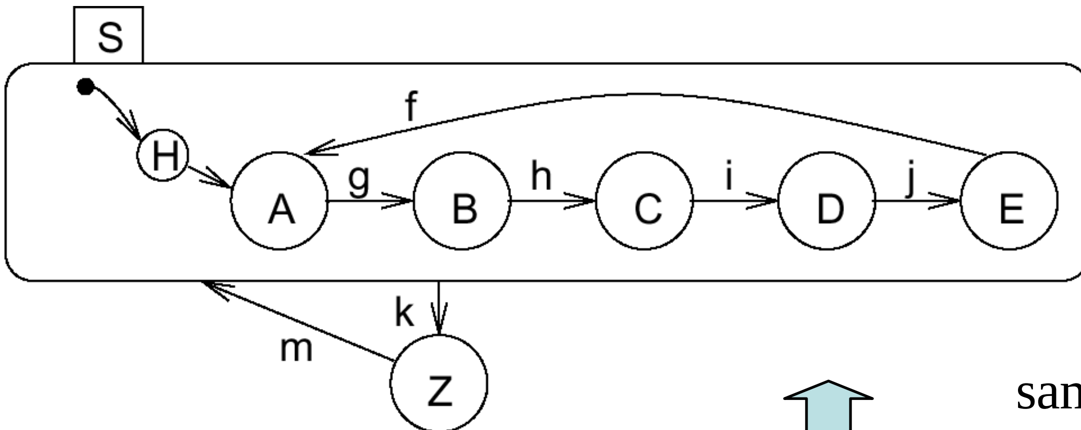


History Mechanism

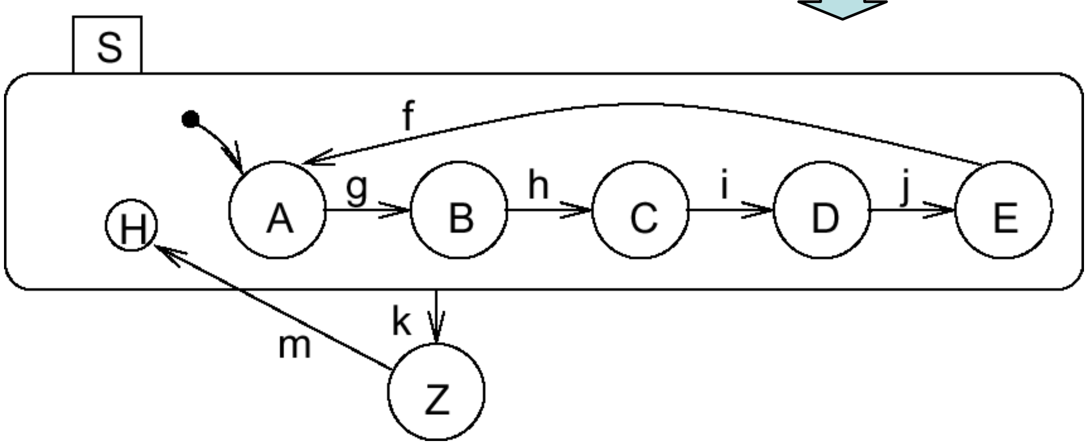
- For input m, S enters the state it was in before S was left
  - If S is entered for the very first time, the default mechanism applies
  - History and default mechanisms can be used hierarchically



Combining History and Default State Mechanism

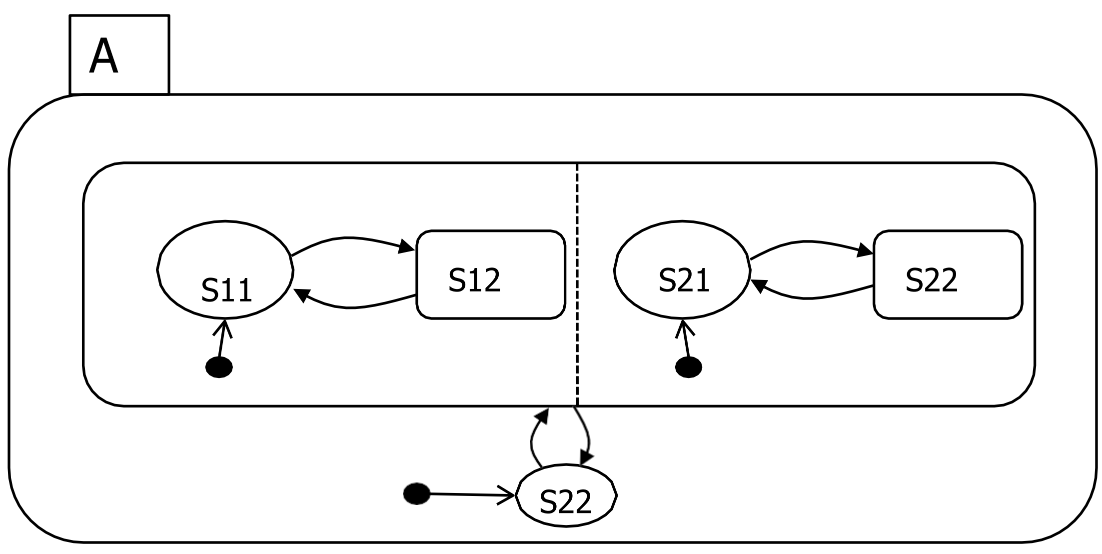


same meaning



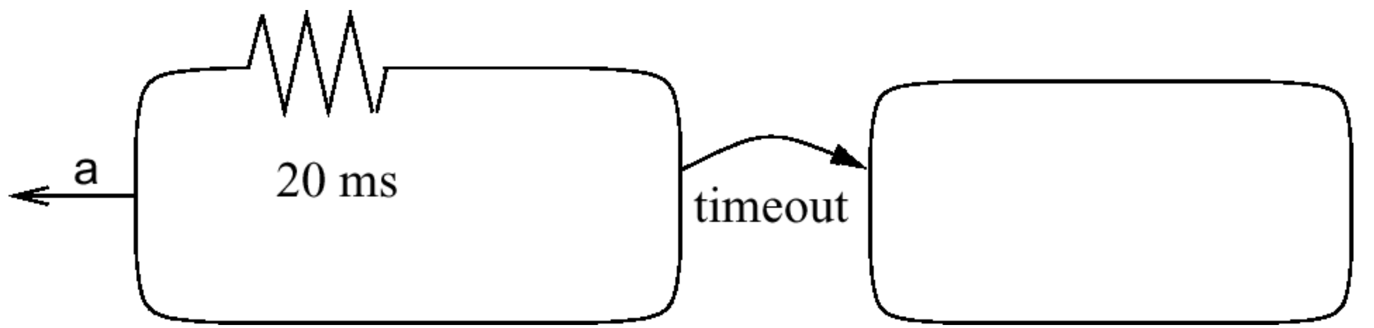
Concurrency

- State-Charts supports concurrency using the notion of the AND super states
  - In AND super states, the FSM is active in all (immediate) sub states simultaneously



Timing Constraints

- State-Chart supports delay/timeout modeling
  - using special edges
  - Do we need it??



If event **a** does not happen while the system is in the left state for 20 ms, a timeout will take place.

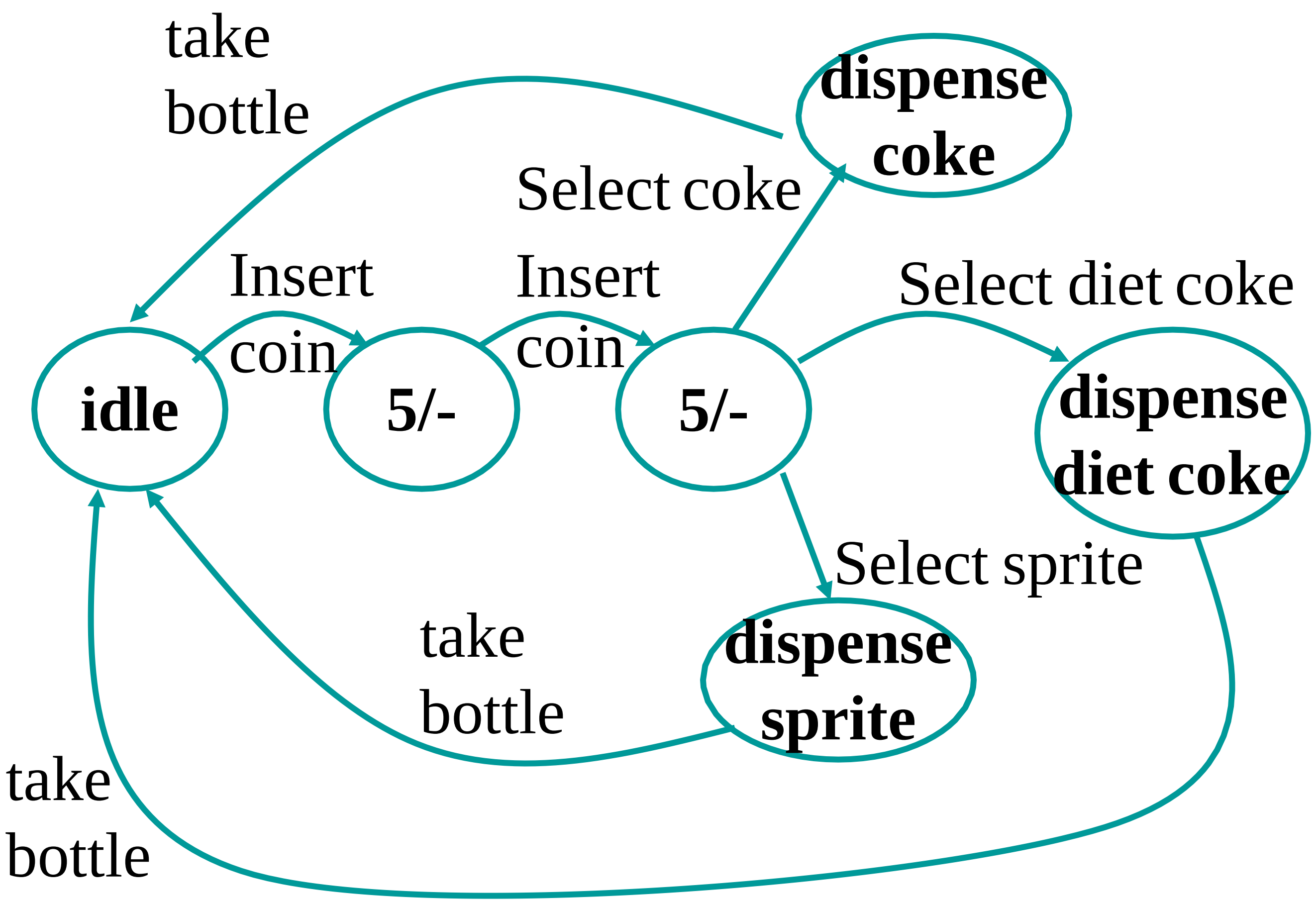
Example: Vending Machine v1.0

- Suppose we have a juice/beverages vending machine:
  - When turned on, this vending machine waits for money
  - When Rs. 5/- coin is deposited, the machine waits for another Rs. 5/- coin
  - When the second coin is deposited, the machine waits for a selection
  - When the user presses “COKE,” a coke is dispensed

Example: Vending Machine v1.0

- Suppose we have a juice/beverages vending machine:
  - When the user takes the bottle, the machine waits again
  - When the user presses either “SPRITE” or “DIET COKE,” a Sprite or a Diet Coke is dispensed
  - When the user takes the bottle, the machine waits again
- Let us represent this behavior using FSM

Vending Machine v1.0



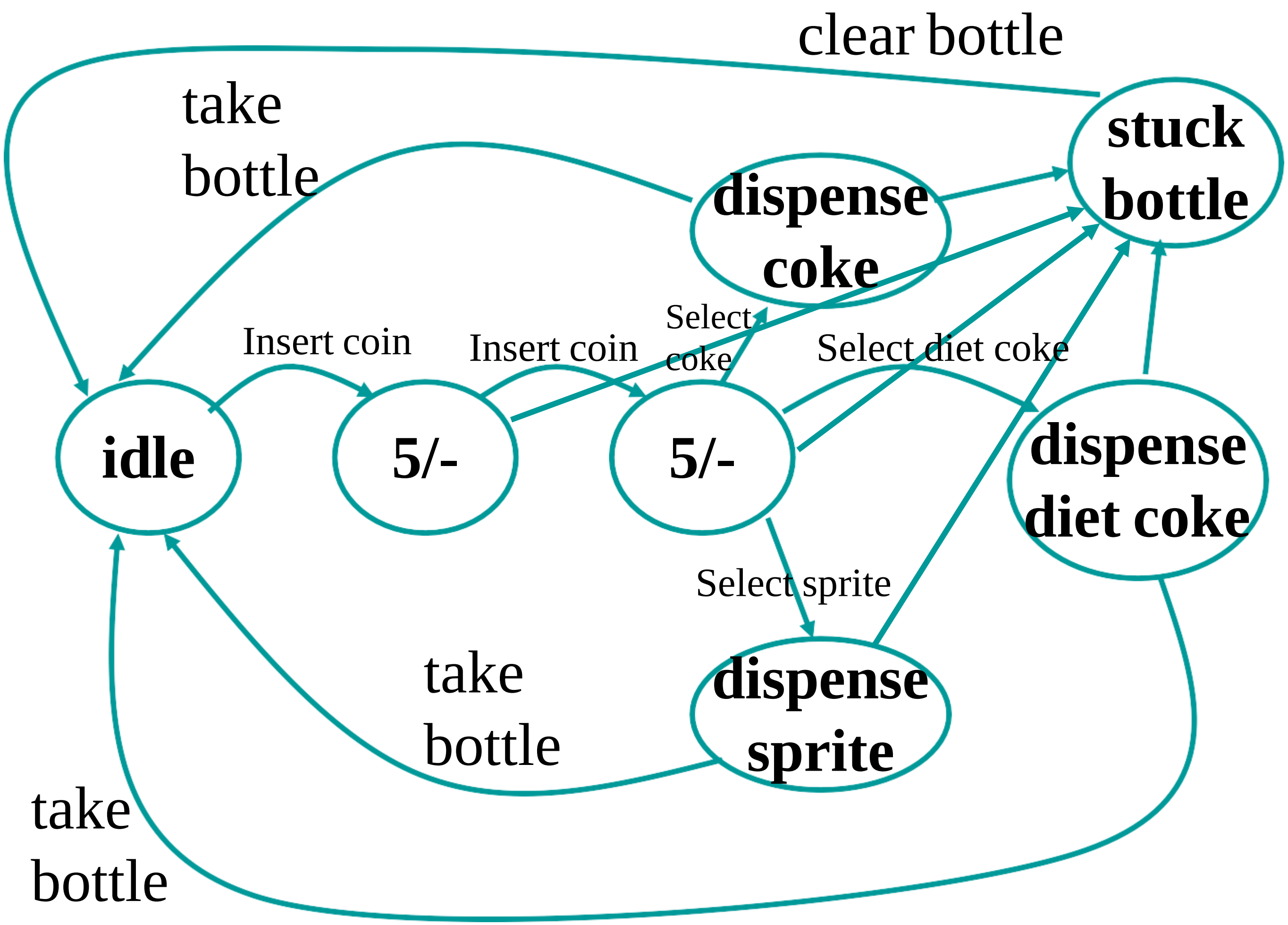
Vending Machine v2.0

- Let us include some more features in the vending machine
- The Bottles can get stuck in the machine
  - An automatic indicator will notify the system when a bottle is stuck
  - When this occurs, the machine will not accept any money or issue any bottles until the bottle is cleared
  - When the bottle is cleared, the machine will wait for money again

Vending Machine v2.0

- State machine changes
  - How many new states are required?
  - How many new transitions?

Vending Machine v2.0



Vending Machine v3.0

- Let us add some more features in the vending machine
- Bottles sometimes shake loose
  - An additional, automatic indicator will indicate that the bottle is cleared
  - When the bottles are cleared, the machine will return to the same state it was in before the bottle got stuck

Vending Machine v3.0

- State machine changes
  - How many new states are required?
  - How many new transitions?

Vending Machine v4.0

- We can add even more features
- Automatic bottle filler
  - If a button is pressed, the machine will toggle between bottle filling and dispensing modes
  - When in bottle filling mode
    - Bottles may be inserted if the Coke/Diet Coke/Sprite machine is ready
    - When a bottle is inserted, the machine will NOT be ready to accept another bottle and will check the bottle

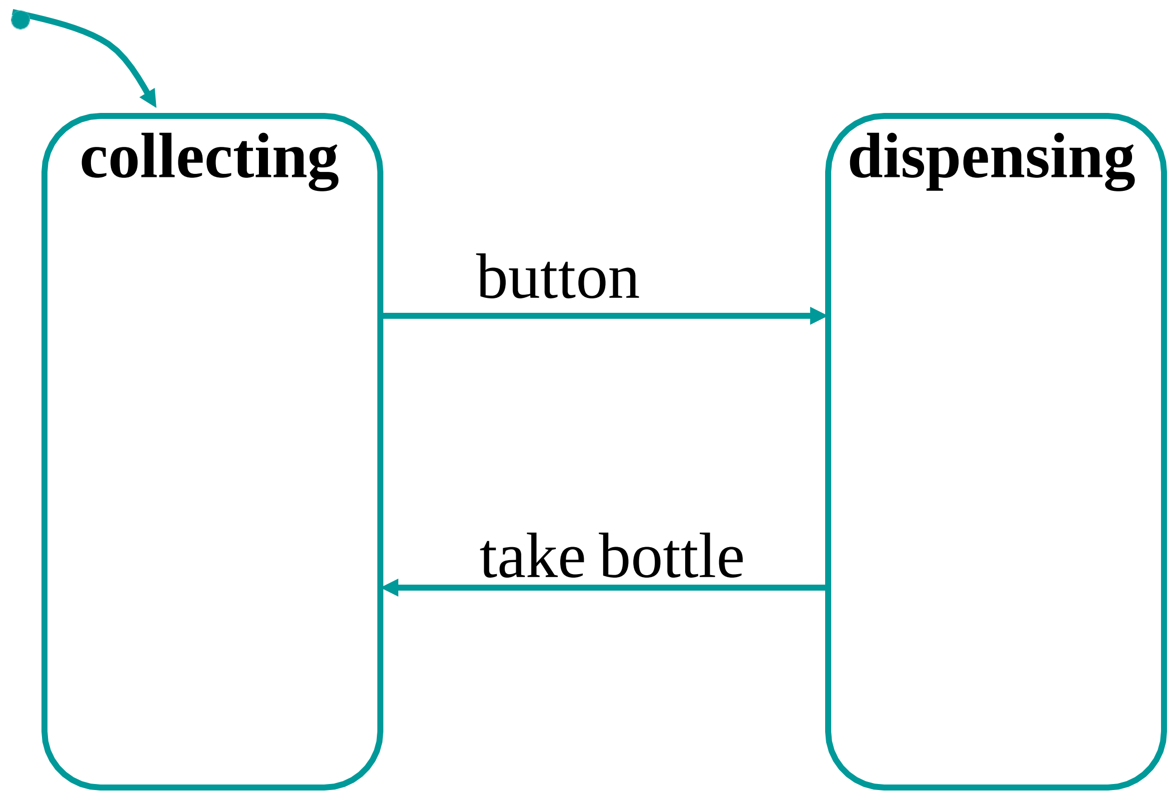
Vending Machine v4.0

- We can add even more features
- Automatic bottle filler
  - If a button is pressed, the machine will toggle between bottle filling and dispensing modes
  - When in bottle filling mode
    - If the bottle check finds a Coke was inserted, it will signal Coke_OK and return to ready
    - If the bottle check finds a Diet Coke was inserted, the coke machine will signal Diet_OK and return to ready
    - Otherwise, the bottle will be immediately dispensed

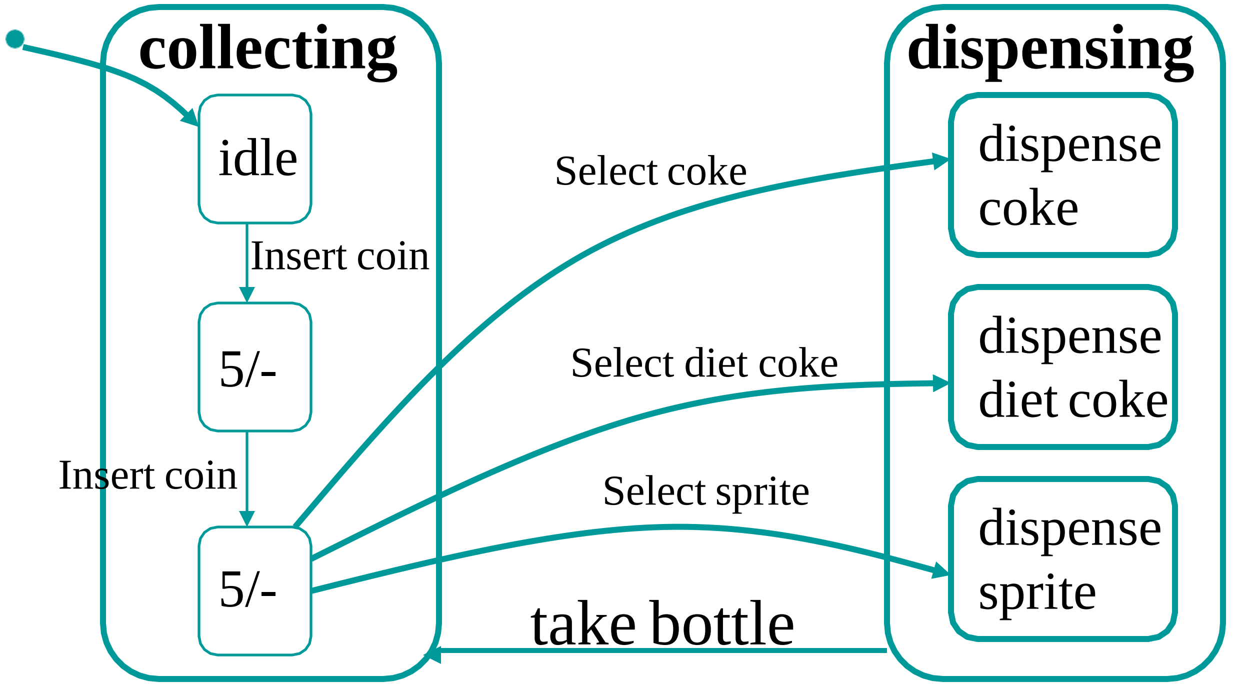
Vending Machine v4.0

- State machine changes
  - How many new states are required?
  - How many new transitions?

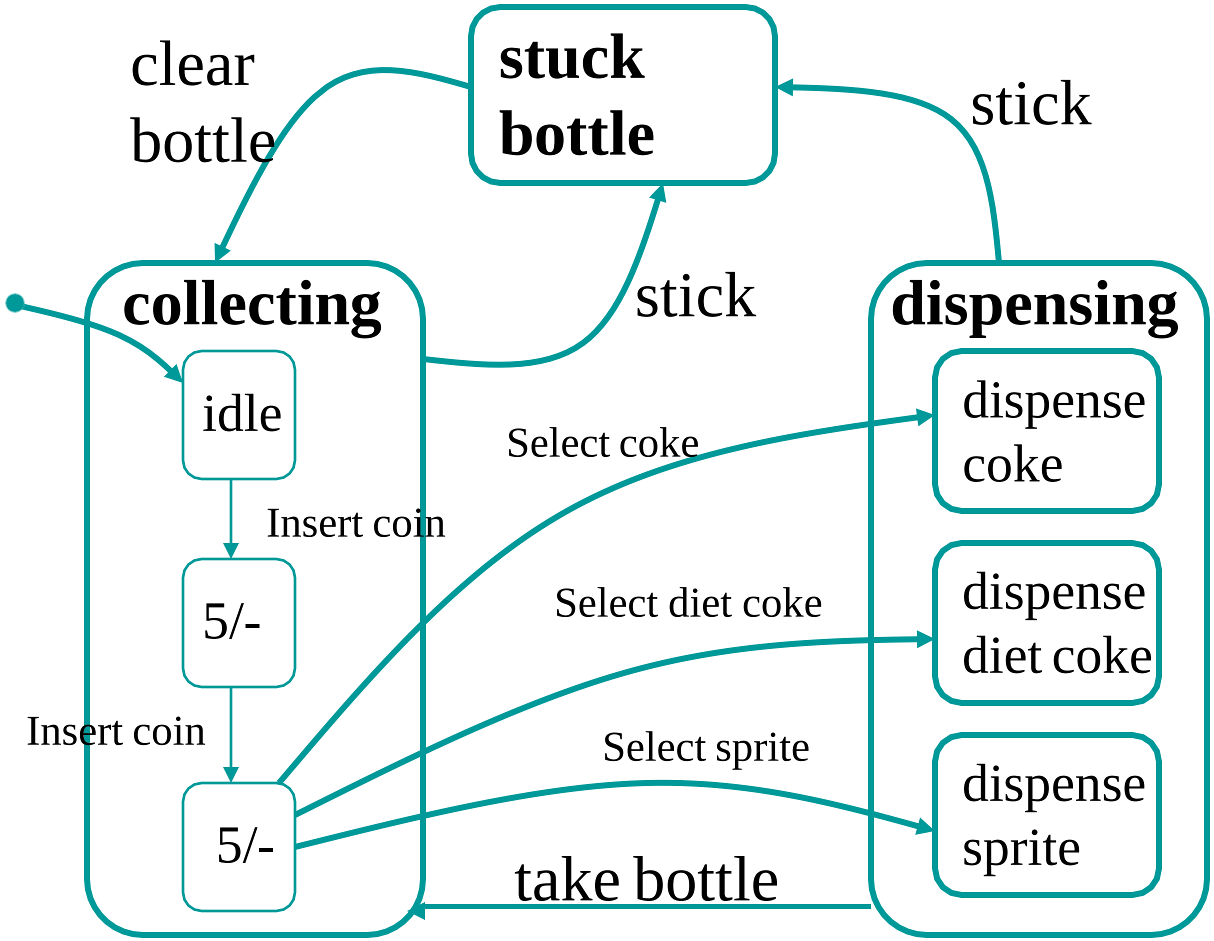
State-Chart Construction: Bottle Dispenser



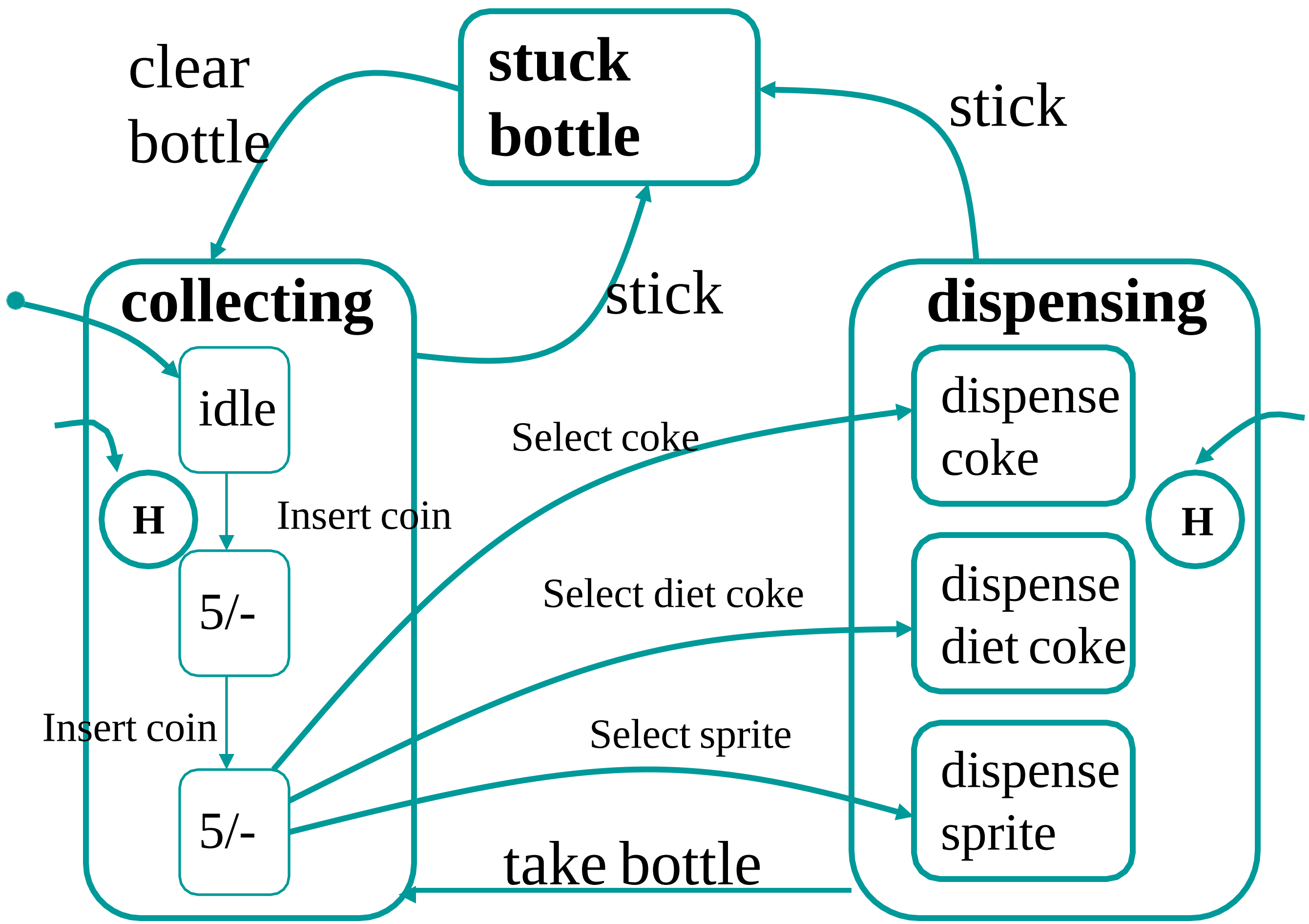
State-Chart Construction: Bottle Dispenser



State-Chart Construction: Bottle Dispenser



State-Chart Construction: Adding History



State-Chart Pros

- Large number of commercial simulation tools available (StateMate, StateFlow, BetterState, ...)
- Available “back-ends” translate State-Charts into C or VerilogHDL or VHDL, thus enabling software or hardware implementations

HCI: Dialog Design Using Petri Nets

Learning Objective

- In the previous lecture, we discussed about State-Charts, a formalism suitable for dialog design, which is potentially more expressive than State Transition Networks (STNs)
- In this lecture, we shall discuss a powerful formalism for dialog design, namely the (classical) Petri Nets

(Classical) Petri Net (PN)

- The formalism was first proposed by Carl Adam Petri (1962, PhD thesis)

- It is a simple model of dynamic behavior
  - Just four elements are used to represent behavior: **places**, **transitions, arcs and tokens**
  - Graphical and mathematical description for easy understanding
  - Formal semantics allow for analysis of the behavior

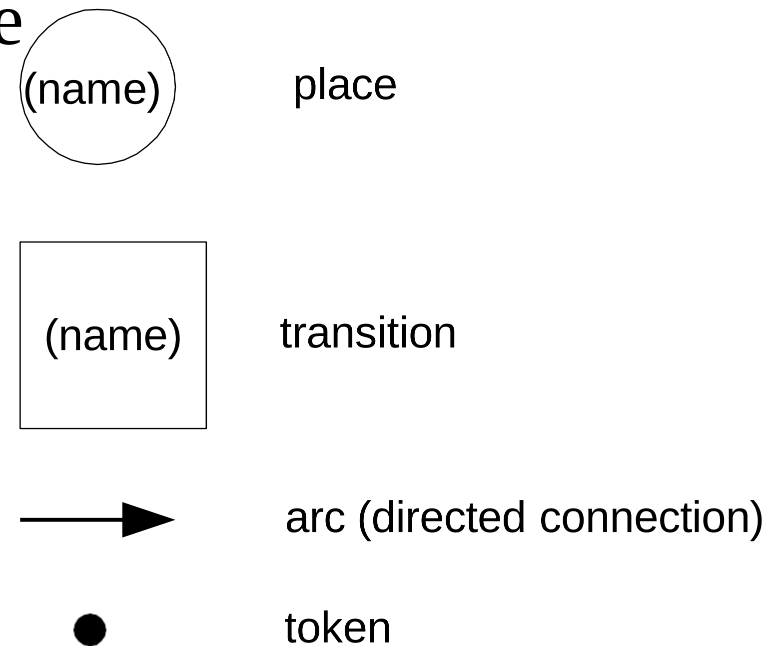
Elements of PN

- Place: used to represent passive elements of the reactive system
- Transition: used to represent active elements of the reactive system
- Arc: used to represent causal relations
- Token: elements subject to change

The state (space) of a process/system is modeled by places and tokens and state transitions are modeled by transitions

Elements of PN: Notation

- A place is represented by a circle
- Transitions are represented by squares/rectangles
- Arcs are represented by arrows
- Tokens are represented by small filled circles



Role of a Token

- Tokens can play the following roles
  - A **physical object**, for example a product, a part, a drug, a person
  - An **information object**, for example a message, a signal, a report
  - A **collection of objects**, for example a truck with products, a warehouse with parts, or an address file
  - An **indicator of a state**, for example the indicator of the state in which a process is, or the state of an object
  - An **indicator of a condition**: the presence of a token indicates whether a certain condition is fulfilled

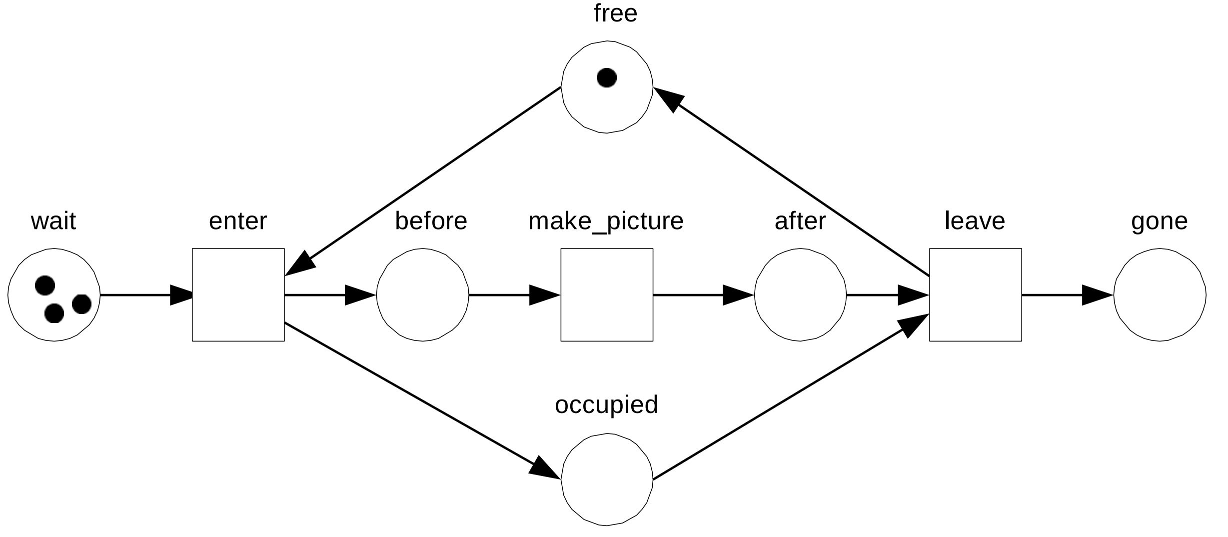
Role of a Place

- A place in a PN can represent the following
  - A type of **communication medium**, like a telephone line, a middleman, or a communication network
  - A **buffer**: for example, a depot, a queue or a post bin
  - A **geographical location**, like a place in a warehouse, office or hospital
  - A possible **state or state condition**: for example, the floor where an elevator is, or the condition that a specialist is available

Role of a Transition

- A transition can be used to represent things such as
  - An **event** (e.g., starting an operation, the switching of a traffic light from red to green)
  - A **transformation of an object**, like adapting a product, updating a database, or updating a document
  - A **transport of an object**: for example, transporting goods, or sending a file

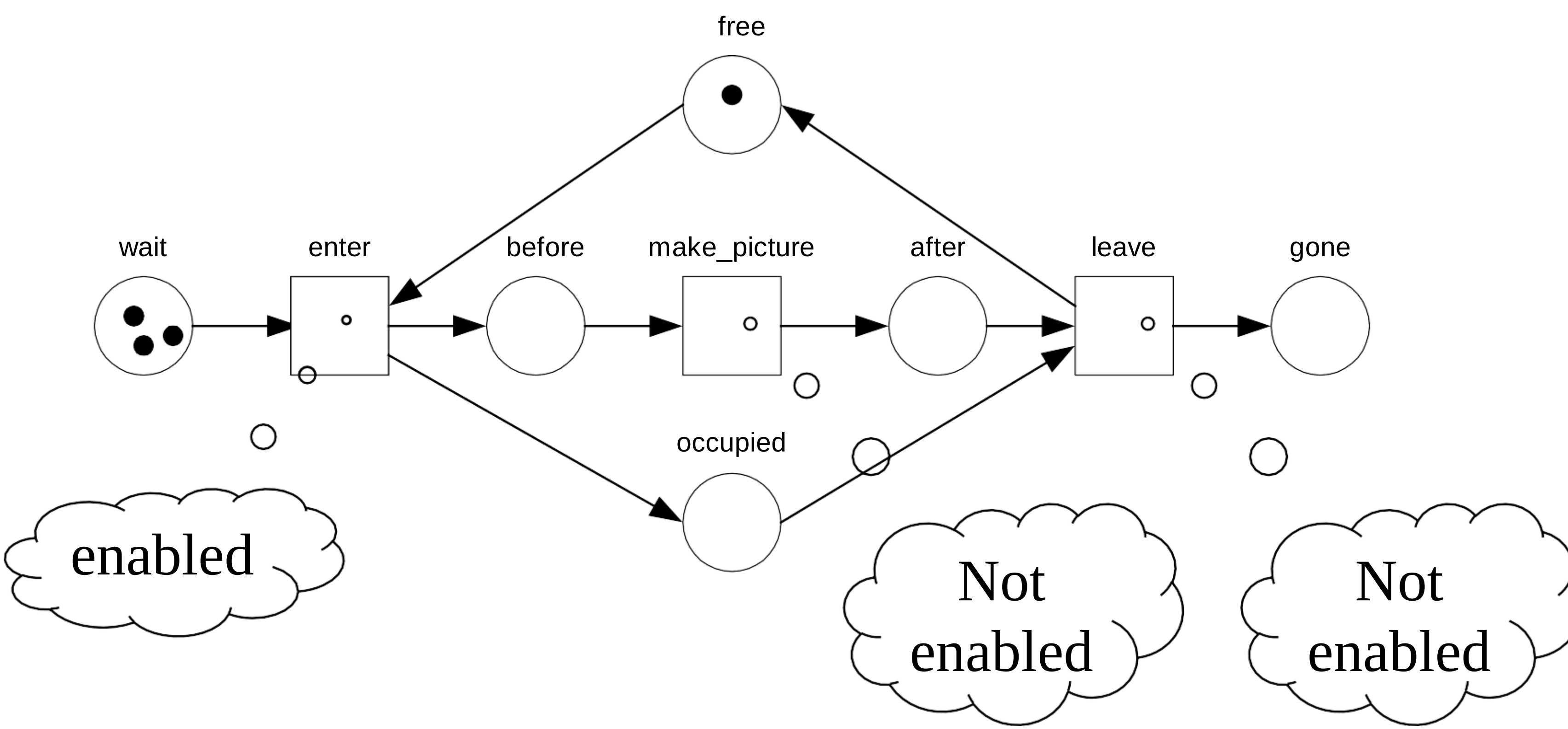
PN Construction Rules



- Connections are directed
- No connections between two places or two transitions is allowed
- Places may hold zero or more tokens

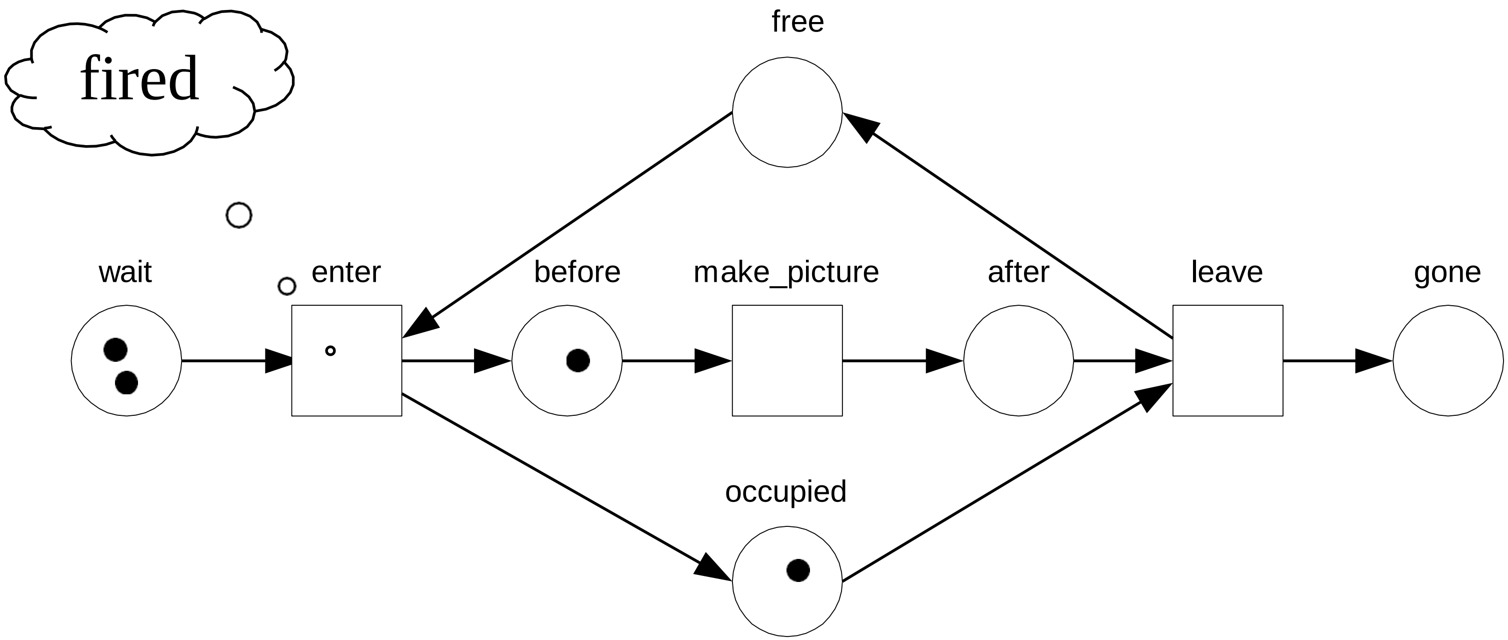
Enabled

- A transition is **enabled** if each of its input places contains at least one token



Firing

- An **enabled** transition can **fire** (i.e., it occurs)
- When it **fires** it **consumes** a token from each input place and **produces** a token for each output place

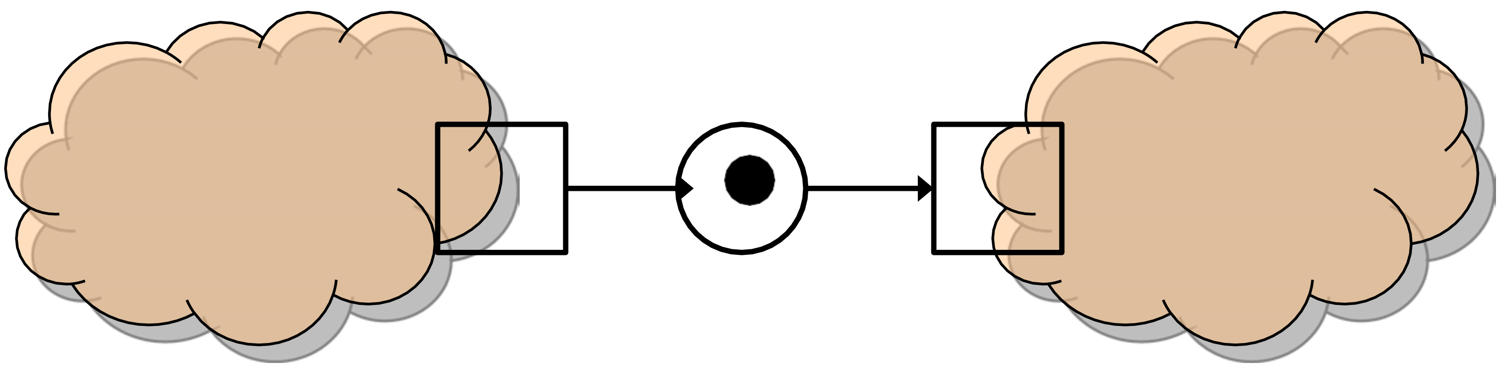


Remarks

- Firing is **atomic** (i.e., it always completes after start)
- Non-determinism: multiple transitions may be enabled, but only one fires at a time
- The **state** of the reactive system is represented by the distribution of tokens over places (also referred to as **marking**)

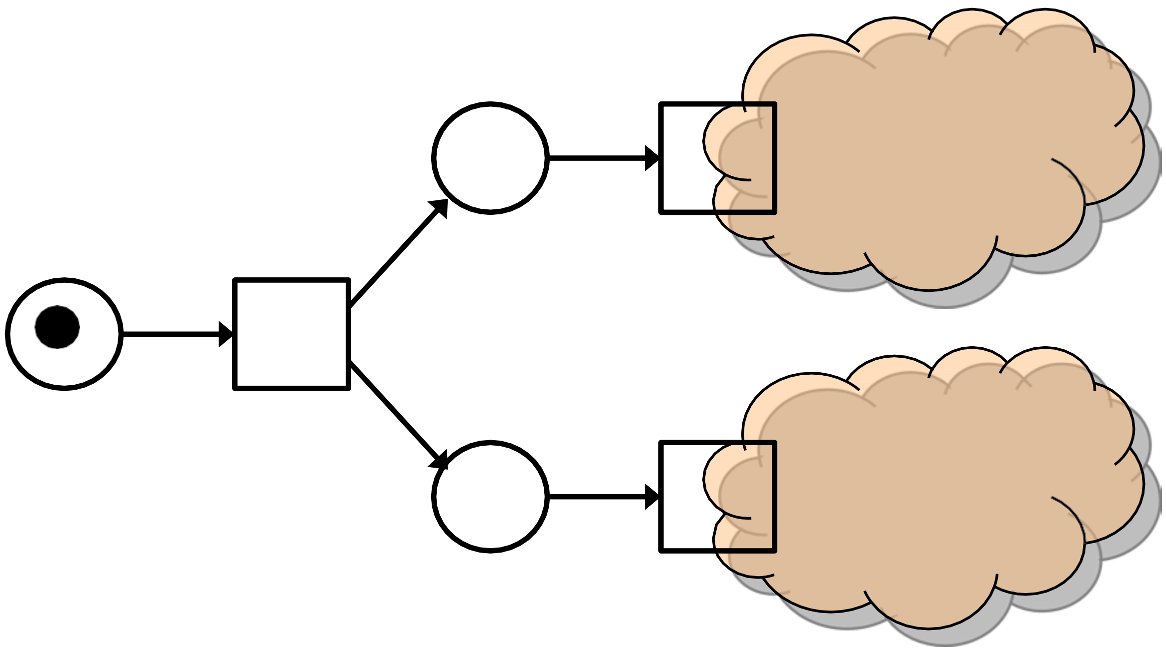
Typical PN Structures

- **Causality**, i.e., one part of the PN is caused by the other part



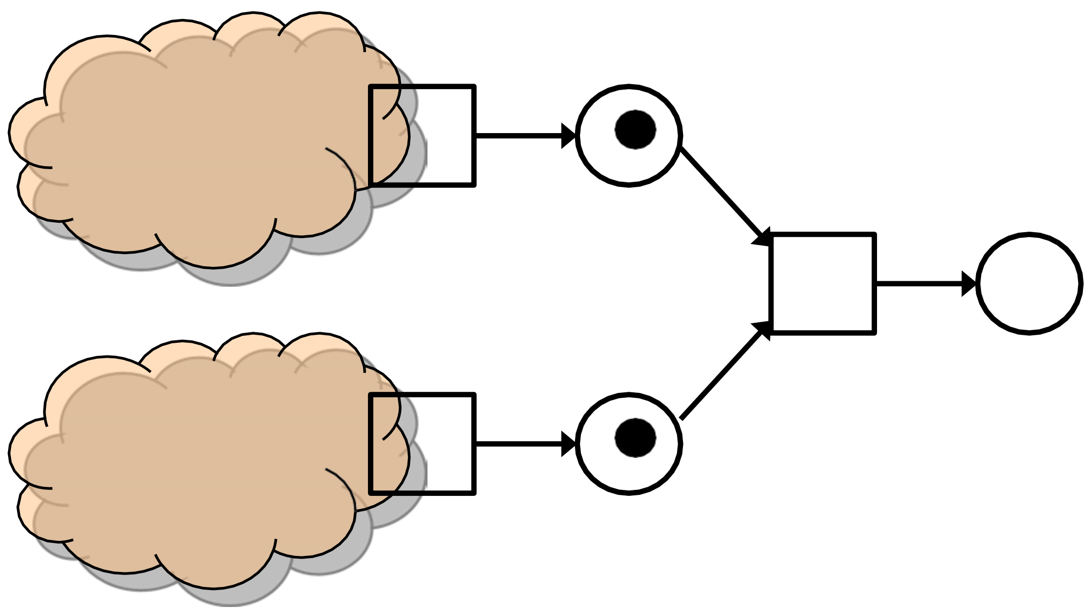
Typical PN Structures

- **Parallelism (AND-split)**, i.e., two parts of the PN can be activated at the same time



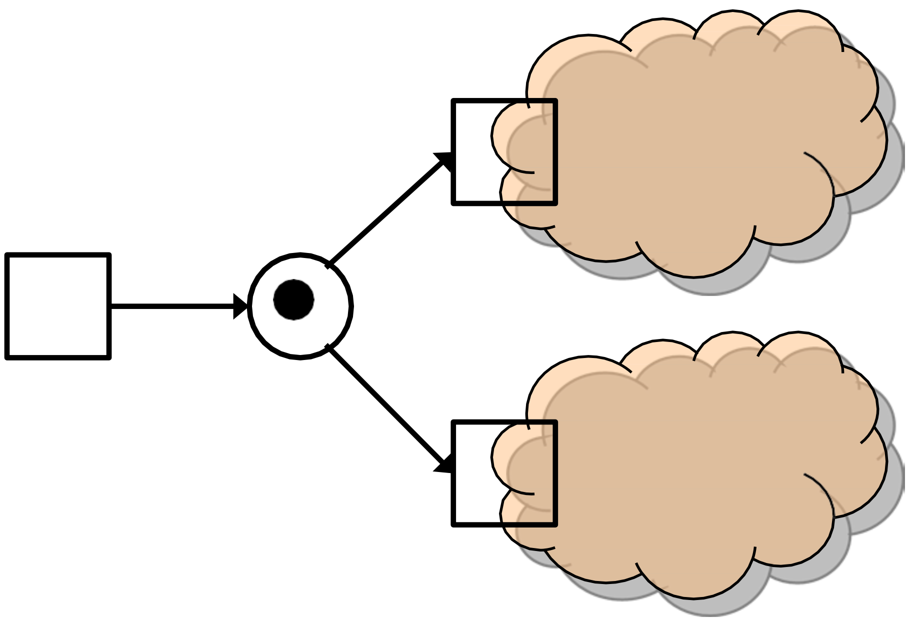
Typical PN Structures

- **Parallelism (AND-join)**, i.e., two parts of the PN must be active at the same time, or enable further firings



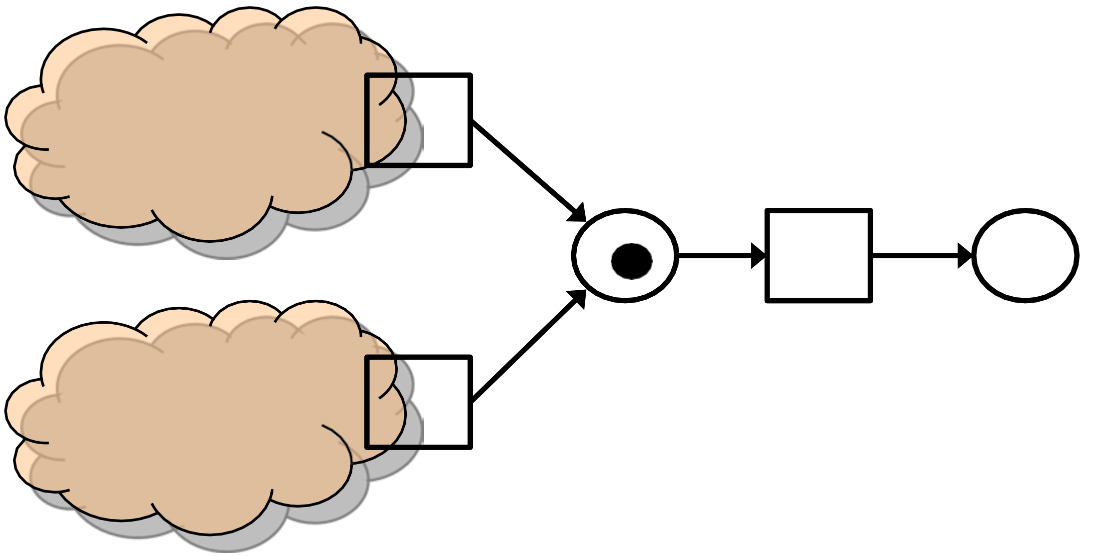
Typical PN Structures

- **Choice (XOR-split)**, i.e., either of the two sub nets of a PN can be activated



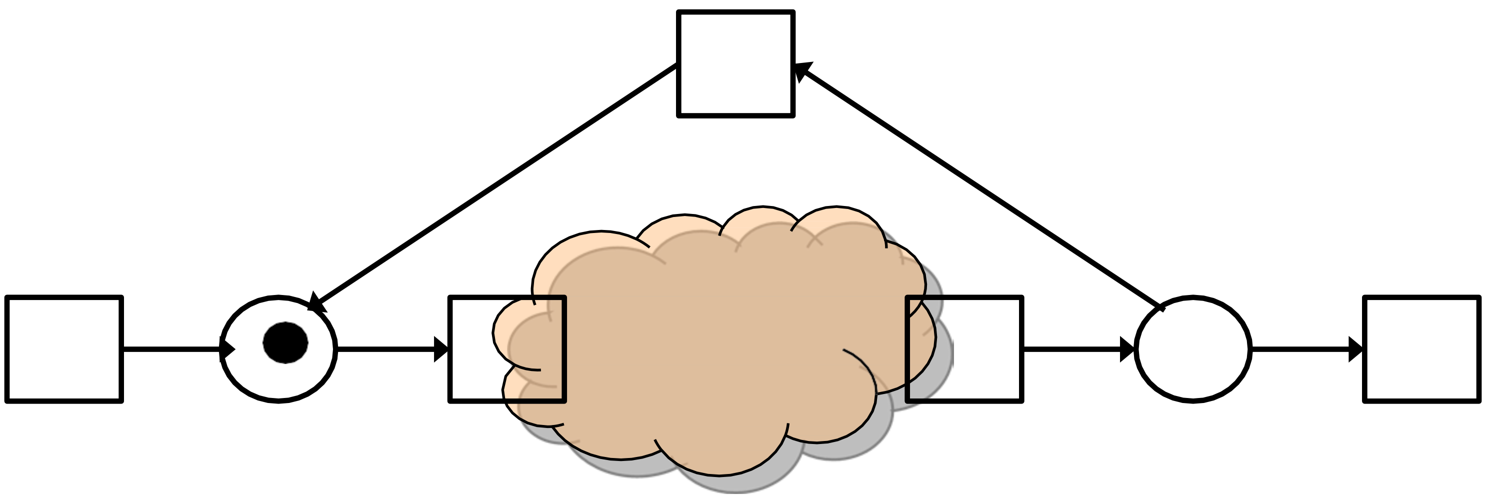
Typical PN Structures

- **Choice (XOR-join)**, i.e., either of the two sub nets of a PN is an enabler



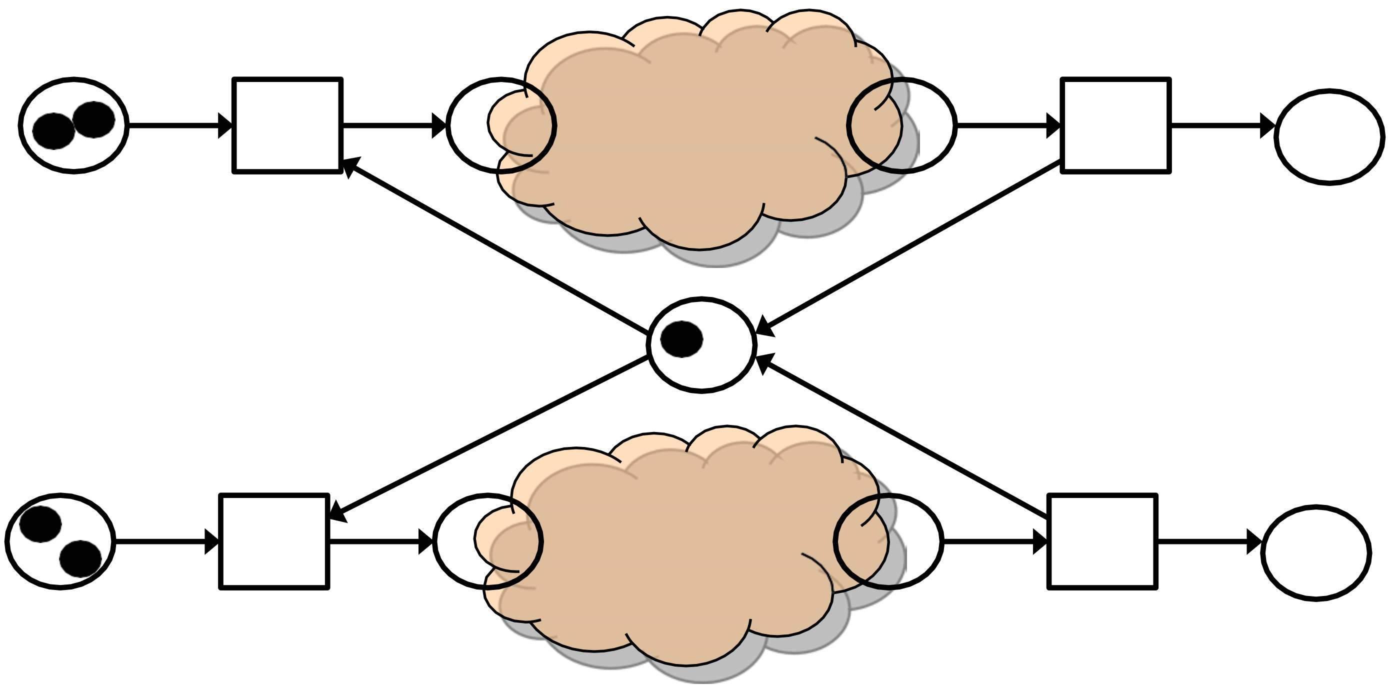
Typical PN Structures

- **Iteration (1 or more times)**, i.e., the firing iterates at least once



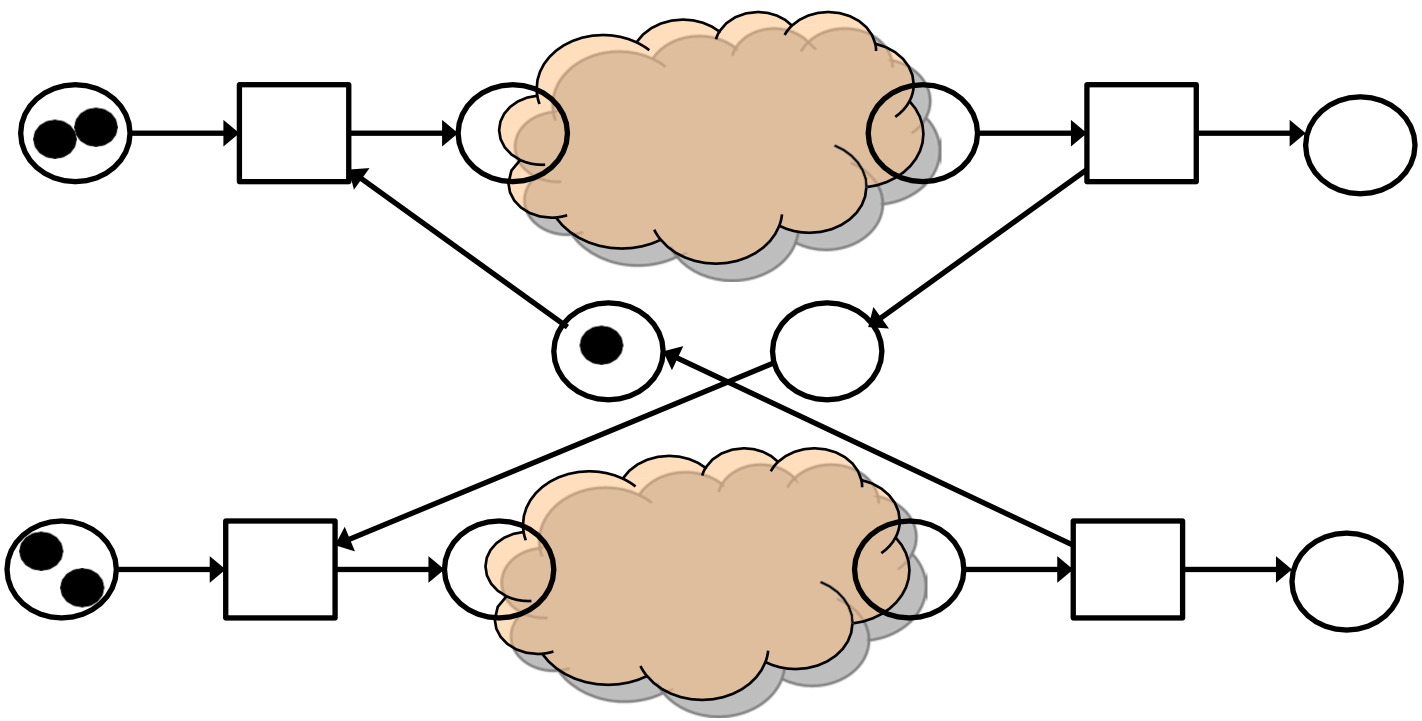
Typical PN Structures

- **Mutual exclusion**, i.e., only one of the sub nets should be active at a time



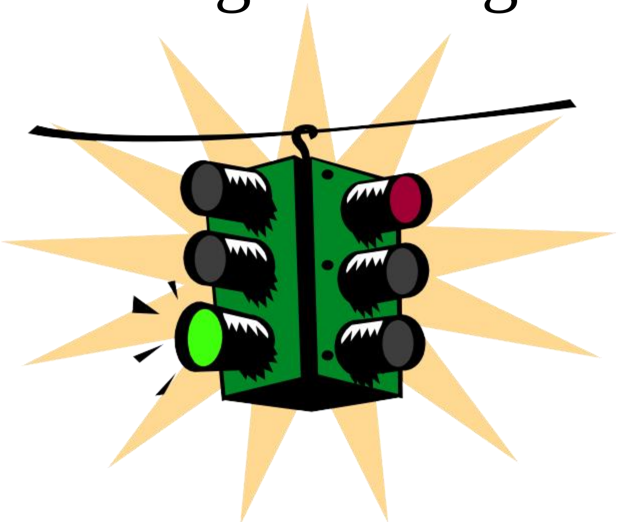
Typical PN Structures

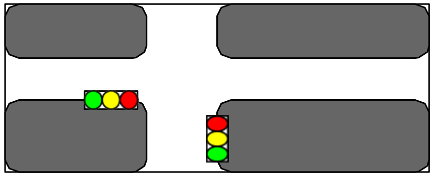
- **Alternating**, i.e., the sub nets of a PN should be alternatively activated



Example: Two Traffic Lights

- Let us illustrate the idea with an example. Suppose there are two traffic lights at a road junction. How we can model the behavior of these two lights using PN?





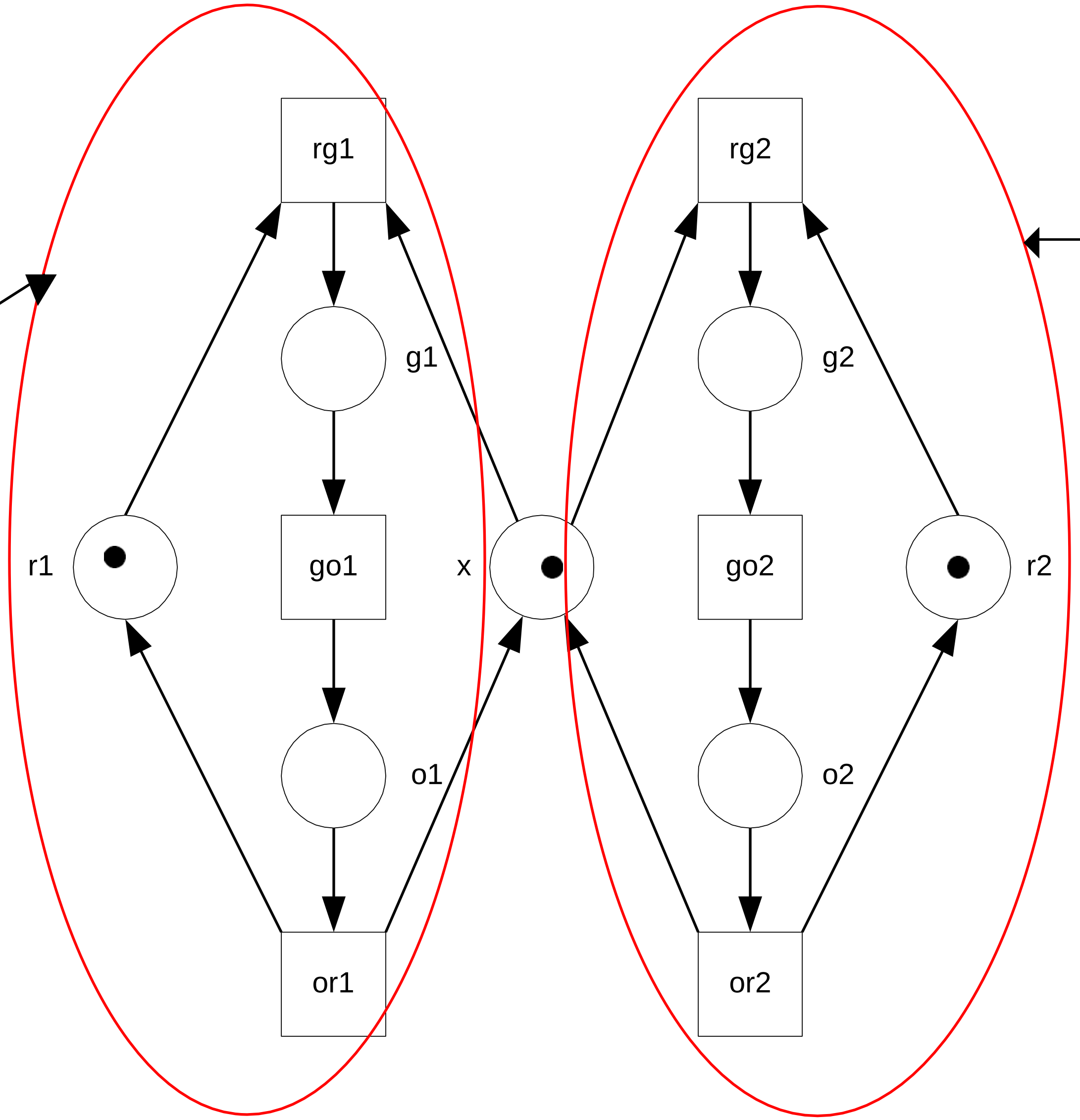
Example: Two Traffic Lights

- The characteristics of the combined system (of two lights)
  - They are mutually exclusive
  - They should alternate
- We can use the typical structures to model the behavior

Example: Two Traffic Lights

One light

(r=red,
g=green,
o=orange)



The other light

(r=red,
g=green,
o=orange)

Summary

- Look closely in the example how the elements of a PN are used to model the behavior of the system
- In the next lecture, we shall discuss with an example the usefulness of formal dialog representation
- Also we shall discuss about the properties we check with the formalisms and how?

HCI: Dialog Design – Use of Formalism

Learning Objective

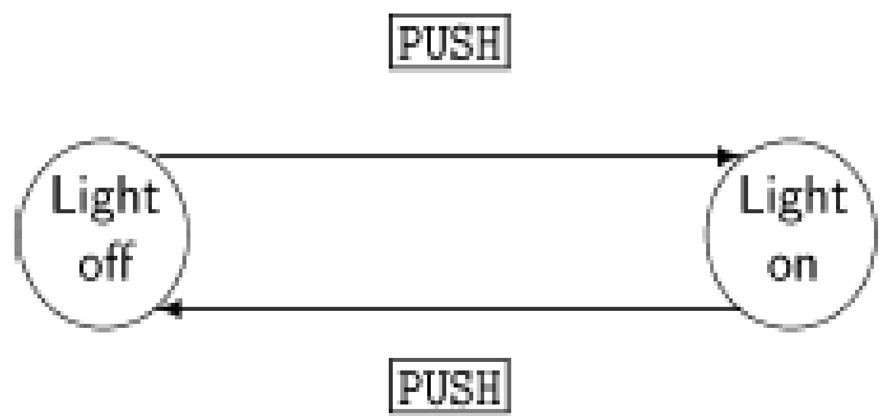
- In the previous lectures, we have learned different formalisms such as State Transition Network (STN), State-Charts (Finite State Machines (FSM)), and the Petri Nets (PN) to represent the dialogs
- Some are simple (STN) but not powerful enough to capture the typical interactive behavior (complexity and concurrency), others are more suitable (FSM, PN)

Learning Objective

- It’s clear from the discussions that representing interactive systems formally is no easy task and requires expertise
- This brings us to the question, why should we spend time and effort mastering formal representation techniques?
- In this lecture, we shall discuss how to answer this question

Example Use of Formalism

- Let us try to understand the use of formalism in dialog design with a simple example. Consider a simple two state system – a light bulb controlled by a push button switch, which alternately turns the light on or off.
- The corresponding STN is



a. Light with push-on/push-off action.

Example Use of Formalism

- The example system belongs to the general class of push button devices
- They belong to an important class of interactive system (Desktop GUI, touch-screen devices, WWW)
  - Ubiquitous (mobile phones, vending machines, aircraft flight deck, medical unit, cars, nuclear power stations)
- Thus, modelling such devices can actually help to model a large number of interactive systems
- Hence, Interaction with such devices can be modelled as the Matrix Algebra (MA)

Push Button Device: FSM to MA

- We shall use the following notations
  - N = number of states
  - States are numbered from 0 to N
  - A transition is represented by a matrix ($N \times N$)
  - A state is represented by an unit vector of size N, with N-1 0s and one 1 at the position corresponding to the state number. e.g., states ON = (1 0) and OFF = (0 1)

Push Button Device: FSM to MA

- We shall use the following notations
  - New state = old state (vector) × transition(s) [a matrix multiplication]

    e.g., one push button action push

$$\boxed{\mathrm{PUSH}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

- We can check several properties of our simple push button device through matrix multiplication

Property Checking: Example

- When light is off, pushing the button puts the light on

$$\begin{aligned} \mathbf{off}\ \boxed{\text{PUSH}} &= \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \end{pmatrix} \\ &= \mathbf{on} \end{aligned}$$

- Similarly, we can show pushing the button when the light is on puts it off

Push Button Device: FSM to MA

- The case for *undo*

$$\boxed{\text{PUSH}}\,\boxed{\text{PUSH}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$
$$= I$$

- Pressing the button twice return the system to the original state (s $\boxed{\text{PUSH}}\,\boxed{\text{PUSH}}$ = s)
  - What about pushing the button thrice, four times,…? Do the calculations and check for yourself

Push Button Device: FSM to MA

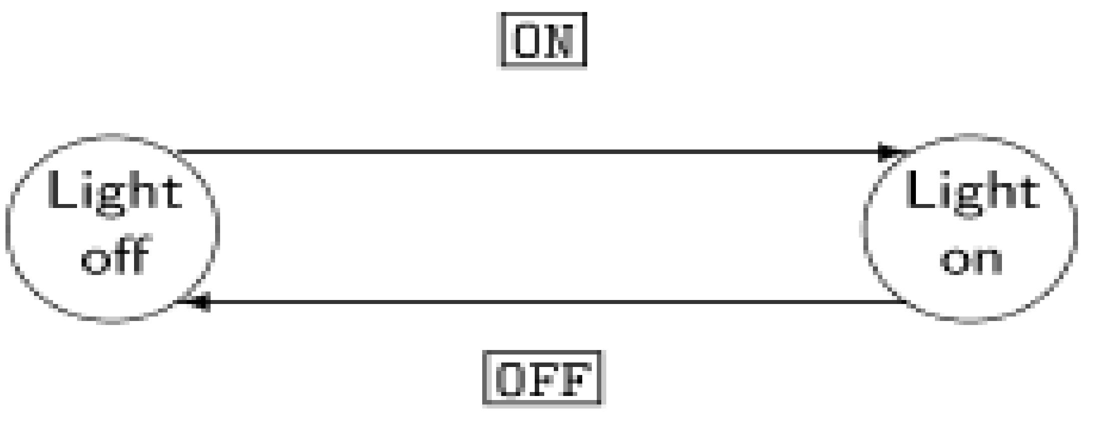
- A problem: the lamp has failed and we want to replace it
  - Need to know if it is safe (i.e. the system is in the off state)

  - How many times the user has to press the button so that he is sure that he is in a safe state (remember, we are not sure of the current state)

$$\boxed{\mathrm{PUSH}}^{px} = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$$

Push Button Device: FSM to MA

- Mathematically, we can show that NO value of n exists for this system that satisfies the equation
  - We need a two-position switch



b. Light with separate on/off actions.

Push Button Device: FSM to MA

- A two-position switch gives two options: ON and OFF

$$[ON] = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$$

$$[OFF] = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$$

- We can check, through matrix algebra, that OFF switch works as intended

$$\mathrm{On}[OFF] = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \end{pmatrix} = [OFF]$$

i.e., pressing **Off** after **On** keeps the system in [OFF] state

$$\mathrm{Off}[OFF] = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \end{pmatrix} = [OFF]$$

i.e., pressing **Off** after **Off** keeps the system in [OFF] state

Motivation to Use Formalism

- From the previous example, we can see that the formal representation (in this case, the matrices) allow us to check for the system properties through formal analysis (in this case, matrix algebra)
- From this analysis, we can decide if there are any usability problems with the system (e.g., single button is not good in case of failure, as it is dangerous to repair)
- That is the main motivation for having formalism in dialog representation (it allows us to check for properties that should be satisfied for having a usable system)

System Properties

- The properties that are checked for a usable system are of two types
  - Action properties
  - State properties

Action Properties

- There are mainly three action properties that a usable system should satisfy
  - Completeness: if all the transition leads to acceptable/final states or there are some *missed arcs*, i.e., some transition sequence don't lead to final states
  - Determinism: if there are several arcs (transitions) for the same action
  - Consistency: whether the same action results in the same effect (state transition) always

How Action Properties Help

- Completeness ensures that a user never gets struck at some state of the dialog, which s/he doesn’t want, and can’t come out from there (leads to frustration and less satisfaction)
- Lack of determinism introduces confusion and affects learnability and memorability
- An inconsistent interface reduces memorability and learnability, thus reducing the overall usability

State Properties

- There are mainly three state properties related to system usability
  - Reachability: can we get to any state from any other state?
  - Reversibility: can we return to the previous state from the current state?
  - Dangerous states: Are there any undesirable states that leads to deadlock (i.e. no further transitions are possible)?

How State Properties Help

- Reachability ensures that all system’s features can be used
- Reversibility ensures that the user can recover from mistakes, thus increasing confidence and satisfaction
- Detection of dangerous states ensures that the user never goes to one, thus avoiding potential usability problems