Usability

Is directly related to the system capacity to allow users attaining their goals through its usage

Principles:

1. easy to **learn** and remember (**learnability**, memorability)
2. easy to **use** (fast and with few errors) (**efficiency**, efficacy)
3. **Satisfaction**
4. **Visibility** (Is the state visible)
5. **Errors** (few and recoverable)

1. Learnability

<u>Cues</u> that communicate the system model:

- **Affordability**: Perceived and actual properties of a thing that determine how the thing could be used
- **Visibility**: Relevant parts of system should be visible
- **Feedback**: Actions should have immediate, visible effects
    - Visual, Audio, Haptic Feedback

<u>Consistency</u> (Similar things should look and act similar. Different things should look different). Internal, External or Metaphorical Consistensy:

- **Speak the user's language**: Use common words, not technological terms. But use domain specific terms where appropriate
- **Metaphors:**
    - <u>Vantagens:</u>
        - Highly learnable when appropriate
        - Hooks into user's existing mental models very easily
    - <u>Perigos:</u>
        - Often hard for designers to find
        - May be deceptive and may be constraining
        - Metaphor is usually broken somewhere
- **Platform Standards**

2. Efficiency

- **Keyboard Shortcuts** - Keyboard commands + Menu accelerators
- **Command Aggregates**
- **Aggregating Questions** - EX: yes to all, no to all…
- **Defaults, Histories and AutoCompletes**

3. Visibility

- **Make Modes Clearly Visible** - EX: Caps lock on and off
  - **Modes**: states in which actions have different meanings
- **Spotlight metaphor** - attention focuses on one input channel (e.g. area of visual field) at a time
- **Visible Navigation State** - Pagination, tabs, etc
- **Visible Model State** - Continuous visual representation of model
  - What to visualize should be guided by the user's tasks. EX: Progress Bar; Number of Pages Word.
- **Feedback:**
  - Actions with immediate visible effects:
    - <u>Low-level</u>: Push Button
    - <u>High-level</u>: New web page start loading
  - Unnecessary Feedback
- **Audio and Haptic Visibility**

4. Errors

- **Error Types**:
  - <u>Slips and Lapses</u> - Failure to correctly execute a procedure
    - **Slips:** failure of execution (Doing the wrong thing)
    - **Lapses:** lapse is a failure of memory (Forgetting to do the right thing)
  - <u>Mistakes</u> - Using wrong procedure for the goal:
- **Avoiding Capture and Description Slips:**
  - Avoid habitual action sequences with identical prefixes
  - Avoid actions with very similar descriptions
  - Keep dangerous commands away from common ones
- **Avoiding Mode Errors:**
  - Eliminate modes
  - Increase visibility of mode
  - Spring loaded or temporary modes
- **Error Prevention:**
  - Disable illegal commands
  - Use menus & forms, not command languages
  - **Confirmation Dialogs**
  - **Writing Error Message Dialogs:** Best error message is <u>none at all</u>
  - **Error Message**: Be Precise and Comprehensible
  - **Suggest Reasons and Solutions**: Give constructive help: why error occurred and how to fix it
  - **Be Polite:** Avoid loaded words: Fatal, illegal, aborted, terminated

- **Clearly Marked Exits:** Long operations should be cancelable & All dialogs should have a cancel button
- Support Undo

Usability Standarts

ISO 9241 11 Ergonomic requirements for office work with visual display terminals
Part 11 : Guidance on usability
Explains how to identify the information needed to specify or evaluate usability in terms of measures of:

- performance
- satisfaction

ISO 13407 --> ISO 9241 210 Human centred design processes for interactive systems

ISO 13.180 Ergonomics

ISO 13407 adresses
Four Principles of Human Centered Design:

- active involvement of users
- appropriate allocation of function to system and to user
- iteration of design solutions
- multi disciplinary design

Four Human Centered Design Activities:

- understand and specify the context of use
- specify user and organizational requirements
- produce more than one candidate design solution
- evaluate designs against requirements

The User

HIPS: Human Information Processing System

1. **Sensorial Subsystem** - Senses
2. **Perceptual sub-system**
   a. system memory perceptual buffer (iconic, echoic, …)
   b. process - pattern recognition
3. **Cognitive sub-system**
   a. <u>memory</u>
      i. short term/working memory (STM)
      ii. long term memory (LTM)
   b. <u>processes</u>
      i. selective attention
      ii. problem resolution
      iii. learning …

1. Sensorial Subsystem

<u>Input</u>: 5 senses (Visio prefered)
<u>Output</u>: Communication system

Vision
- Compensate movements and illumination changes
- Context used to solve ambiguities
- May be tricked: Visual illusion came from excessive compensation.

<u>Visual Illusions</u>: What we see does not depend only of the stimulus
<u>Pattern Recognition</u>: Process that matches information from a stimulus with information retrieved from memory

Audition
- Information on direction, objects and distance

Touch
- Important feedback
- Several receptors in skin:
  - Termoreceptors: cold and hot
  - Nociceptor: pain
  - Mecanoreceptor: pression

Smell and Taste
- High latency

2. Memory
- **Sensory memory**/iconic memory: very short - 1 a 2 seconds
- **Short term memory**/working memory: 18 seconds, 7 +/- 2 items
    - Quick Access, Quick Forget
    - **Increased with chunking:** : the largest meaningful unit that a person recognizes; depends on the person knowledge
- **Long term memory**
    - Slower access (+/- 1/10 sec)

Processes related to LTM:
- **Storage/Remembering**
    - Repetition (from STM to LTM)
- **Forgetting**
    - <u>Decay</u>
        - Information is lost gradually but slowly
    - <u>Interference</u>
        - More recent information replace older information
- **Information Retrieval**
    - <u>Recognition:</u> remembering with the help of a visible cue - aka "Knowledge in the world"
    - <u>Recall:</u> remembering with no help - aka "Knowledge in the head"

3. Emotion
Stress will increase the difficulty of problem solving
Relaxed users will be more forgiving of shortcomings in design
Aesthetically pleasing and rewarding interfaces will increase positive affect

4. Physical Characteristics
- <u>Long term</u>
    - color vision deficiencies, physical deficiencies, handedness, age
- <u>Short term</u>
    - Stress, fatigue

5. Experience and Knowledge

- Education and reading level
- experience with the system and task ...

6. Work and Task

- Usage frequency
- Training ...

Models for Design

Iterative Human-Centred Design: Requirements, Prototypes, Evaluation
Models used to obtain user requirements in their social and organization context

- **Personas:** fictional characters based upon research in order to represent the different types of users
- **User Models:** of the users' mental, perceptual and motor processes

User Models

Know your user:

- Identify characteristics of target user population
- Identify types of users
    - By role (student, teacher)
    - By characteristics (age, motivation)

How to perform user analysis:

- **Techniques:** Questionnaires, Interviews, Observation
- **Obstacles:** Developers and users are isolated

Personas

Fictitious character used as representative of a user class
Advantages:

- Convenient handle for talking about user classes
- Focuses on a typical user, rather than an extreme
- Encourages empathy

Disadvantages:

- May be misleading
- Stereotype trap

Personas decifram

- **Activities:** What the user does; frequency and volume
- **Attitudes:** How the user thinks about the system's domain and technology
- **Aptitudes:** What education and training the user has and ability to learn
- **Skills User:** capabilities related to the system's domain and technology
- **Motivations:** Why is the user engaged in the system's domain

Cognitive Models

Represent the user in an interaction with the system: i.e. model aspects of user knowledge, intentions or processing.

- **Object and tasks hierarchies:** GOMS
- **Linguistic models**
- **Physical and device**: KLM - Keystroke Level Model

GOMS aims at understanding the user's cognitive processes while performing the task

GOMS - Goals, Operators, Methods and Selections

- **Goals:** what the user wants to attain
- **Methods:** possible decompositions of the goal into sub goals (e.g. Select an option "Save" or press "ctrl
- **Operators:** basic operations that the user has to perform to use the system; may affect the system or not (press a key or read a message)
- **Selections rules:** to select the possible methods (taking into account the type of user and the system status)

A typical GOMS analysis consists in **decomposing** a high level goal in a sequence of tasks (sub goals).

Analyzing the structure of the GOMS **decomposition** may give an approximate measure of :

- <u>Short Term Memory</u> load (depth of the goal structure)
- <u>Time needed</u> (a time for each operator)

Capacities:

- It has been used in cognitive model research
- It may describe adequately how experienced users perform routine tasks
- Associated to a device model allows time estimates

Limitations:

- It does not give information concerning user knowledge to estimate training or transfer times

KLM - Keystroke Level Model

Predicts user performance based on motor system characteristics
Models unitary interaction tasks (simple command sequences <20s)
(e.g. change the font of a word)
These tasks have two phases:

- **Acquisition** (building the mental representation of the task)
- **Execution** (using the system)
    - Motor:
        - **K** Keystroke (varies with typing skill)
        - **B** Button press of the mouse)
        - **P** Pointing at a target (Fitts' law) - Predictive model of human movement
        - **H** Homming between mouse and keyboard
        - **D** Drawing using mouse
    - Mental:
        - **M** Mentally preparing for physical action
    - System:
        - **R** System Response (often may be ignore)

It allows only approximate results ; thus reasonable estimates concerning the user are enough

Task Analysis

It is the analysis of how people perform their work

- what they **do**, **use** and **need to know**

TA models also aspects of the real world not part of the system
TA describes the tasks users perform from an external point of view and has more detail
TA is more used in early phases of the S/W lifecycle and GOMS for evaluation

HTA - Hierarchical Task Analysis produces:

- A task and sub task hierarchy
- Plans with a sequence and execution conditions

Different analysts usually produce different results

TA Info Source
The process of analysis in general **triggers new questions**, thus several phases of data collection and analysis are needed

- There are several types of **information sources**:
  - Documentation
  - Observation (expensive)
  - Interviews

Dialog Notation

Dialog is the syntactic level of human computer interaction
<u>Related to:</u>
- the system **semantic:** what it does
- the system **presentation:** how it looks

Prototyping

Paper Prototype

- Get **feedback earlier**, **cheaper**
- Obtain **users feedback** concerning general aspects
- Easy to **modify** and **throw away** even during user tests
- Valuable to **test** the UI **conceptual model**

Specifically adequate to get feedback concerning
- Concepts and terminology
- Navigation
- Contents
- Functionality

It does not need to have much detail, nor to be very realistic.
WHY?
- **Faster to build:** Sketching is faster than programming
- **Easier to change:** Easy to make changes between user tests, or even during a user test. No code investment everything will be thrown away (except the design)
- **Focuses attention on big picture:** Designer does not waste time on details. Customer makes more creative suggestions
- **Non Programmers can help:** Only simple skills are required

Test user - 3 roles:
- <u>Computer</u>: Simulates system
- <u>Facilitator</u>: Presents interface and tasks to the user
- <u>Observer:</u> Don't talk, tae notes

What you can learn:
- **Conceptual model**: Do users understand it?
- **Functionality:** Does it do what's needed? Missing features?

- **Navigation** & **task flow:** Can users find their way around? Are information preconditions met?
- **Terminology:** Do users understand labels?
- **Screen contents:** What needs to go on the screen?

Computer prototype

- Interactive software simulation
- High fidelity in look & feel
- Low fidelity in depth

What you can learn:
Everything you learn from a paper prototype, plus:

- **Screen layout:** Is it clear, overwhelming, distracting, complicated? Can users find important elements?
- **Colors, fonts, icons, other elements:** Well chosen?
- **Interactive feedback:** Do users notice & respond to status bar messages, cursor changes, other feedback
- **Efficiency issues:** Controls big enough? Too close together? Scrolling list is too long?

Computer Techniques

- **Storyboard:** Sequence of painted screenshots. Sometimes connected by hyperlinks hotspots
- **Form builder:** Real windows assembled from a palette of widgets (buttons, text fields, labels, etc.)
- **Wizard of Oz:** Computer frontend, human backend

StoryBoard:
<u>Pros;</u> You can draw anything
<u>Cons</u>: No text entry; Widgets are not active. Better evaluated with other technique such as Heuristic Evaluation

Form Builder:
<u>Pros:</u> Actual controls, not just pictures of them. Can hook in some backend if you need it - <u>But then you won't want to throw it away</u>
<u>Cons:</u> Limits thinking to standard widgets. Less helpful for rich graphical interfaces.

Wizard of Oz:
Software simulation with a human in the loop to help

UI Software Architecture

View Tree

A GUI is structured as a **tree of views**

View: is an object that displays itself on a region of the screen

Output: GUIs change their output by mutating the view tree. A redraw algorithm automatically redraws the affected views•
Input: GUIs receive keyboard and mouse input by attaching listeners to views
Layout: Automatic layout algorithm traverses the tree to calculate positions and sizes of views

Listener

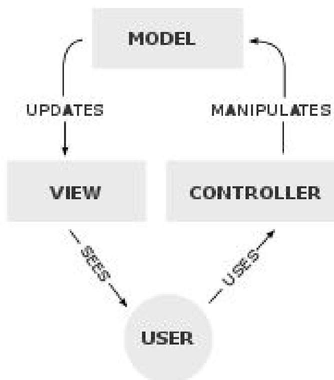
GUI input handling is an example of the Listener pattern

- An **event source** generates a stream of discrete events - e.g., mouse events
- Listeners **register interest** in events from the source
    - Can often register only for specific events e.g., only want mouse events occurring inside a view's bounds
    - Listeners can unsubscribe when they no longer want events
- When an **event occurs**, the event source distributes it to all interested listeners

Input and output in GUIs are separated:

- **Output** is represented by the **view tree**
- **Input** is handled by **listeners** attached to views

Model view controller



MVC is an architectural pattern commonly used for developing user interfaces
Separating internal representations of information from the ways it is presented to and accepted from the user

Model - central component of the pattern:

- expresses the application's behavior in terms of the problem domain, independent of the UI.
- It directly manages the data, logic and rules of the application

View - any output representation of information:

- Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants

Control - accepts input and converts it to:

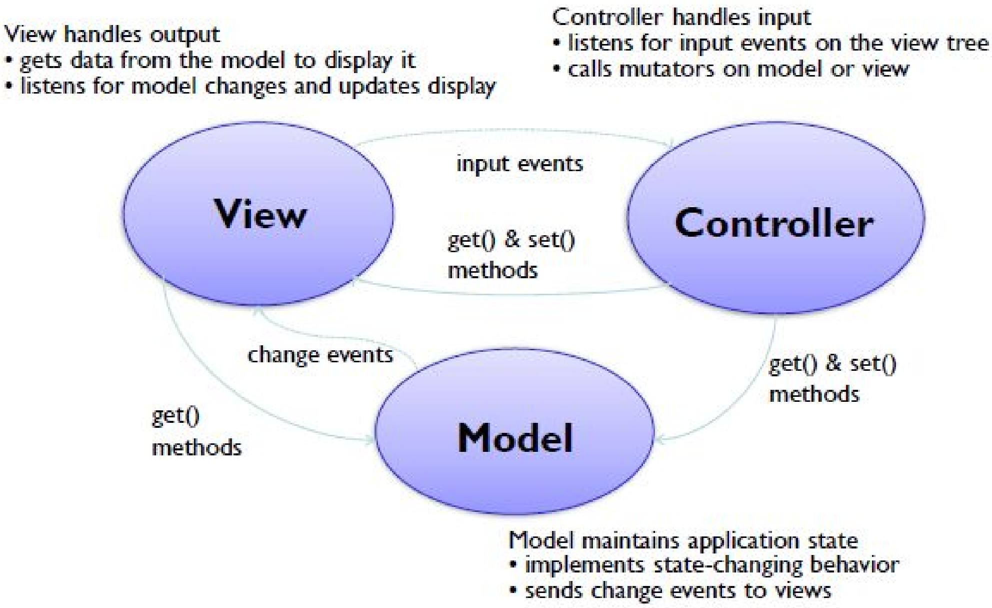
- accepts input and converts it to commands for the model or view

Advantages:

- Separation of responsibilities
    - **Model**: data | **View**: output | **Controller**: input
- Decoupling
    - Views and models can be changed independently, reused or share models
    - Multiple views for a shared model
- Ease of modification

Disadvantages:

- Code navigability
- Multi-artifact consistency
- Pronounced learning curve



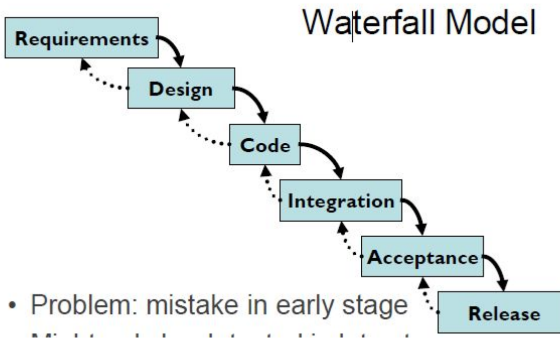
Widget - Tightly Coupled View & Controller

The MVC idea has largely been superseded by a MV (Model-View) idea
A widget is a **reusable view** object that **manages** both its <u>output and its input</u>

- Widgets are sometimes called components (Java) or controls (Windows)

User-Centered Design | Interactive Software Lifecycle

-> Iterative Design

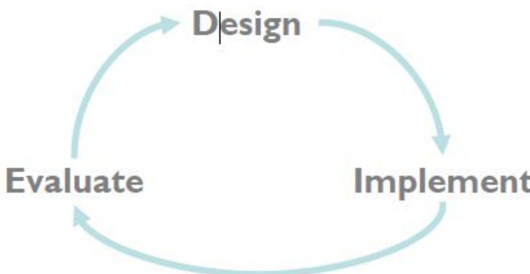


Waterfall Model:

Problem: mistakes in early stage might only be detected in late stages
IS BAD BECAUSE:

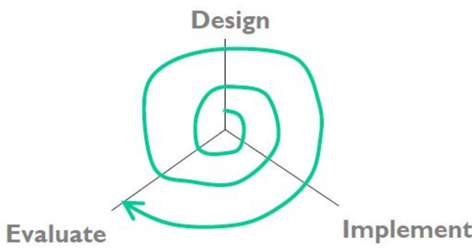
- **User interface design is risky:** So we re likely to get it wrong
- **Users are not involved in validation until acceptance testing:** So we won't find out until the end
- **UI flaws often cause changes in requirements and design:** So we have to throw away carefully written and tested code

Iterative Design:



Every iteration corresponds to a release: Evaluation (complaints) feeds back into next version s design
Using your paying customers to evaluate your usability: They won t like it

Spiral Model



User-Centered design

- Considers the **needs of the users** during the entire design process Iterative design
- Early focus on users and tasks
    - **user analysis:** who the users are
    - **task analysis:** what they need to do
    - **involving users as evaluators**, consultants, and sometimes designers
- **Constant evaluation**
    - Users are involved in every iteration
    - Every prototype is evaluated somehow

-> Early Prototypes Can Detect Usability Problems

Early iterations use cheap prototypes: Parallel design is feasible: build & test multiple prototypes to explore design alternatives

Later iterations use richer implementations: after UI risk has been mitigated

U-C Design

- Involves knowing:
    - **Usability principles** (independent from technology)
    - **Usability paradigms** (more technology dependent)
    - **Methods**
    - **Evaluation**
- We must know the success examples (usability paradigms)
- Understand why they work (usability paradigms)
- Use the adequate methods (user centered approach)
- And test , re design, etc.., until we attain the usability goals

Methodologies:

Usability testing, Focus groups, Online surveys, Personas, Scenarios, Task analysi

ISO standard 13407 (1999) Human centered design processes for interactive systems:

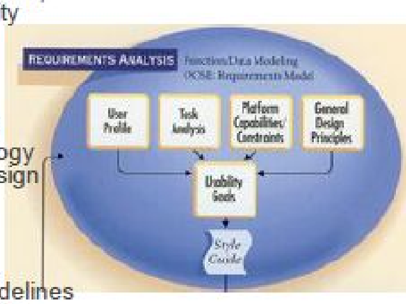
"Human centered design is an approach to interactive system development that focuses specifically on making systems usable. It is a multi-disciplinary activity."

Benefits:

- Following UCD best practices, helps to identify challenges upfront so that a solution can be found early
- By putting a larger emphasis on UCD principles and practices, iterative improvements can be made and avoid costly large scale rework

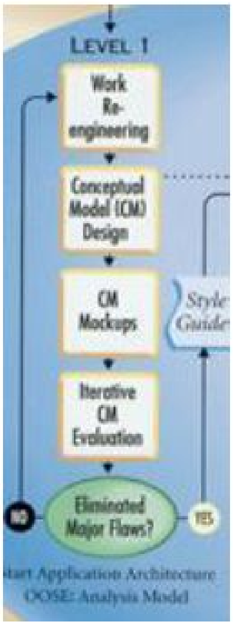
Example of UCD Methodologies: Requirement analysis

User Profiles	Establish user characteristics important for UI design
Contextual Task Analysis	Obtain a user-centered model of work as it is currently done; extract the product usability requirements
Platform Capabilities and Constraints	Establish capabilities and constraints of the technology platform which limit UI design alternatives
General Design Principles	Identify principles and guidelines that may be relevant for the product under development
Usability Goal Setting	Establish specific quantitative and qualitative usability goals to drive UI design



Design testing and development – Level 1

Work Reengineering	Reengineer the current user work model to realize the potential of automation and more effectively support business goals
Conceptual Model Design	Establish a coherent and rule based, high level UI design framework to set the stage for design at lower levels
Conceptual model Mock-ups	Support evaluation, refinement, and validation of the Conceptual Model Design
Iterative Conceptual Model Evaluation	Evaluate, refine and validate the Conceptual Model Design



Design testing and development – Level 2

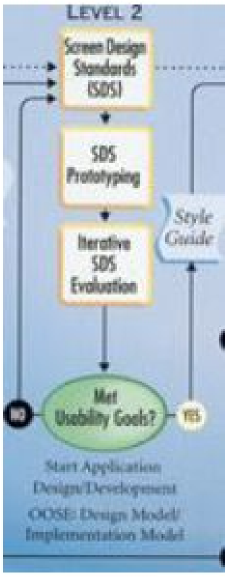
Anteri

Screen Design Standards — Establish a set of design standards to set the stage for detailed UI Design

Screen Design Standards Prototyping — Support the evaluation, refinement and validation of the Screen Design Standards

Iterative Screen Design Standards Evaluation — Evaluate, refine, and validate the Screen Design Standards

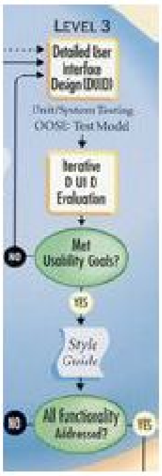
Style Guide Development — Document the Conceptual Model Design, the Screen Design Standards and the output of Requirement Analysis



Design testing and development – Level 3

Detailed User Interface Design — Design the complete, detailed product User Interface

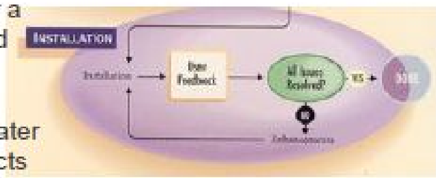
Iterative User Interface Evaluation — Evaluate, refine, and validate key subsets of the detailed User Interface Design



Installation

User Feedback — Obtain usability data after a product has been installed and used

Inform the UI Design for later releases or related products



Interaction Styles - Direct Manipulation

Actions are performed **directly on visual representations** of the objects (táctil pex.)
It is **characterized** by:

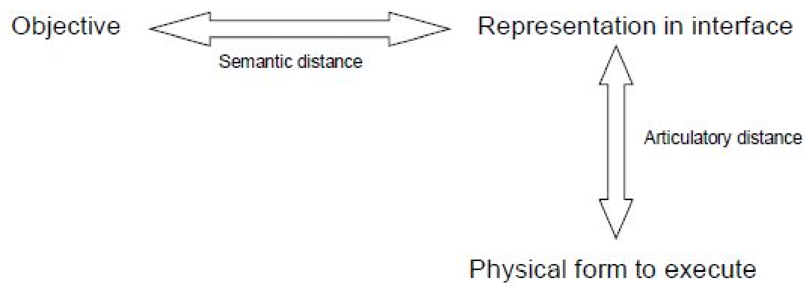
- Continuous representation of objects
- Physical actions instead of command languages
- Fast, incremental, reversible actions with visible results

UI has the following:

- **"Distance"** between user's thoughts and system's requirements for using it to perform task
- **Engagement**: Feeling that the user is directly manipulating the objects of interest

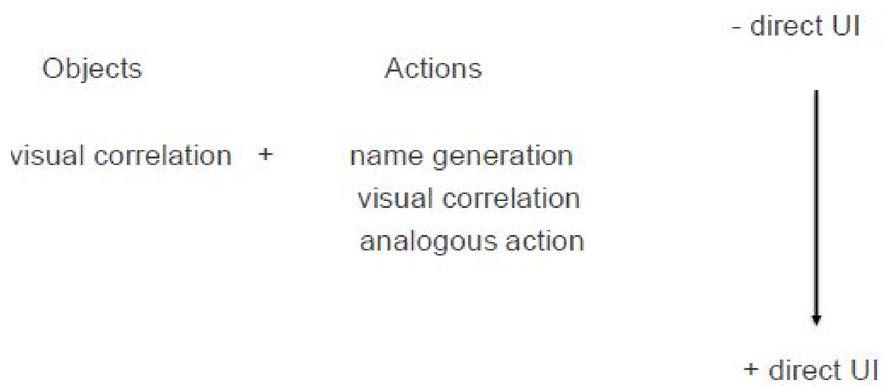
Distance:

- **Semantic Distance:** subjective distance between the user's goal and interface semantics
- **Articulatory distance:** distance between the meaning of the actions and their physical form



If the objects and actions do not support the users' goals, **semantic distance** is **high**
When the physical way actions are performed is more similar to their meaning, **articulatory distance** is **smaller**

Specifying objects by visual correlation implies the presence of direct manipulation
How actions are specified defines the degree of direct manipulation



Advantages:

- Easy to learn and remember
- Direct, WYSIWYG
- Flexible, easily reversible actions
- Immediate visual and context feedback
- Less prone to errors

Disadvantages:

- Not auto explanatory
- May be inefficient
- Difficult to draw recognizable icons (particularly for actions)
- Icons occupy more screen real estate then text
- Need to move between mouse and keyboard
- Small display might limit interaction

User Profile:

Knowledge and experience:

- Moderate system experience
- Moderate to high task experience
- Frequent usage of other systems
- Low computational literacy

Work and task

- Low frequency of use
- Moderate training
- Optional usage
- Low structured tasks

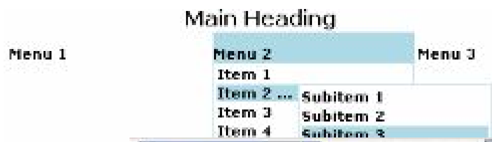
Guidelines:

- **Coherence**; Coherent Icon scheme
    - Express relation through icon similarity
    - Add names to icons to make them more recognizable
    - Icons must be conceptually and visually distinctive
    - Icons should be specific/familiar not abstract/non familiar
    - Express objects attributes through icons
- Good conceptual model
- Feedback
- Adequate organization of functionality
- Adequate screen layout
- Adequate colour usage
- Adequate error handling

Menus - Interaction Styles

Types of menus:

- **Single Menu:** Selection between one or several items
    - Binary
    - Multiple Options - Radio Buttons/Check Box
- **Pull Down Menus**: Always available/Shortcuts
- **Cascading Menus:** Anchor ->



- **Textual and Iconic**
- **Always Visible vs Popups**
- **Extensive lists** ->

<u>Advantages:</u>
- Auto explanatory
- Do not load memory (recognition rather than recall)
- Require little typing
- Prevent syntactic errors
- Visible improvements

<u>Disadvantages:</u>
- Not efficient
- Not flexible
- Not practical for many options

User profile to whom menus are adequate:

- <u>Knowledge and experience:</u>
    - Low system and task experience
    - Frequent usage of other systems
    - Low computational literacy
- <u>Work and Task:</u>
    - Low frequency of use
    - No training
    - Optional usage
    - Highly structured tasks

Menu Design Relevant Aspects

- **Menu structure**
- Adequate the menu structure to the task structure
- Minimize depth increasing breadth (within reasonable limits)
- Use an adequate ordering method
- Indicate currently unavailable options

- **Option ordering**
- **Conventional ordering**
    - Chronological (Time)
    - Numeric (Ascending or descending)
    - Physical properties (Size, speed, area,...)
- **When no obvious ordering**
    - Frequency of use
    - Order of use
    - Related categories
    - Alphabetical order

- **Option selection**
- Give feedback of selections to user
- Include tooltips if names or icons are not auto explanatory
- Indicate currently unavailable options
- Defined a predefined selection type
- Distinguish simple or multiple (radio button or checkbox)

Make clear the available options:

In grey to let users know they exist, but are unavailable

Make clear the difference between choices of only one or several

- "Radio button": Only a single option can be selected
- "Check box": Permits to make a binary choice.

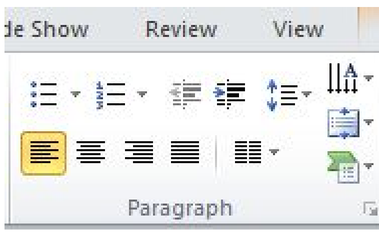
Show alternative ways of accessing the same functionality:

Example with "ctrl+o" for open

Include tooltips describing options if the names or icons are not clear:

MouseOver apresenta texto sobre o que é o ícone

Make clear that there are more options **->**



- **Menu invocation**
- **Navigation**

Fill-in-Forms - Interaction Styles

Fill in forms are particularly useful for routine, clerical work or for tasks that require much data entry

Advantages:

- Self-explanatory
- Recognition instead of recall
- Allow many different inputs (unlike menus)
- Give context and guide the user
- New functionality is visible (unlike command languages)

Disadvantages:

- Imply knowledge of valid inputs
- Error prone
- Not very flexible
- Consume screen space

User profile to whom fill-in-forms are adequate:

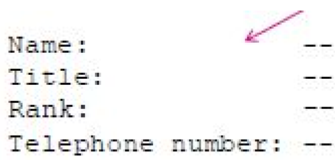
Knowledge and experience:

- Moderate or high typing skill
- High or moderate task experience
- Moderate or low application experience
- Moderate to high computer literacy

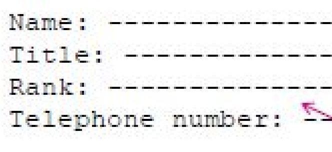
Task characteristics:

- Moderate to high frequency of use
- Low training
- Highly structured task

- **Organization and layout**
  - Avoid unfamiliar layouts (Ex: Name > LastName | NOT: LastName > Name - Order)



- **Titles and fields**
  - Alignment of titles ->
  - Differentiate titles and fields; do not show the cursor over fields



- **Input formats**

- **Instructions and help**
    - Obligatory with *
    - Input format must be familiar and clear (Date use __/__/____ | NOT ______)
    - Instructions to fill the fields should be clear

- **Navigation**

- **Error handling**

Clear error messages and Visible

Dialog Boxes

Combination of **Menus** and **Fill-in-Form**.

A few directives (**intern**)

- Representative title
- Sequence from Top-left to bottom-right
- **Consistent layouts**
    - Margins, grids, space, lines, boxes
    - Terminology, font, alignment
    - Standardized buttons (OK, Cancel)

A few directives (**extern**)

- Smooth appearing / disappearing
- Size to avoid occlusion
- Appearing close to items but without occlusion
- Easy to close

Function Keys

Hard Keys – Always invoke the same functionality (as the keys of a calculator and some specific keys of PCs)

Hard function keys have abbreviations of default actions printed on/besides them

Soft Keys – invoke different functionality according the context of use (as the keys (F1...Fn) and the generic keys of an Automated Telling Machine, e.g. Multibanco)

Soft function keys don't have abbreviations of default actions printed on/besides them, they may have "F-number" designations. Their value can be programmed and may depend on combinations (shift, alt +...)

Advantages (potential):
- Self-explanatory
- Recognition instead of recall
- Easy to use
- Flexible
- Require little or no screen real estate

Disadvantages:
- Limited number of keys
- Hardware expansions are expensive

User profile to whom function keys are adequate

Knowledge and experience:
- High or moderate task experience
- Moderate application experience

Task characteristics:
- Low to high frequency of use
- Low training or no training

-> Often used keys should be near the "home row"
-> Keys with serious consequences should not be easy to activate (e.g. ctrl Alt Del)

Command languages

User types in commands in an artificial language
- all knowledge in the head; low learnability

Command languages may be used not only through text but also via voice
- Ex phone or car commands (siri)

Goals when designing:
- Precision
- Compactness
- Ease in writing and reading
- Speed in learning
- Simplicity to reduce errors
- Ease of retention over time

Usability questions:
Does the language support necessary functions? Is it fast to enter a command? Is it easy to recognize what the command might do? Is it easy to recall a command? Are there few errors when using the language?

Advantages (potential):

- Powerful
- Flexible
- Efficient
- Do not take much screen real estate

Disadvantages:

- Difficult to learn
- Not self-explainable
- Error prone
- Improvements are not visible

User profile to whom Command languages are adequate:

Knowledge and experience:

- High task experience
- High application experience
- High computational literacy
- High typing skill

Task characteristics:

- High usage frequency
- Formal training

-> Balance richness and minimalism (similar to semantic distance in direct manipulation)
-> Use a coherent syntaxe
-> Use a natural and easy to remember action-object grammar
-> Command abbreviations should be simple and coherent
-> Easy to remember (not easy to recognize as for function keys)
Allow the following interaction features:

- Defaults
- Command edition
- Intelligent interpretation
- Type-ahead
- Feedback
- Help and documentation
- Make the language "user tailorable"

Relevant Issues

- Lexicon
- Semantics
- Syntax
- Interaction

Natural Languages - Mobile phone personal assistants

Communication between humans and computers through natural language involves:

- recognition
- generation

Natural languages as dialog style are not full blown natural languages, they are restricted natural languages

Natural languages (as dialog style) differ in “habitability” (how easy and natural is it for users)

Habitability (mismatch between the users' expectations and the capabilities of a natural language) is related to the language domains:

- **Conceptual** - the set of objects and actions provided by the language
- **Functional –** what may be directly expressed by the language
- **Syntactic –** syntactic forms that may be understood
- **Lexical** - the variety of words that may be understood

Conceptual model limitations are not very disturbing; however, limitations in any other domain make the language less habitable

<u>Advantages (potential):</u>

- Powerful
- Flexible
- Efficient

<u>Disadvantages</u>

- Assume problem domain knowledge
- Imply clarification dialogs
- Imply typing skills (if written)
- Improvements are not visible
- May create unrealistic expectations, foster irresponsible behaviours and generate negative reactions
- Difficult and expensive to implement

User profile to whom Natural languages are adequate:

<u>Knowledge and experience:</u>

- High task experience
- High application experience
- High computational literacy

- High typing skill

<u>Task characteristics:</u>

- High usage frequency
- Formal training

Interaction style	Main advantages	Main disadvantages	Application examples
Direct manipulation	Fast and intuitive interaction Easy to learn	May be hard to implement Only suitable where there is a visual metaphor for tasks and objects	Video games CAD systems
Menu selection	Avoids user error Little typing required	Slow for experienced users Can become complex if many menu options	Most general-purpose systems
Form fill-in	Simple data entry Easy to learn Checkable	Takes up a lot of screen space Causes problems where user options do not match the form fields	Stock control Personal loan processing
Command language	Powerful and flexible	Hard to learn Poor error management	Operating systems Command and control systems
Natural language	Accessible to casual users Easily extended	Requires more typing Natural language understanding systems are unreliable	Information retrieval systems

3D Interfaces

User interfaces involving 3D interaction (i.e. interaction in which the user's tasks are performed directly in a 3D spatial context).

Issue: User disorientation

Mobile User Interface

Usability on mobile devices is still a challenge

Desktop vs Mobile:

- Small Screen
- Fat Finger
- Context

	Mobile device	Desktop
Use	On the move, fast (voice, sms)	Task more complex, navigation, e-mail, etc.
Form	Small, in table/pocket	Needs a table, sit, possibly transportable.
Mobility	Used for mobility, battery life significative	Need cable or frequent charging
Connectivity	Slower with interruption	Faster and more reliable
Input	Challenge: touch	Keyboard and Mouse
Output	Small screen	Monitor with larger size and resolution
Storage	Still limited	Much superior

Mobile devices have specificity that might be considered in application design:

- Ambiental variations

Mobile device use is unpredictable and occurs in several contexts:

- temperature, climate, illumination affects performance (attery, processing, readability)
- Noisy or distraction rich environments make interaction more difficult
- User can move and system must allow to continue interaction
- User might be doing another task while using the device
- User might need to manipulate other objects during interaction

These variations are intrinsic to the use of mobile devices (not going to disappear with evolution)

Fundamental aspects to be considered in prototyping, developing and evaluating the systems.

- **Hardware limitations**

Mobile devices are a compromise between portability and performance what limits their performance:

- Slower, cheap CPUs
- Memory limitations
- Battery limitations
- Connectivity is unpredictable
- Screen size is reduced – Main limitation regarding interaction.

Many of these limitations improve but screen size will always be reduced.

Usability in mobile - Challenges:

-> (input) Entry inadequate:

- Difficult to operate an interface without mouse, interaction longer and with errors. Text entry particularly difficult.

-> (output) Small screens

- Less visible options, STM overload making interaction more complex

-> Bad interface design

- Often optimized for computers with no adaptation for mobile devices.

Input:

TouchScreen, MicroPhone, Camera, Sensors, …

Output:

Screen, Sound, Vibration, Lights, …

Information Presentation:

Main challenge, <u>information presentation </u>must be:

- Short and concise
- Carefully selected
- Easy to read
- Adapted to the situation

<u>Avoid presenting </u>too much information:

- It use valuable space in the screen
- Might distract user from what is relevant while doing another task
- Might be difficult to read on the move

Guidelines:

- <u>Simplify</u>
- <u>Minimize text input</u>: Typing will always be slower - Use intelligent entries
- <u>Stability</u>: Consider possible interruptions - Battery, etc
- <u>Error Prevention</u>: Error are even more easy to occur

Advantages:

- Additional sensors and possible interactions

Screen Layout and colors in user interface

Screen Layout

- The screen design is an important part of the UI development
- A poor screen design may degrade user performance
- Screen layout must be carefully designed
- There are numerous guidelines (we have seen already some of them)

Guidelines:

- **Information layout**
- Include only the relevant information
- Include all necessary information
- Begin at the top left corner and align left (Western Culture)
- Group items according to type
- Leave plenty of white space
- Use leaders in multiple columns

- **Text**
- Avoid using only capital letters
- Avoid text with many capital letters
- Do not use too many fonts for emphasis
- In multiple columns use leaders or greying

-> Use greying
-> Use fonts for emphasis
-> Left aligned: more readable
-> In multiple columns it is difficult to read across gaps:
Use **Leaders** - linhas
Use **Greying** - por fundo cinzento alternado

- **Error Messages**
- Brief and concise
- Be specific and understandable
- Be positive
- Be helpful
- Have a detail level adequate to user knowledge and experience

Too verbose	better
The processing of the text editor yielded 23 pages of output	Output 23 pages
Error in SIZE field Too vague	Error: SIZE range is 4 to 16
Cannot exit before saving file Negative	Save file before exiting
Bad/illegal file name	Maximum file name length is 8 chars
Syntax error 1542 Not helpful	Unmatched left parenthesis in line 210

- **Numbers**
  - Integers shall be right justified
  - Real numbers shall be aligned by the decimal point
  - Avoid unnecessary zeros (at left)
  - Long numbers shall be divided in groups of 3 or 4

	Better
10 100 1000 10000	10 100 1000 10000
100.00 25.365 5432.01 1.45591	100.00 25.365 5432.01 1.45591
10:1 p.m. 002	10:02 p.m. 2
6173954686	617-395-4686

Color

Photoreceptors:

<u>Rods:</u>

- Only **one kind** (peak response in green wavelengths)
- **Sensitive** to **low light** (scotopic vision
  - Multiple nearby rods aggregated into a single nerve signal
- **Saturated** at **moderate light intensity** (photopic vision)
  - Cones do most of the vision under photopic conditions

<u>Cones:</u>

- Operate in brighter light
- **Three kinds**: S(hort), M(edium), L(ong)
- S cones are **very weak**, centered in blue wavelengths
- M and L cones are **more powerful**, overlapping
- M centered in **green,** L in **yellow** (but called red)

Color Blindness:

Common deficiencies are explained by the lack of cones (color sensor cells in the retina) sensitive to the long and medium λ (dicromacies)):

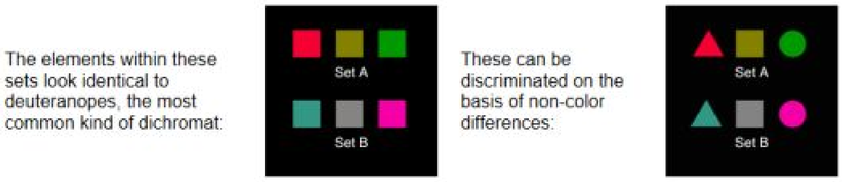
- Deuteranopia or Red Green blindness (MW ––"Green"
- Protanopia (LW ––"Red"

Avoid red and green to show difference since difficult to interpret by color blind (Deuteranopia)

Guides Color:

- **Use color parsimoniously**
  - Use a limited number of colors
  - Avoid using simultaneously several saturated colors
  - Avoid Saturated Colors
- **Use Few Colors**
- Do not expect to easily perceive order from color
- Don't use color coding on small elements
- Use neutral gray surrounds where color judgments are critical.

- **Background Colors**
    - Good contrast with the foreground
    - White good choice but bright displays
    - Pale (desaturated) yellow and very light good choice
    - Dark not so good: easy to have less legible text
- **Color deficiencies**



Advices Color:

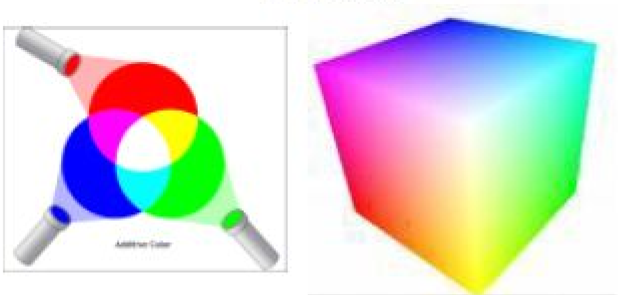
- Avoid saturated colors
- Get it right, Black an White
- Use less intense colors (medium hues or pastels)
- Avoid rainbow and primary colors
- Use color to highlight

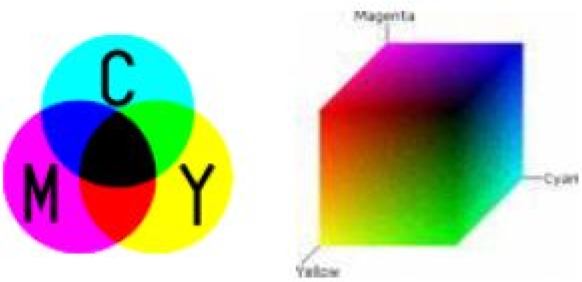
Color Models

Objects are perceived as having a color depending on the spectrum of the reflected light (or emitted)
Any color may be represented by the superposition of 3 basic colors, adjusting their intensity to match the intended color (RGB in additive systems)

- **RGB - (Red Green Blue)**
    - is H/W oriented and standard for computer monitors
- **CMY (Cian, Magenta, Yellow)**
    - is H/W oriented and standard for printers

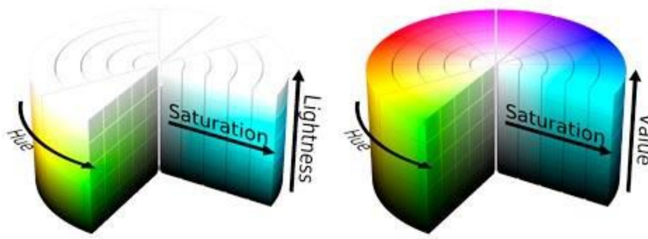
Additive





Subtractive

- **HSV (Hue, Saturation and Value)**
    - Hue is wavelength of color
    - Saturation is amount of pure color
        - 0% = gray, 100% = pure
    - Value is brightness
        - 0% = dark, 100% = bright
- **HLS(Hue, Lightness and Saturation)**
    - White has lightness 1.0
    - Pure colors have lightness 0.5



Input devices

Keyboards

- Tactil and auditory feedback important
- Some keys should be bigger (e.g. ENTER, SHIFT, CTRL)
- Some keys needs indicators (e.g. CAPS LOCK)

Pointing Devices

<u>Direct control:</u>

- **touch screen**
- **light pen**

<u>Indirect control:</u>

- **Mouse**
- **trackball**

TouchScreen:

- **Resistive:**
    - Two conductive sheets with a gap between
    - Responds to finger, stylus, any object
- **Capacitive**
    - Human skin changes surface capacitance
    - Responds only to bare skin
- **Inductive**
    - EM field from tablet induces signal from stylus
    - Responds only to special stylus
- **Optical**
    - Camera watches the surface
    - Responds to anything

Advantages:
- Direct hand cursor relation (distance, speed, direction)
- No need for additional space
- Continuous movement in all directions

Disadvantages :
- Lack of precisions due to finger
- Tiring for long periods
- Finger occlusion problem
- Screen might get dirty
    - Parallax (optical)
    - Temperature and humidity (capacitive)
    - Misalignment (resistive)

Light Pen:

Light pen/stylus:
- Point to a screen area and allow selection, position, etc.
- Direct control
- Button for selection

Problems:
- Hand occlusions
- Hand of the keyboard
- Hold the pen

Mouse:

Advantages:
- Direct relation between hand and cursor movement (distance, speed, direction)
- Allow speed control
- Allow continuous movement in all directions

Disadvantages:
- Require hand movement between mouse and keyboard
- Additional space (footprint)
- Hand-eye coordination

Track Balls:

Advantages:
- Direct relation between hand and cursor movement (speed and direction)
- Allow speed control
- Allow continuous movement in all directions
- May not need additional space (footprint)

Disadvantages:

- Require hand-eye coordination
- May require hand movement between trackball and keyboard

Guidelines for input devices:

Choose a device after a careful task analysis and test
Minimize hand and eyes movements
-> Use cursor keys for tasks involving:

- A lot of text manipulation
- Traversing a structured array of discrete objects

-> Use touch screens when

- There is no training
- Targets are large, discrete and scattered
- Space is important
- No (or little) text entry

Voice Recognition

Has two types of challenges:

- **Technological** (room for improvement…)
    - Continuous speech processing
    - Understanding of extended vocabularies
    - User independence
    - Very low error rates
    - Operation in noisy environments
- **Human factors**

Has advantages when the user:

- Has physical deficiency
- Must move around
- Has eyes busy
- Is in a low visibility or cluttered environment

Has inherent disadvantages:

- Voice is transient
- Does not have natural feedback
- May disturb other people
- May result in lack of privacy
- May be slower and more tiresome (overloading STM)

Guidelines:
Provide output dialog with structure to guide input
Use a distinct and familiar vocabulary to avoid errors
Consider voice input if technology constraints are acceptable considering:

- Ambient noise
- Privacy
- Vocabulary extent
- Error cost

Consider voice input when:

- The user has to move
- Has eyes or hands busy

Avoid voice input when:

- Privacy is important
- Error taxes, even low, are not acceptable
- Usage frequency is high
- Speed is important

3D interfaces

Inputs:

- Trackers: Magnetic, Optical, ...
- Navigation and manipulation interfaces: Tracker-Based, TrackBalls, ...
- Gesture interfaces: Glovess

Output Devices

Vision / Graphic displays

- Personal
  - Monitors
- Large Volume Display
  - Projectors, Walls
- Technology: CRT, LCD, e paper, projector
- Stereoscopic displays
- Autostereoscopic
- HMDs

Voice

- Digitized concatenates recorded basic sounds
- Synthesised concatenates sounds generated with models

Advantages of using voice output user has
- physical deficiency
- to move around
- hands and eyes busy
- Adverse conditions: low visibility, low O_2, high Gs

Disadvantages:
- Is tiresome and uncomfortable for long periods
- Is transient (overload STM)
- May have privacy issues
- May disturb other people

Consider voice output as an alternative when the user must move around, has hands and eyes busy
Avoid voice output in open environments, when the privacy and security are important issues and frequency of usage is high

Touch and Force Feedback
Touch: Relies on sensors in and close to the skin
Force Feedback: Relies on sensors on muscle tendons and bones/joints proprioception
Touch easier to implement then Force Feedback

Touch feedback mouse; Force feedback joysticks

Taste and Smell

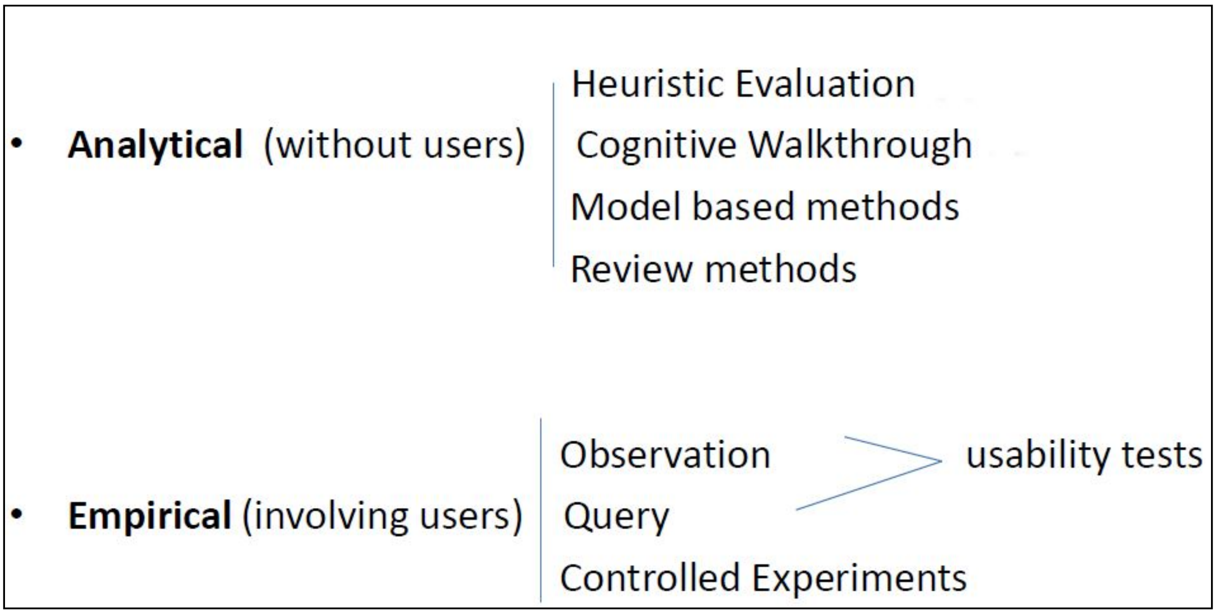
Importance of **smell**:

- Commonly accepted that influence how we act and fell
- Can stimulate memorization
- Can enhance sense of presence by recalling experience and modifying emotional state.

Taste:

- Useful as interaction modality? Unexplored research area.
- Important role in perceiving the world

Evaluation Methods



Heuristic Evaluation - (Nielsen and Molich 1990)

A “discount usability engineering method” for quick, cheap, and easy evaluation of a UI design
Meant to find the usability problems in the design so that they can be attended to as part of an iterative design process.

How to Perform?

- First make a general analysis to get to know the UI
- Then, make a systematic analysis having in mind the heuristics
- Take note of each potential problem, the heuristic and the severity grade
- Finally, compile all the potential problems

10 Heuristics

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition rather than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users recognize, diagnose, and recover from errors
- Help and documentation

Advantages:

- May produce useful results with modest investment
- Simple to apply even by not very experienced evaluators
- May be used along the development process from early phases

Disadvantages:

- Subjective (partially overcome with more and more experienced evaluators)
- Tends to find many small problems which may not be very important
- Can't find all usability problems
    - evaluation involving users is needed!

Cognitive Walkthrough (Wharton, et al., 1992)

Usability inspection method not involving users (analytical)
Focused on **assessing learnability** (i.e., how easy it is for new users to accomplish tasks with the system)

How to perform?

1- Task analysis: sequence of steps or actions required by a user to accomplish a task, and the system responses
2- Designers and developers walkthrough as a group, asking themselves a set of questions at each step
3- Data gathering during the walkthrough: answering the questions for each subtask usability problems are detected
4- Report of potential issues
5- UI redesign to address the issues identified

CW Four questions:

- **Will the user try to achieve the effect that the subtask has?**
    - (Does the user understand this subtask is needed to reach the goal?)
- **Will the user notice that the correct action is available?**
    - (E.g. is the button visible?)
- **Will the user understand that the wanted subtask can be achieved by the action?**
    - (E.g. the button is visible but the user doesn't understand the text and will not click on it)
- **Does the user get feedback?**
    - Will the user know that they have done the right thing?

Common issues

- The evaluator may not know the optimal way to perform the task; the method involves the optimal sequence of actions
- Involves an extensive analysis and documentation and often too many potential issues are detected, resulting very time consuming

Thus:

Lighter variants of Cognitive Walkthrough were proposed to make it more applicable in S/W development companies

Streamlined Cognitive Walkthrough (Spencer, 2000)

Only <u>two questions</u>:

- **Will the user know what to do at this step?**
- **If the user does the right thing, will they know that they did the right thing, and are making progress towards their goal?**

And a set of rules to streamlining the walkthrough and trade-off granularity for coverage

This will work if the usability specialist:

- takes care to prepare the team for the walkthrough,
- avoids design discussions during the walkthrough,
- explicitly neutralizes defensiveness among team members,
- streamlines the procedure by collapsing the first three questions into one question,
- and captures data selectively

Limitations of Analytical Methods

- Are subjective
- Involve several usability experts
- Cannot find all usability problems

Thus, empirical methods (involving users) are needed

Ethics in applying empirical methods

Involving users implies specific cautions:

- Asking for explicit consent
- Confidentiality
- Security (avoid any risk)
- Freedom (users may give up at any time)

- Limit stress

Observation

Has many variants from very simple to very complex and expensive:

- **Direct**: observer takes notes
- **Undirect**: through audio/ video – more complex and time consuming
- **Think Aloud**: users are asked to explain what they are doing
- **Logging**: users activity is logged by the system
    - **Combinations** of the previous, etc

Query

Two main variants:

- **Questionnaire** (reach more people; less flexible)
- **Interview**

Should be carefully prepared and tested

Usability Tests

Involve observation and query

<u>Main aspects</u>:

- Participants
- Tasks
- Test facilities and systems
- Protocol
- Usability measures
- Data analysis

Have a complex logistics

Standard: Common Industry Format **(CIF)** for usability test reports

Participants:

- The total number of participants to be tested
    - (a valid statistical analysis implies a sufficient number of subjects)
- Segmentation of user groups tested, if more than one
- Key characteristics and capabilities of user group
    - (user profile: age, gender, computing experience, product experience, etc.)
- How to select participants
- Differences between the participant sample and the user population
    - (e.g. actual users might have training whereas test subjects were untrained)

Tasks:

- The task scenarios for testing
- Why these tasks were selected
    - (e.g. the most frequent tasks, the most troublesome tasks)
- The source of these tasks
    - (e.g. observation of users using similar products, product specifications)
- Any task data given to the participants
- Completion or performance criteria established for each task
    - (e.g. n. of clicks < N, time limit)

Test Facilities and equipment:

- The setting and type of space in which the evaluation will be done
    - (e.g. usability lab, cubicle office, meeting room, home office, home family room, manufacturing floor, etc.)
- Any relevant features or circumstances that can affect the results
    - (e.g. video and audio recording equipment, one-way mirrors, or automatic data collection equipment)
- Participant's computing environment
    - (e.g. computer configuration, including model, OS version, required libraries or settings, browser name and version; relevant plug-in, etc.)
- Display and input devices characteristics
- Any questionnaires to be used

Protocol:

- Procedure: the logical design of the test
- Participant general instructions and task instructions
- The usability measures to be used:
    - a) for effectiveness (completeness rate, errors, assists)
    - b) for efficiency (times)
    - c) for satisfaction

Common Industry Format (CIF) for usability test reports ISO/IEC 25062:2006

Specifies the format for reporting the results of a summative evaluation
The most common type of usability evaluation is formative, (i.e. designed to identify problems that can be fixed)
A summative evaluation produces usability metrics that describe how usable a product is when used in a particular context of use
The CIF report format and metrics are consistent with the ISO 9241-11

Controlled Experiences

The "workhorse" of science ...
Important issues to consider:
- Hypothesis
- Variables (input or independent; output or dependent)
- Secondary variables
- Experimental design (within groups; between groups)
- Participants (number, profile)
- Statistics

- Define an **hypothesis**
- Define **input** (independent), **output** (dependent) and **secondary variables**
- Define **experimental design** (within-groups / between groups)
- Select the **participants**
- Prepare all the **documentation**:
    - list of tasks and perceived difficulty
    - final questionnaire
    - list of tasks for the observer to take notes
- Run a **pilot test**
- Take care of the logistics ... and after the experiment analyze data