ÔN THI CUỐI KỲ

Môn: Thiết kế giao diện

MỤC LỤC

1. Introduction

1.1. Brief definitions

- **User interface** – UI, “all components of an interactive system that provide information and controls for the user to accomplish specific tasks with the interactive system” (ISO 9241-110:2006)
- **Usability** - “extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use” (ISO 9241-11:1998)
- **User experience** – UX, “person's perceptions and responses resulting from the use and/or anticipated use of a product, system or service” (ISO 9241-210:2010)

1.2. Usability definitions

Usability is how well users can use the system’s functionality. There are five **dimensions** to be considered including within:

- Learnability - How easy it is to learn and use
- Efficiency - How quickly users perform tasks using the UI
- Memorability - How easy it is for users to reestablish proficiency
- Errors - Are the errors committed by users often? Is it easy to recover from errors?
- Satisfaction - Are users satisfied with the UI?

(Jakob Nielsen, Usability Engineering, 1994)

- Usability is highly dependent on user
    - Novice users need learnability
    - Infrequent users need memorability
    - Experts need efficiency
- Novice or expert is dependent on
    - Domain experience
    - Application experience
    - Feature experience
- Motivations for interests on Human Computer Interaction
    - Use of computers in life critical systems
        - Hospitals, Emergency Room (E.R.), etc.
        - Nuclear reactor, air traffic control, utilities
        - Emergency (911) dispatching, fire, police
    - Need for high-volume commercial systems
        - Banking/insurance/reservations/credit card
        - Inventory control/sales/billing
    - Use by a diverse population with limited training
        - Office automation
        - Home/personal/educational computing
    - Opportunity to enhance human intellect
        - Artist/movie effects workstations
        - Decision support/expert systems
        - Information retrieval
        - Simulations and training
- Usability is important
    - Usability is a condition for survival
    - Users often judge a system by its interface rather than its functionality.
    - A poorly designed interface can cause a user to make catastrophic errors.
    - Poor user interface design is the reason why so many software systems are never used.
- It is a matter of cost savings
    - User's time isn't getting cheaper. It doesn't follow Moore's Law.

    - Saving user's time from using user interface and resolving errors.
    - Disasters happen.

1.3. Problems with UI

- User interface accounts for a large portion of lifecycle costs, in which:
    - Application algorithms - 40%
    - Dialogue management – 40%
    - Presentation – 20%
- UI development is extremely labor intensive.
- UI's require frequent and extensive modifications.
- Modifications to the UI are difficult with the UI and the applications are tightly interwoven.
- Designing UI is hard.
    - Software engineers are not the user, they are trained to communicate with software engineers. But UI requires communicating with users.
    - The user is always right. Consistent errors are the result of the wrong interface.
    - But the user is not always right. They are not designers.

1.4. Human factors in UI Design

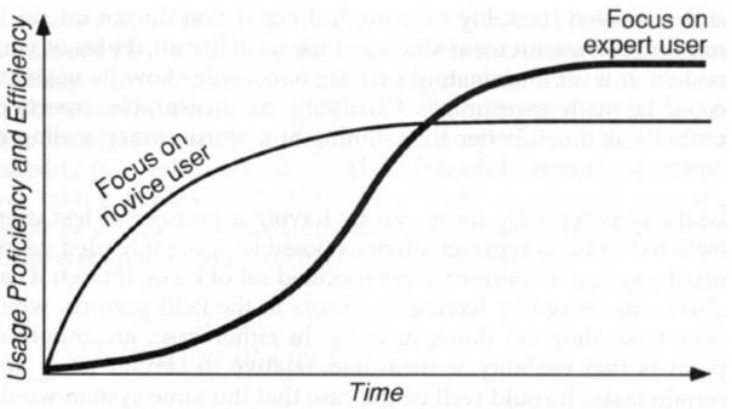
- Limited short-term memory
    - People can remember about 7 items of information.
    - If you present more than that, they tend to make mistakes.
- People make mistakes. And when they do, they tend to make more.
- People are different in physical capabilities.
- People have different interaction preferences. Some like pictures, others like text.

2. Usability Dimensions

2.1. Learnability

2.1.1. Recognition and Recall

- Recognition: Remembering with the help of a visible cue (evidence – bằng chứng). For example, you recognize your friend easily when seeing his/her face, but you may not remember his/her name.
- Recall: Remembering with no help. For example, you remember a person when someone refers to him. Do you remember their name?
- It is **easier to recognize** than recall things.
    - You don't remember every item in the File menu of Notepad
    - But you recognize their functions when you look at them.
- Implications
    - Performing operations via visual presentation is more learnable than via command line.
    - Direct manipulation is more learnable than other interface styles.
        - Example: pinch to zoom in and zoom out, drag and drop files between folders, etc.
- Nielsen's Learning Curve



(Jakob Nielsen, Usability Engineering, 1994, page 28)

2.1.2. Models

- Model of a system is a presentation of its operations.
    - Elements of a system.
    - How these elements work together to carry out its operations.
- Three kinds of models relevant to UI design.
    - **System model** or **implementation model**
        - Internal structure and interactions of the system's operations.
        - How systems work internally.
        - Visio's objects vs. Photoshop images
    - **Interface model**
        - How systems work through its interface.
        - Command line vs. Menu.
        - Editing Visio's objects vs. editing Photoshop images
    - **User model** or **mental model** or **conceptual model**
        - How the user thinks the system works
- Interface model encapsulates or hides system model. It should be simple and appropriate.
- Interface models should closely reflect the user model. User model may be wrong, so errors might happen.

2.1.3. Principles

Ways to communicate and present the system model

- **Affordances**: "Perceived and actual properties of a thing" – Don Norman. "Perceived" may be different from "actual". Example: folder icon, recycle bin icon, etc.
- **Natural mapping**: Physical arrangement of controls matches arrangement of their operations. It's best to map directly, but not always have to be (e.g. Light switches, Car's turn signals, etc.)
- **Visibility**: Operations should be visible to users
    - Unix commands are very **invisible** vs. Windows' menus.
    - Right click menus (or context menus) are not very visible → why iOS doesn't support much right-click.
    - Drag-drop is not either. But it's a direct manipulation style reflecting the real world.

→ Visibility versus Simplicity: More visibility may result in reduced simplicity

- **Feedback**: Actions should have immediate effects (e.g., push buttons, scroll bars, mouse icons). Feedback types: Audio, Visual, Haptic (giving a feeling, e.g., vibration of a mouse click)

2.1.4. Consistency

- Similar things should work similarly: Fonts, colors, icons, layouts, etc.
- Different things should look different.
- Consistency types:
    - Internal: within the system
    - External: across different systems
    - Metaphorical: reflecting real-world objects (e.g. A print icon is a metaphor of the printer, clicking on it should print the document, not delete it)
- Speak the user's language (e.g. Use common words, avoid slangs and jargon, avoid wordy and overly verbose)

2.1.5. Metaphors

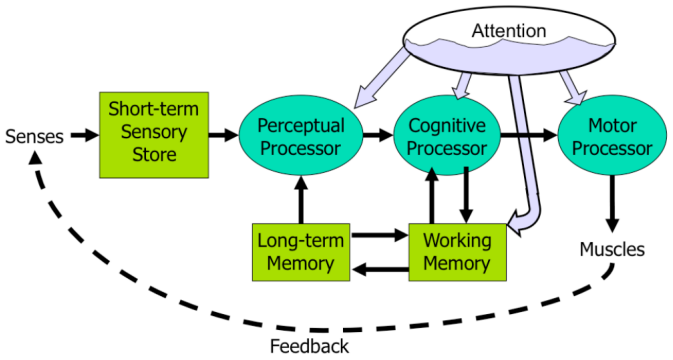
- Metaphor is a presentation of the real-world in user interface. E.g. A button with floppy disk icon should save the current document, not quit the application
- Advantages:
    - Highly learnable
    - Connect with user's existing model easily
- Problems:
    - Hard to design metaphors that are appropriate
    - Potentially deceptive and misleading
    - May not be used consistently everywhere
    - Culturally dependent (localization issue)

2.1.6. Platform Standard

- Follow guidelines of platforms.
    - MS Windows user interface guidelines
    - Apple user experience guidelines
- Follow frameworks. Various frameworks have their own looks and feels guidelines
- Learn from existing applications

2.2. Efficiency

2.2.1. Human information processing



(Source: MIT CS Course 6.813/6.831)

2.2.2. Laws

- Fitts's Law: calculate time T to hand to a target of size S at distance D away from the mouse pointer
- Implications of Fitts's Law;
    - Similar targets should be grouped
    - Targets at screen edge are easy to hit
    - Pie menu is faster to use than linear menu. About 15-20% according to a study by Callahan, 1994
    - Lengthy menus should be avoided
- Power Law of Practice: calculate time T to perform a task n-th time. Implications:
    - With practice, novices get better
    - But their performance becomes nearly flat
    - Refer to Nielsen Learning Curve

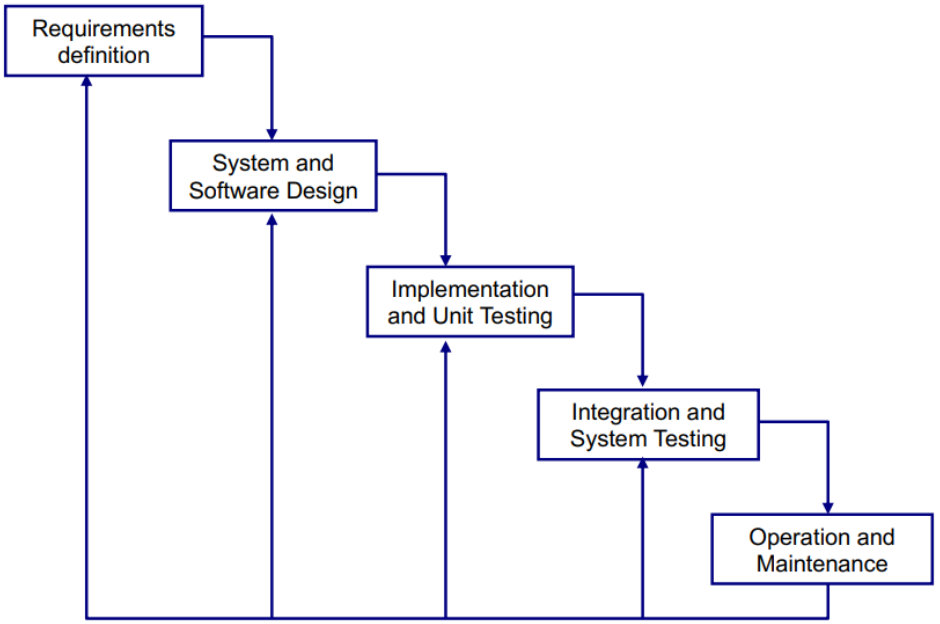
2.2.3. Principles

- Make frequently used targets big
- Group targets that are used together (e.g. Grouped toolbar buttons, menu items, etc.)
- Place frequently used menu items on top of menu
- Use screen corners and edges. E.g. When the users hover mouse to screen edge, it opens something
- Use keyboard shortcuts and menu accelerators. But **do not use too much**
- Predefine a group of styles
- Aggregating and choosing the most common selections by default. Use defaults
- Keep history (e.g., recent files in Word)
- Auto completion and auto suggestion. E.g. Google search
- Anticipation: Anticipate what users will do next and present corresponding operations for them. E.g. Clicking a picture in Powerpoint opens a Picture Format menu
- Try to avoid making menus that are more than 2-depth.
- **"UI design is like a joke. If you have to explain it, it's not that good."**

3. UI Design Process

3.1. UI Design Processes

3.1.1. Waterfall Model



Disadvantage: difficult to handle changes

- Users are not involved in evaluation until acceptance testing
- UI problems result in changes in requirements and design → Waste of effort spent earlier
- Inflexible partitioning of the project into distinct stages. It is difficult to respond to changing customer requirements
- It is only appropriate when the requirements are well understood. Few business systems have stable requirements.

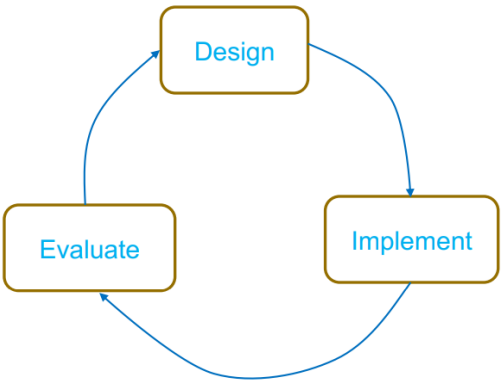
3.1.2.1. Shneiderman's Interactive Systems Lifecycle:

Software development lifecycle for interactive systems:

1. Collect Information
    - Organize the design team
    - Obtain management and customer participation
    - Conduct interviews with users
    - Submit written questionnaires to users
    - Estimate development, training, usage, maintenance costs
    - Prepare a schedule with observable milestones and reviews
2. Define Requirements and Semantics
    - Define high-level goals and middle-level requirements
    - Consider task flow sequencing alternatives
    - Create task objects and actions

    - Obtain management and customer agreement on goals, requirements, and semantic design
3. Design Syntax and Support Facilities
    - Compare alternative display formats
    - Design informative feedback for each operation
    - Review, evaluate, and revise design specifications
    - Carry out paper-and-pencil pilot tests or field studies with an online mock-up or prototype
4. Specify Physical Devices
    - Choose hard- or softcopy devices
    - Select audio, graphics, or peripheral devices
    - Consider work environment noise, lighting, table space, etc.
    - Carry out further pilot tests and revise design
5. Develop Software
    - Use appropriate development tools
    - Develop code
    - Perform unit test
6. Integrate System and Disseminate to Users
    - Assure user involvement at every stage
    - Conduct acceptance tests and fine tune the system
    - User documentation and training
7. Nurture the User Community
    - User support
    - Monitor usage and measurement
8. Prepare Evolutionary Plan

3.1.2.2. Interative Design

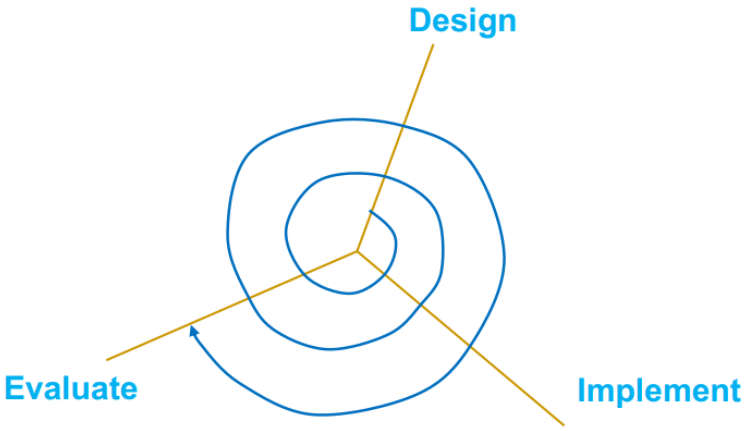


- Each cycle is one iteration
- Release is produced at the end of each iteration
- Customer's feedback and evaluations can be incorporated into next release
- Problems:
    - It's expensive to use customer's time to test
    - Customers may not be available
    - Customers don't like → they don't buy

3.1.2. Spiral Model

- Process is represented as a spiral rather than as a sequence of activities with backtracking.
- Each loop in the spiral represents a phase in the process.
- No fixed phases such as specification or design - loops in the spiral are chosen depending on what is required.
- Risks are explicitly assessed and resolved throughout the process.

An improvement of iterative design:



- Early cycles use cheap prototypes
    - Paper prototypes
    - Sketches on computer
    - Quick prototyping tools
- Providing multiple prototype alternatives
    - Parallel prototyping
- Later cycles should be better than early ones
- Only mature releases of later cycles can be distributed to users

3.1.2.1. User-centered Design

Also known as Participatory Design. A type of iterative design with Spiral. Focusing on users and tasks

- User analysis: who uses the system
- Task analysis: what users need to do

Getting users involved in the process (e.g. Users as evaluators, consultants and designers). Constant evaluation

Users evaluate prototypes and releases.

Advantages:

- Accurate information and useful suggestions
- Opportunity to argue over design decisions
- Increased ego involvement in system success

Potential problems:

- Users are not always available to participate
- Their time maybe expensive
- Users are not UI designers
- Users have strong ego and preferences
- UI designers overly obey users' preferences

3.1.2.2. Applying user-centered design

- You are all potential users of proposed apps.
- Users help identify problems
    - Members of other groups review a group's proposal
    - Collect feedback from potential users
    - Observer existing users' actions
- Users review and provide feedback: In each milestones, members of other groups provide feedback on design of a group
- User evaluation: By the end of the project: everyone will review the design of another group

3.1.2.3. How to be successful in teamwork?

- Define clear goals and expectations
- Assign clear responsibilities and tasks for everyone
- Talk about accountability: Who is responsible when things go wrong
- Meet weekly to review status, even if no assignment is due that week: Record meeting minutes
- Work early than late
- Understand your teammates: Motivation, commitment, capability

3.2. UI Design principles and rules

3.2.1. Principles

Determine users' skill levels ● Novice/first-time users ● Knowledgeable intermittent users ● Experts and frequent users, Identify the tasks ● Frequent actions ● Less frequent actions ● Infrequent actions
Choose appropriate interaction styles ● Direct manipulation ● Menu selection ● Form filling ● Command language ● Natural language
Use Shneiderman's eight golden rules of interface design
Prevent errors ● Constructive and informative error messages ● Organizing screens and menus functionally ● Providing feedback about the state of the interface ● Correct actions (e.g., grayed menu items) ● Complete sequences (e.g., wizard windows often have both Next and Finish buttons)

- Increase automation while preserving human control
    - Auto suggestion
    - Auto completion
    - Allowing users to change

3.2.1.1. Shneiderman's Eight Golden Rules

1. Strive for consistency
2. Cater to universal usability
3. Offer informative feedback
4. Design dialogs to yield closure
5. Prevent errors, rapid recovery
6. Permit easy reversal of actions
7. Support user control
8. Reduce memory load

4. Task Analysis

4.1. User analysis

User analysis is the process of identifying and describing the users who use the system.

Characteristics of target users:

- Age, gender, culture, language
- Computer experience
- Domain experience, application experience
- Usage frequency
- Physical limitations
- Education
- Motivation
- Work environment
- User relationships
- User social status (e.g., role, position)

Description of target users

- General information
- User characteristics (discussed above)
- User environment: Where will the tasks be performed?
- Major goals of the job: What is the end result?
- User roles (e.g., buyer, seller): if any
- User preferences
- Relationships among users: if any

Example: HaiLua.com.vn

A web-based application for users to sell and buy farming products.

Key features:

- Post products to sell (by farmers and others)

- Search for products
- Buy products
- Compare products' prices and other characteristics
- Rate sellers and buyers
- Provide comments or feedback on products or transactions

User analysis:

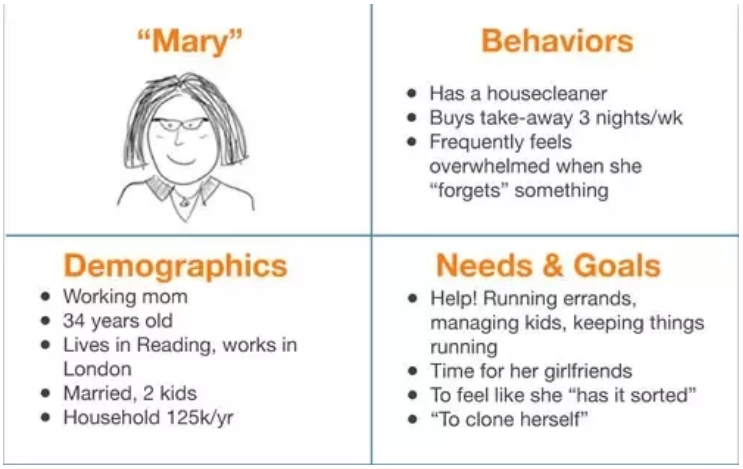
- By role
  - Buyers/customers
  - Sellers (farmers and traders)
  - Administrator
- By language/culture: Focusing on Vietnam farming products from Vietnamese farmers

Techniques to do user analysis:

- Recording
- Interviews
- Questionnaires
- Observation
- Combination of the above

Obstacles/challenges

- Designers and users are sometimes isolated
- Users may be overlooked by designers (e.g. Designers may make wrong assumptions about users)
- It's expensive and difficult to talk to some users (e.g. high-ranking people, doctors, executives, etc)



4.2. Task analysis

The process of analyzing and documenting the tasks that the system may provide to users.

- What needs to be done (goal)
- What conditions to do the task (precondition)
- What steps to be taken (subtasks)

Each task is often a goal to achieve by users

- Task analysis is an early step in UI design that provides basis for

- UI designing
- UI evaluation and improvement
- User documentation

Procedure: Two main steps

- Model tasks
    - Gathering information
    - Describing tasks into requirements
- Evaluate and refine
    - Review and update requirements

4.2.1. Model tasks

- Create a list of all tasks to be performed by users
- Rank the tasks by frequency of use and importance
- Gather other detailed information about each task
- Model the relationships (e.g., using use-case model)
    - between tasks and users
    - among tasks
- Present/describe tasks in forms of documents, diagrams, etc.

4.2.1.1. Techniques to do task analysis

- Techniques to gather information (same as doing user analysis)
    - Data recording
    - Interviews
    - Questionnaires
    - Observation
    - Combination of the above
- Technique to analyze: Task decomposition

4.2.1.2. Data recording

- Documents, manuals, instructions Notes, audio, photographs
- Notes + photographs
- Audio + photographs
- Video

4.2.1.3. Interviews

- Structured: tightly scripted, often like a questionnaire; replicable but may lack richness
- Unstructured: not directed by a script, rich but not replicable
- Semi-structured:
    - guided by a script but interesting issues can be explored in more depth
    - can provide a good balance between richness and replicability

4.2.1.4. Questionnaires

- Paper, email and the web used for dissemination

- Questions can be closed or open: closed questions are easier to analyze, and may be done by computer
- Can be administered to large populations
- Sampling can be a problem when the size of a population is unknown → is common online

4.2.1.5. Online questionnaires

- Advantages
    - Responses are usually received quickly
    - Data can be collected directly into database for analysis
    - Time required for data analysis is reduced
    - Errors can be corrected easily
    - Many online survey tools available (e.g. survey monkey)
- Problems
    - Sampling is problematic if population size is unknown
    - Preventing individuals from responding more than once
    - Delayed response

4.2.1.6. Observations

- Direct observation
    - in the field or in controlled environments
    - Structuring frameworks
    - Think-aloud protocol (e.g. Person talks about what they are doing, while they are doing it (or just before or after; Observer can ask probe questions)
    - Probe questions affect performance, as does thinking aloud

- Indirect observation
    - tracking users' activities
    - Physical location/movement
    - Interaction logging, timers

4.2.1.7. Task decomposition

Aims:

- describe the actions people do
- describe order of subtasks
- structure them within task subtask hierarchy

Hierarchical Task Analysis (HTA)

- introduced by Annett and Duncan (1967) to evaluate an organization's training needs
- very useful for analyzing and representing the behavioral aspects of complex tasks
- now widely used in interface design

4.2.1.8. Hierarchical Task Analysis (HTA)

- Breaks tasks into subtasks and operations or actions: These components are represented using a structure chart

- Includes:
    - identifying and categorizing tasks
    - identifying the subtasks
    - checking the overall accuracy of the model
- Useful for UI design: Enabling designers to envision the goals, tasks, subtasks, operations, and plan essential to users' activities

4.2.1.9. Generating the Hierarchy

1. Start from overall goal, e.g. clean the house
2. Get list of tasks
3. Break down into numbered sub-tasks
    - Group tasks into higher level tasks
    - Decompose lowest level tasks further
4. Describe each sub-task
    - How do we know when to stop? Is "empty the dust bag" simple enough?

4.2.2. Evaluate and refine requirements

- Evaluate, simplify and fix issues in the task description
- Evaluation techniques
    - Walk-through
    - Formal review/inspection
    - Offline review
    - Online review

4.2.2.1. Domain analysis

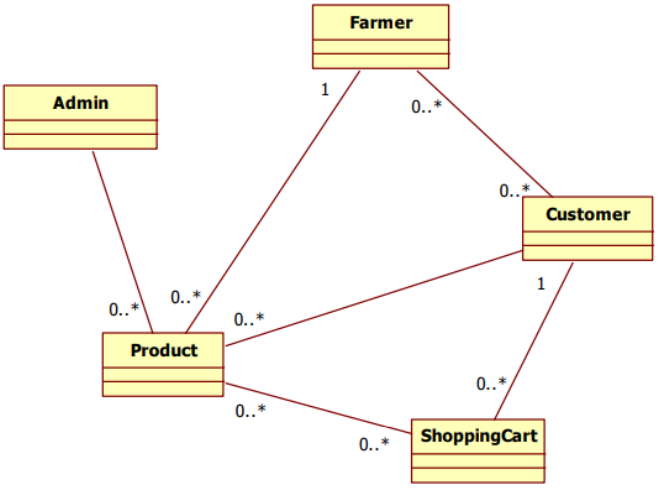
The process identifying data models for the system domain

- People and things
- How they are related

Outputs:

- Object or class models (e.g., using UML diagram)
- Data models (Entity Relationship models)

Example: HaiLua.com.vn's class model (high-level)



4.2.2.2. Requirement documents

- User analysis: Description of target users
  - General information
  - User characteristics (discussed above)
  - User environment: Where the tasks will be performed?
  - Major goals of the job: What is the end result?
  - User roles (e.g., buyer, seller)
  - User preferences
  - Relationships among users

- Task analysis, for each task
  - **Goal, precondition, subtasks**
  - Where the task is performed:
    - On Internet, desktop, mobile
    - At a kiosk, a workstation
  - How often is the task performed?
    - every hour, every day
    - once a day, once a month
  - What are resource constraints: One second, one minute, or not constrained
  - How the task is learned?: Training, install-and-use, by trying, by watching others
  - Task exceptions: What are exceptions for the task and how exceptions are handled
  - Who else are involved in the task

- User and Task analysis: Use-case model
- Domain analysis:
  - Object model
  - ER model

5. Prototyping

5.1. What is prototype?

A **prototype** is an original type, form, or instance of something serving as a typical example, basis, or standard for other things of the same category.

Examples:

- Screenshots
- Paper drawings
- prototype software - referred to as alpha grade, meaning it is the first version to run

The reasons why we need to prototype designs is to:

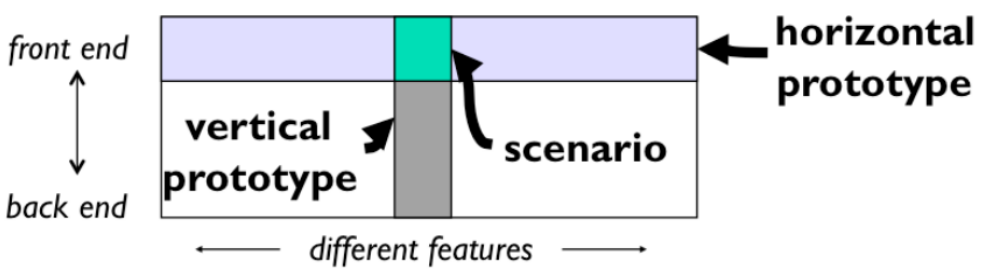
- Experiment with alternative designs
- Provide an early, concrete representation of design ideas
- Provide hands-on experience for all stakeholders (design teams, users, etc.)
- Easier to change or throw away
- Keep the design centered on the user: must test and observe ideas with users
- Facilitate iterative design and evaluation
- Reduce the risk of making customers surprise - “I won’t believe it until I see it”

5.2. Prototype fidelity

Low-fidelity	**High-fidelity**
• a set of sketches/storyboards providing a static, non computerized, non-working mockup of the planned product • often omits details	• a set of screens that provide a dynamic, computerized, working model of the planed product • working software

Dimensions of fidelity

- **Horizontal**: Prototypes cover many features but with little detail
- **Vertical**: Prototypes cover few features but with much in detail
- **Diagonal**: Prototypes cover down to a certain level and vertical



- Look:

    - Is the appearance and graphic design of the UI
    - Can be sketchy and hand-drawn
- Feel
    - Referring to input methods to interact with the UI
    - Pointing and writing (in paper-mockup) is different from mouse and keyboard
- Can paper prototypes have the feel attribute?

5.3. Paper prototype

- Using paper mockup to represent the UI
    - Sketches of screen appearance
    - Paper pieces show windows, menus, dialog boxes, toolbars
- Interaction
    - Pointing with a finger = mouse click
    - Writing = typing
- A person simulates the computer's operation
    - Putting down and picking up pieces
    - Writing responses on the mockup (screen)
    - Describing effects that are hard to show on paper
- Characteristics
    - Low fidelity in look and feel
    - High fidelity in depth as there is person to simulate the operation
- Advantages
    - Faster to build
    - Easier to change
    - Cheap
    - Focuses attention on big pictures: Designers don't waste time on details. Attract ideas from customers
    - Non-programmers can help
    - Convenient: You can prototype on the bus. You can utilize your time efficiently

5.3.1. Tools for paper prototyping

- White poster board: Used as background or window frame n Index cards
- Restickable glue
- White correction type
- Photocopier (for making copies)
- Pens, markers, scissors, etc.

5.3.2. Tips for good paper prototypes

- Make it large
- Make it monochrome (single color)
- Use description where necessary: You cannot represent tricky interactions like drag & drop, animation, progress bar
- Keep pieces organized: Use folders and envelopes
- Produce multiple alternatives: Better to get feedback

5.3.3. Role

Paper prototyping helps better understanding of:

- Conceptual model: Do users understand the UI?
- Functionality of the system: What features are missing in the UI?
- Navigation and task flow: Do users understand the navigation of the UI?
- Terminology: Are terms and levels understood?
- Screen content and layout: What are there in the UI?

However, it does not help in:

- Showing “look”: color, font, whitespace, etc.
- Demonstrating “feel”: efficiency issues: Interactions are in low fidelity (not real)
- Measuring response time
- Demonstrating animation and high-level of interaction: Actions like drag and drop, drawing, etc.

5.4. Computer prototype

Computer prototype is highly interactive software simulation. It also is high-fidelity in look and feel, while being low-fidelity in depth. Computer prototypes might not require backend procedures, they mostly focus on covering on horizontal level. Furthermore, it does not have a human simulating the backend like a paper prototype.

5.4.1. Role

Computer prototyping covers almost everything from paper prototypes, with a few additions:

- Better and higher-fidelity look
    - Screen layout
    - Colors, fonts, icons, etc.
    - Choices of controls
- Interactive feedback
- Efficient issues:
    - Controls are big enough?
    - Whitespace?
    - Distance between controls?

5.4.2. Advantages of computer prototype

- Faster than coding
- No debugging
- Easier to change and throw away
- Separate UI design ideas from what offered by UI toolkit (e.g., Visual Studios, C++ Builder): Your thinking is not limited to available widgets
- Non-programmers can do it

5.4.3. Techniques

- Storyboard: Sequence of painted screenshots, sometimes connected with links
- Form builder: Creating real windows with widgets such as buttons, windows, labels, etc
- Wizard of Oz: Computer frontend, human backend

Storyboarding tools: Figma, Invision, Photoshop, Balsamiq Mockup, Mockingbird, Justinmind, Axure, Pidoco, Excel, Visio, etc.	
Pros	**Cons**
• You can draw anything • Fast	• No or limited interaction: No text entry, Widgets aren't active

Form builders: FlexBuilder, Silverlight, Visual Basic, C++ Builder, Visual C#, Qt Designer	
Pros	**Cons**
• Actual controls → high-fidelity in terms of look • You can reuse the design for implementation à save effort from doing again	• Limits thinking to standard and available widgets • Content in each widget is not visible

Wizard of Oz: "Wizard of Oz" = "man behind the curtain"

- Software simulation with human in the loop to help
  - Human "wizard" mimics computational functionalities: system response interprets user input, controls computer to simulate appropriate output
  - Wizards are not always hidden
- Example: Simulate the speech recognition which is not available (human is needed to recognize speech)

Issues:

- Wizard has to be mechanical (pretending to be non-human)
- Worry about both UIs: for wizard and users

6. Web UI

6.1. Prevalence and importance of Web design

6.1.1. Prevalence of Internet

- Internet users (today)
  - World 3.6 billion; Vietnam 54 million
  - Facebook users: 2.2 billion monthly active users (MAU)

- Websites: over 1 billion
- Internet is now a dominant activity of a day
    - Average person spends 2 hours/day on social networks, 9 hours/day on Internet
    - Teens spend about 27 hours/week online

6.1.2. Importance of Web Design

Millions of websites on the Internet are growing everyday, even faster than before. Some are visited often, some to a small audience. Websites serve many predominant purposes, in which are: communication, education/information, e-commerce/e-business, and entertainment.

A good website design can lead to good return on investments. One must achieve these two elusive goals to make website design effective: user satisfaction and return on investment (ROI)

Usability is still the critical issue for Web design:

- if your site is difficult to use - people leave
- if it is not clear what is offered - people leave
- if it is hard to read - people leave
- if it is unattractive - people leave
- if it is boring - people leave
- if it doesn't work the way they are used to - they leave
- if they get lost - they leave
- if they get frustrated - they leave

A story - NY Times, Aug 30 1999, on IBM Web site

- "*Most popular feature was … search … because people couldn't figure out how to navigate the site.*"
- "*The second most popular feature was the help button, because the search technology was so ineffective.*"

After redesign: use of the "help" button decreased 84 percent, and sales increased 400 percent.

6.2. Text and Hypertext

6.2.1. Text

Texts impose relatively strict linear regression on the reader. Its structure and order of ideas are dependent on the author.

6.2.2. Hypertext

Hypertext is a dynamic organization of information through links and connections (hyperlinks). It follows a non-linear structure: blocks of text (pages), links between pages create a mesh or network, users follow their own path through information.

Advantages and challenges of non-linear structure:

Advantages	**Challenges**

● Very powerful ● Users can follow their own path through information.	● Hugely expanded connectivity ● Consequent loss of logical structure ● Cognition and content: fragmentary information – no integration → confusion ● Navigation and structure: hyperlinks move across structure – where am I? ⇒ search and delivery methods are important

6.2.3. Hypermedia

Hypermedia not only corners around texts, but also covers in Videos, Photographs, Audios, Links, and Applications, etc.

6.3. Web Usability considerations

6.3.1. Usability for web design

- Many usability principles applicable to Web UI design
  - Iterative design, user-centered design
  - Modeling approaches
  - Development principles, rules and standards
  - Information layout and perceptual factors
  - Evaluation methods
- But Web UI has restrictions different from traditional UI design

Traditional Desktop-Based UI vs. Web-based UI

Traditional Desktop-Based UI	**Web-based UI**
Users are more loyal	You cannot control users. People will leave if your website is difficult to use
You control each pixel	You give up control to meet client/server hardware/software needs
You know for what system you are designing	You could end up on WebTV!
Designers can control where the user goes and when	Users have more control of their navigation, and may only stay for a few moments!
You are part of an enclosed experience	Obviously not the same for Web design!

Web is distributed into: Web servers, web browsers, web pages. Diverse devices. Diverse users.

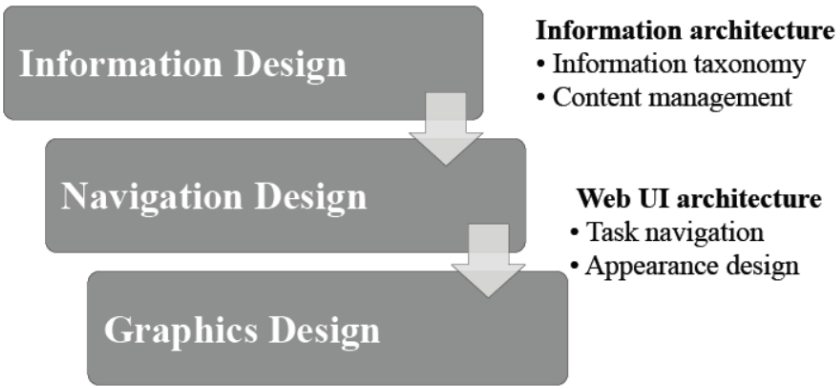
6.3.2. Specific web usability considerations

- Network issues:
  - Bandwidth: how much information per second
  - Latency: how long it takes (delay)
  - Jitter: how consistent is the delay

    - Reliability: some messages are lost; need to be resent → increases jitter
    - Connection set-up: need to "handshake" to start
    - Design implications
        - Bandwidth: Consider download time. Different media qualities: different formats
        - Connection time: Big files vs. small multiple files
        - Latency: Giving feedback (e.g. using progress indicator n Breaking large service into small pieces)
- Architectures:
    - Information architecture: Information taxonomy, Content management
    - Web UI architecture: Task navigation, Appearance design

6.3.3. Architectures

6.3.3.1. Web development architecture



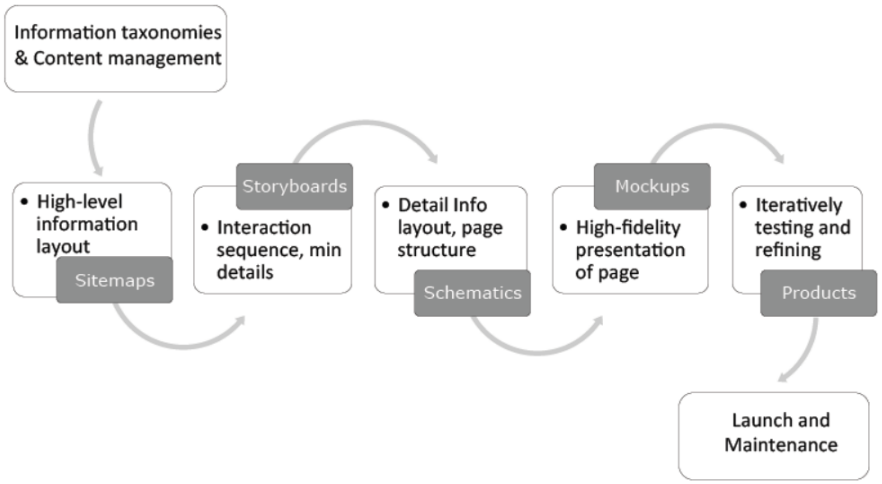
6.3.3.2. Information architecture

Information architecture is a combination of organizing a site's content into categories and creating an interface to support those categories

Information taxonomy is the core of information architecture: refers to the effective structuring of content within a defined scope to facilitate easy and accurate access

- Presents Web content with logically grouped topics or categories so a site visitor can navigate and locate information easily
- Helps better Web development: content management: design process: search process
- Even the most efficient search engine cannot completely overcome problems caused by: poorly conceived information taxonomy, or completely absent information taxonomy

6.3.3.3. Web UI architecture



Sitemaps:

- Site maps organized hierarchically: used as planning tool, provide the designer a high-level view of entire site
- Use textural labels, diagrams, flowcharts.
- Pay special attention on the paths of navigation through the information space
- Tools used n Hand-drawn (small site), PowerPoint, Visio or Illustrator
- Information taxonomy is used as input to design site maps

Storyboards:

- Provide a concrete presentation of interaction sequence for each page, item, or subtopic
- Tools used: Hand-drawn (small site),PowerPoint, Visio or Illustrator

Schematics:

- Provide detailed navigation content of pages, page layout and structures
- Consider graphics elements

Mockups and products: Detailed and correct presentation of content

6.4. Guidelines and Tips on Web UI design

Top 10 Mistakes in Web Design (from Jakob Nielsen, 2011)

1. Bad Search
2. PDF Files for Online Reading
3. Not Changing the Color of Visited Links
4. Non-Scannable Text
5. Fixed Font Size
6. Page Titles With Low Search Engine Visibility
7. Anything That Looks Like an Advertisement
8. Violating Design Conventions
9. Opening New Browser Windows
10. Not Answering Users' Questions

Ten Good Deeds in Web Design (Jakob Nielsen)

1. Place the name and logo on every page and make a link
2. Provide search if the site has more than 100 pages
3. Write straightforward and simple headlines and page titles
4. Structure the page to facilitate scanning
5. Use hypertext to structure the content space into a several subtopics
6. Use product photos, but avoid cluttered and bloated pages
7. Use relevance-enhanced image reduction for small images
8. Use link titles to provide users with a preview of where will take then
9. Ensure all important pages are accessible for users with disabilities
10. Do the same as everybody else

Some Web Design Guidelines

- Site tourists: you cannot control the user
- Loyal visitors: you almost want to say "Please bookmark me"
- Avoid frames
- People will not "hang around" if they see the site is not maintained current
- Users are impatient
- Reduce scrolling, especially at the home page
- Avoid animation unless it has a purpose, like showing how a game is played
- Don’t animate forever
- Don’t do “Enter Here”
- Limit the number of colors used
- A site needs to make clear what it is supposed to do and how it is organized
- Don't match other media, e.g. brochures, TV ad, and push that onto the web page. Do the web first
- Link wisely, not everything. Also, don't just say "click here" without a highlight or reference
- Tell people where you are. Provide navigation buttons and/or site maps.
- On-line surveys & forms need smooth interaction, "clear" button, easy to back up and correct, pull down choices, etc.
- Strong metaphor can be "over cute"
- Spend a moment on URL design
- White background preferred with easy-to-read (dark) lettering

Top 10 Guidelines for Homepage

1. Include a One-Sentence Tagline
2. Write a Window Title with Good Visibility in Search Engines and Bookmark Lists
3. Group all Corporate Information in One Distinct Area
4. Emphasize the Site's Top High-Priority Tasks
5. Include a Search Input Box
6. Show Examples of Real Site Content
7. Begin Link Names with the Most Important Keyword
8. Offer Easy Access to Recent Homepage Features
9. Don't Over-Format Critical Content, Such as Navigation Areas

10. Use Meaningful Graphics

Tips on Web writing:

- People rarely read Web pages word by word: instead, they scan the page, picking out individual words and sentences
- Good Web pages often employ scannable text: Highlight keywords, Use bulleted lists, Have one idea per paragraph

Tips on Hyperlink and Navigation:

Apply HCI rules: Where & what of navigation in hyperspace

- Where are you?
- Where are you going or what will happen?
- Where have you been or what has been done?
- What can you do now?

Tip on link: informative title ⇒ Basic Rule – helping users to make an informed decision before they click.

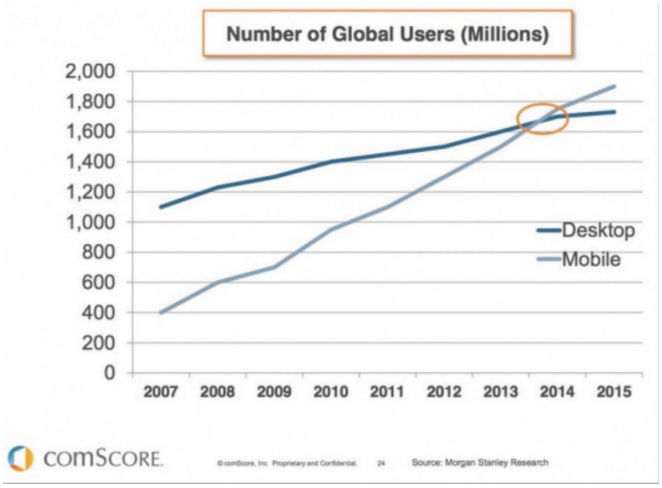
Tips on graphical elements: Apply general UI principles

- Color should be used with care
    - Don’t use many colors in text unless it is a link
    - Don’t use non-standard link colors
    - White background preferred with easy-to-read lettering
- Font should be used carefully
    - Don’t use too small fonts, non-standard, frozen size
    - Avoid clutter and low contrast and resolution
- Image/video/animation
    - Avoid “eye candy” unless it supports a message
    - Motion attracts attention. Useful if important, otherwise distracting
    - Optimize pages for loading speed

7. Mobile UI

7.1. Prevalence and Importance of Mobile Computing

7.1.1. Prevalence of mobile devices



- More than just “Talking”
  - Mobile phones with significantly increased capability
    - Greater on-board intelligence (software & hardware)
    - Larger on-device data storage
    - Better connectivity to networks & to other devices
    - Larger, clearer, colorful screens
    - Easier input of data (via pens, keys, touch, voice…)
  - With easier programming of the on-board intelligence
  - And allow users to easily use the intelligence
- Waves of mobile computing: Portability, Miniaturization, Connectivity, Convergence, Divergence, Apps, Digital Ecosystems
- Dimensions of meeting extra:
  - Communication: voice, message, image, video, VoIP…
  - Information access: location, navigation, presence, education
  - Entertainment: audio, graphics, video, multimedia…
  - Gaming capabilities: multi-player, gaming, mobile access to online worlds
  - Personal/Business productivity: PDA, scheduling, email, meeting…
  - Electronic commerce: shopping, electronic wallet, banking…

7.1.2. Mobile computing

- Mobile computing: is a generic term describing one’s ability to use computing technology while moving, as opposed to PCs, which are only practical for use while deployed in a stationary configuration.
- Types: Wearable computer, Ultra-Mobile PC (UMPC), Handheld computer (PDA), Smart phone, Tablet computer
- >50% of Internet access is by non-PCs

- Diverse form factors: size, shape, interaction elements…
- Diverse set of functions: multimedia, game, mini-computer…
- Users' attachment to them: wearable, handheld, RFID, implants…
- Hot research area in HCI (Human Computer Interaction)

7.2. Mobile UI Design

7.2.1. Mobile HCI

- HCI and Mobile 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"
- Mobile HCI deals with the above issues for mobile devices such as PDAs, mobile phones/pagers, wrist watches, memo devices, GPS, small size embedded systems, etc. … and has to concentrate on more restricted issues raised by mobility.

7.2.2. Mobile is really different

- They are mobile, otherwise, no one will carry
- More connectivity: wireless, Bluetooth, Infrared, WLAN
- More sophisticated expectations: user patterns, personalization, economics…
- More advanced in: hardware features, software features and necessities

7.2.3. Mobile UI restrictions and constraints

- Restrictions
    - Claim: people want PC functionality but they don't want the PC's overhead
    - Cannot simply make the screen bigger and add a keyboard, because no one will carry it!
    - Cannot use a too fast processor or add more memory, because they use too much power.
    - Hard to interact, because mobile phone has no screen when it is at your ear
- Constraints
    - Limited input and output
        - Keypad, small screen size, low resolution
        - Data entry is error prone and slow
        - User's environment: glare
    - Slow processor, small memory, slow network connection as compared to desktop
    - Limited power consumption

    → We need imaginative ways of overcoming those restricted issues

- Design goal for Mobile HCI is still the same: designing for **maximum usability**

7.2.4. Golden Rules

- Understand computers: limitations, capacities, tools, platforms
- Understand people: psychological, social aspects, human error
- Understand tasks: goals, constraints
- Understand user's interaction

7.2.5. Understand users and tasks

- What features do users use?
- Where do they go?
- What actions do they take?
- When do they use?
- How much time do they spend on the app?
- How much time do they spend on tasks?

7.2.6. User interface design components

- Metaphors: the fundamental concepts the UI communicates
- Mental models: the ideas trying to be conveyed by the UI
- Navigation: movement through the mental models
- Interaction: areas where the user is allowed to have input into the system and output provided

7.2.7. Discovering user's needs

- Follow user's through role play
- What current artifacts are they using? E.g. a joystick, stylus, keyboard, etc.
- What is the current HW/SW capability available? E.g., on-board camera, motion sensor

7.2.8. Input solutions

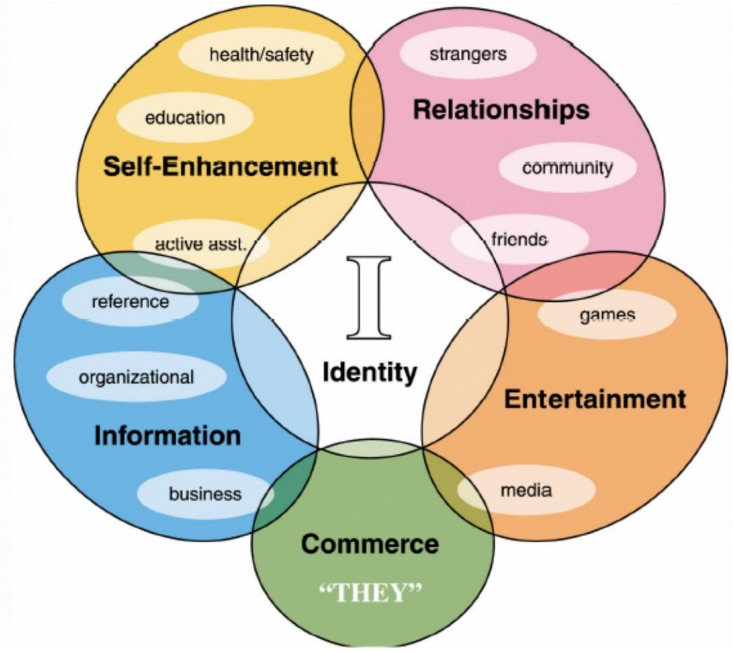
- Small physical keypad, on-board or attached keyboard
- Ease of navigation
- Touch screen
- Alternative ways:
    - Handwriting recognition:
        - Solution as writing is natural and well learned
        - Better recognition rate
        - Fast training to learn a new alphabet or put up with errors
        - Sometimes, processor too slow for handwriting recognition
    - Speech recognition and synthesis
        - Look natural choices for mobile interaction
        - Problems, speaker dependent, domain dependent, recognition rates, noise environment, out-of-vocabulary words, cognitive burden
        - Deep learning
        - Technologies: Apple Siri, Amazon Alexa, Google Now, Samsung Bixby, Microsoft Cortana
    - Gestural interfaces:
        - Micro-gestures
        - Device-based gesture, e.g. Apple's multi-touchpad
        - Embodied interaction, e.g. EyeToy (PlayStation 2)
    - Sensor:
        - Tilted, gyro, acceleration sensors for orientation control
        - Camera sensor with computer vision
        - Mobile augmented reality (MAR)

7.2.9. Output solutions

- How can problems with output be solved?
  - Sound output (speech and non-speech)
  - Better display color, resolution
  - Screen sizes, screen resolutions
- VR/AR:
  - Virtual touch screens (VTS)
  - Metaverse
  - Products: MS HoloLens, Facebook's Oculus, Samsung's Gear VR
- Context-aware, location-based, activity-based computing
  - Take into account the user's state and surroundings
  - Enable the user's mobile computer or environment to adapt accordingly

7.3. Mobile Usage Space

7.3.1. Analytical framework for usage space



7.3.2. Information space

- The device's content of static reference information
- Examples: Contact information, Class schedule, Blood glucose history, Tasks/to-do list

7.3.3. Self enhancement space

- How the device extends the user's normal capabilities
- Examples: Alarm clock program reminds you to be on time to meetings, Voice recording program allows you to take quick notes of forgetful things

7.3.4. Relationships space

- How the device extends the user's relationship with others, beyond normal phone to phone conversation
- Examples: Community programs helps chatting with others in your area, Social networking

7.3.5. Entertainment space

- How the device's programs give enjoyment and stress relief
- Examples: Games, Karaoke, Music

7.3.6. M-commerce space

- How the device extends the user's commercial capability
- From the business world to individual
- Examples: E-coupons if in nearby area of shop, Mobile banking, Trade stock, Online Shopping

7.3.7. Identity space

- How the device becomes a part of the identity of the user
- Over time the device gains knowledge about the user
- Remembering complex behavior patterns

  → Someday your mobile phone knows you more than you to yourself

7.4. Mobile Operating System

7.4.1. Introduction

- Desktop OS is just too big for mobile computers
- Major Mobile OS
  - Palm OS (Palm Computing)
  - BlackBerry OS (Research In Motion)
  - Windows CE / Windows Mobile (Microsoft)
  - iPhone OS (Apple Inc)
  - Android, open system (Google)
  - Symbian OS, open system (Symbian Ltd, Nokia)
  - Maemo OS, open system (Nokia)
  - MeeGO, open system (Intel, Nokia)

7.4.2. The role of an OS

- Tame the underlying technological complexities, abstract out the "accidental differences"
- Allow myriad software components to co-exist & collaborate: present-day and future components
- Make it look simple, the end user should find it straightforward & natural
- A good OS works very hard on behalf of the user, even though the user is unaware of the OS

7.4.3. Mobile platforms and tools

- iOS

    - Apple mobile platform: iPhone, iPod Touch, iPad
    - Four layers: Core OS layer, Core services layer, Media layer, Cocoa touch layer
    - Includes core applications: email, calendar, photos, camera..
    - Xcode programming environment, SDKs
- Android
    - Google mobile platform
    - A software stack for mobile devices: OS, middleware, tools, apps.
    - Includes core applications: email, browser, maps, SMS
    - Application Framework, Apps and GUI builder
    - Linux kernel
    - Java programming language
- Windows Mobile platform
    - Pocket PC, Windows CE, Window Mobile, Windows .NET
    - Windows GUI builder: integral components
    - Object oriented programming environment: VC++ IDE, .NET Compact Framework, SDKs, Emulator, Tools

8. Graphic Design

8.1. Graphic Design

- Graphic design: refers to artistic and professional disciplines which focus on visual communication and presentation
- In UI, it refers the look and feel portion of an interface, initially encountered by users, eye catching
- Conveys an impression, mood, beauty, etc.
- Facilitates finishing the task at hand
- Suggests trust

8.2. Affordance

- "Perceived and actual properties of a thing that determine how the thing could be used" – Don Norman
- An affordance is a quality of an object that allows an individual to perform an action: Related to look and feel of an object
- Two types of affordance
    - Perceived affordance: design invites people to take possible actions
    - Actual affordance: the actual actionable properties of the product

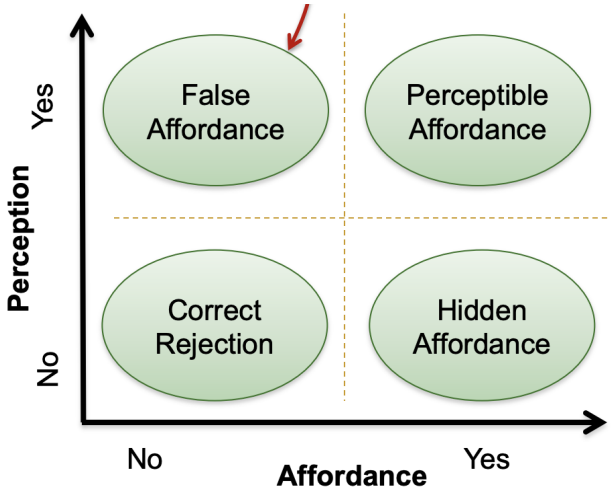
8.2.1. Perceived affordance

- Is the perceived properties of the object that suggest how we could use it, e.g. buttons for pressing, switches for toggling
- Poor affordances: Pull or Push? (refers to a door)
- Affordance can be dependent on: Experience, Knowledge, Culture of users
- Door handle – a classic affordance problem
    - Vertical bar: pull

    - Horizontal bar or flat surface: push
    - Knob or handle: grab and twist
    - Bar or handle location: door opens to the left or right
- Affordance can be dependent on the: Context, Layout, Locations of the objects placed
- Perceived affordance (perception) can differ from real affordance
    - The paper-made chair has a perceived affordance for sitting
    - But it doesn't actually afford sitting: it collapses under your weight

8.2.2. Perceived and actual affordance

- Affordance
    - Perceived affordance (perception): design invites people to take possible actions
    - Actual affordance: the actual actionable properties of the product
- Problems occur when
    - These are not the same
    - People's perceptions are not what the designer expects
- A false affordance exists when there is no action offered but the information that specifies it is. E.g. paper-made chairs



8.2.3. Signifiers

- An affordance is a quality of an object that allows an individual to perform an action
- A signifier is a thing that communicates the affordance of an object
- A signifier can be labels, instructions, shapes, colors, layouts, sound, videos, animations, mouse shapes, etc.

8.3. Affordance in HCI design

- In HCI, interfaces are virtual and do not have affordances like physical objects
- It does not make sense to talk about interfaces in terms of 'real' affordances - Norman
- Interfaces are better conceptualized as 'perceived' affordances
- In graphical, screen-based interfaces, the designer can control over perceived affordances
    - Display screen, pointing device, selection buttons, keyboard

    - Actions including touching, pointing, looking, clicking on every pixel of the display
- GUI design
    - Perception only through visuals
    - Designer creates appropriate visual affordances via: familiar idioms, metaphors
- Cognitive considerations: Perceived affordances, Constraints, Feedback, Mapping, Mental Models, Conceptual models

8.4. Visible constraints

- Limitations of the actions possible perceived from the object's appearance. Provides people with a range of usage possibilities
- The more constraints, the less opportunity for error, particularly important for managing user input
- Benefits
    - Restricting user actions to valid actions
    - Helps prevent from selecting incorrect options
    - Eliminate need for perfect knowledge
    - Recognition over recall
- The more constraints, the less opportunity for error
    - But too much constraint, less flexible and less efficient
    - E.g., Expert users prefer typing than clicking to select choices
- Three main types (Norman, 1999): physical, cultural, logical
- Physical constraints
    - Refer to the way physical objects restrict the movement of things
    - E.g. How many ways can you insert a CD or DVD disk into a computer?
- Logical constraints
    - Exploit people's everyday common sense reasoning about the way the world works
    - E.g., the logical relationship between physical layout of a device and the way it works
- Cultural constraints
    - Groups of people learn idioms: red = danger, green = go
    - But these differ in different places
        - Red is not at all danger (preferable) in many countries
        - Light switches: America: down is off, Britain: down is on
        - Faucets: America: counter-clockwise is on

8.5. Graphic design philosophies

- Aesthetic appeal does not automatically confer usability
- UI design must balance the meaning of its visual elements that conform the mental model of operation
- Preferences
    - Simple and natural user's "language"
    - Economy of visual elements
    - Clean, well organized
    - Less is more

8.6. Guidelines for good graphic design

- Basic principles
    - Metaphor
    - Simplicity and Clarity
    - Consistency
    - Organization/Alignment/Proximity/Grid
    - Legibility and readability
    - Color/Contrast

8.6.1. Simplicity

- “Keep it simple, stupid.” (KISS)
- “Less is more.”
- “When in doubt, leave it out.”
- Every element in an interface should have a reason for being there, so make that reason clear too
- Techniques
    - Reduction
        - Decide what essentially needs to be conveyed by the design
        - Examines every element if it serves essential purposes
        - Remove inessential elements
    - Regularity
        - Use a regular pattern, minimize the unnecessary differences between elements with regularity. Elements provided by the operating system with which the users are already familiar
        - Limit inessential variation among elements. Use the same font, color, line width, dimensions, orientation for multiple elements. Irregularities in your design will be magnified in the user’s eyes and assigned meaning and significance
    - Combine elements
        - Let elements play multiple roles, e.g.Scroll bar, Title bar
- White space
    - Leads the eye, provides symmetry and balance through its use
    - Allows eye to rest between elements of activity
    - Used to promote simplicity, elegance, class, refinement
        - Margins to draw eye around design
        - Don’t crowd controls together
    - Put labels in the left margin with white space leading and highlighting them
    - Put labels on left of the controls, not above
    - Use placeholders

8.6.2. Consistency

- Similar things should work similarly
- Different things should look different
- Consistency types
    - Internal: within the system, elements follow same conventions and rules
    - External: across different systems, follows platform and interface style conventions

- Metaphorical: reflecting real-world objects. A print icon is a metaphor of the printer

8.6.3. Organization

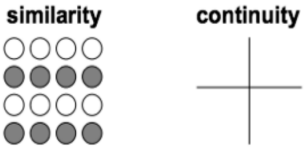
8.6.3.1. Grid system

- Grid is an essential tool for graphical design. A uniform grid (a grid with equal-width columns) is one effective way to achieve both alignment and balance
- Horizontal and vertical lines to locate window components, aligns related components
- Consistency: location, format, element repetition

8.6.3.2. The Gestalt principles of grouping

- Discovered in the 1920s by the Gestalt school of psychologists
- Describe how early visual processing groups elements in the visual field into larger wholes
- **Proximity**: elements that are closer to each other are more likely to be grouped together
  - Example: You see four vertical columns of circles, because the circles are closer vertically than they are horizontally.

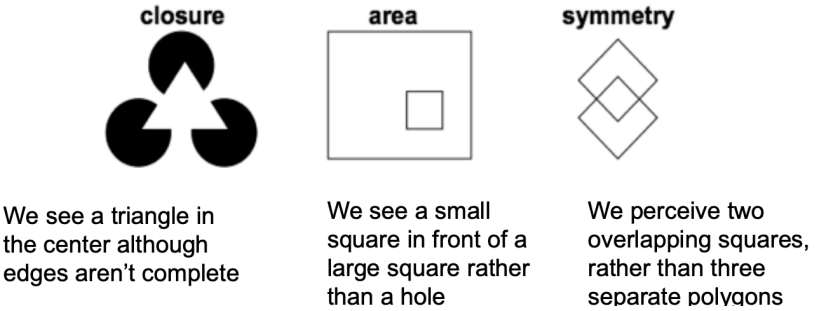
- **Similarity**: Elements with similar attributes are more likely to be grouped
- **Continuity**: The eye expects to see a contour as a continuous object



You see four rows of circles, because the circles are more alike horizontally than they are vertically

You primarily perceive this as two crossing lines, rather than as four lines meeting at a point, or two right angles sharing a vertex

- **Closure**: The eye tends to perceive complete, closed figures, even when lines are missing
- **Area**: When two elements overlap, the smaller one will be interpreted as a figure in front of the larger ground
- **Symmetry**: The eye prefers explanations with greater symmetry



8.6.3.3. Alignment

- Aligning elements horizontally and vertically → Improve the simplicity of a design
- Bad alignment: no flow, causing the eyes to zig-zag around the screen as the user attempts to locate a field of interest
- Poor contrast: cannot distinguish colored labels from editable fields
- Poor repetition: buttons do not look like buttons
- Poor explicit structure: blocks compete with alignment

8.6.3.4. Balance and symmetry

- Choose an axis (usually vertical)
- Distribute elements equally around the axis. Equalize both mass and extend

8.6.3.5. Economy of visual elements

- Less is more - unless more is more…
- Minimize number of controls
- Include only those that are necessary. E.g. 3D effects
- Appropriately arrange the size, layout of controls, e.g., text fields, combo boxes, checkboxes
- Minimize clutter, so information is not hidden
- Tabs: excellent means for factoring related items, but can be overdone

8.6.3.6. Legibility and readability

- Characters, symbols, graphical elements should be easily noticeable and distinguishable
- Text that is in horizontal orientation is difficult to read. Don't try too hard to harm your neck

8.6.3.7. Imagery

- Signs, icons, symbols: right choice within spectrum from concrete to abstract
- Meaningful icon design is hard! Except for most familiar, always label them
- Consistent and relevant image use: identifies situations, offerings, etc.
- Avoid 'eye candy' unless it supports a message
- Motion attracts attention: useful if important, otherwise distracting

8.6.3.8. Color and contrast

- Contrast encodes information along visual dimensions

10. Interaction Styles

10.1. Interaction Styles

10.1.1. Dialog types

- Dialog type = Interaction style
- Menu selection: Discriminator of options, recognition over recall
- Form Fill-in: Integrator of data values, higher skill, more flexible
- Question & Answer: Series of values, easy for untrained
- Function Keys: Hardware, software or labels
- Command Language: Naming and syntax issues
- Query Language: Specialized command language
- Natural Language: Most general purpose for untrained users
- Direct Manipulation: Physical properties reflected in objects
- Virtual Reality, Multimedia & Animation: Complete, realistic, interactive spaces
- Combinations of the above

10.1.2. Menu

- Advantages of menu
    - Self-explanatory: reduces need for manuals, requires little or no training, makes both semantics and syntax explicit
    - Requires little memory: recognition vs. recall
    - Fewer keystrokes: less opportunity for user input error
    - Easy error handling: only limited valid inputs at any point
    - Enhancements are visible
- Disadvantages of menu
    - Inefficient for experts and high frequency users
    - Inflexible: system controlled, forced choice
    - Take up screen ‘real estate’: only limited valid inputs at any point
    - Real estate = pixels on screen
- When to use menu?
    - Menu is most appropriate for
        - Knowledge and experience: low typing skill, little system experience, low task experience, low application experience, frequent use of other systems, ow computer literacy
        - Job and task characteristics: low frequency of use, little or no training, discretionary use, high turnover rate, low task importance but high task structure
- Design guidelines: structure
    - Create logical, distinctive categories with clear meanings
    - Menu items should be brief, consistent in grammatical style and placement, and matched with corresponding menu titles

    - Minimize menu hierarchy depth at the expense of breadth. If going deep → slow response time
- Design guidelines: ordering
    - Order menu items according to functional groups, frequency of use, order of use and/or alphabetical order
- Design guidelines: navigation
    - Establish conventions for menu design and apply them consistently on all menu screens
- Menu design guideline
    - Prefer broad and shallow menus to narrow and deep ones
    - Use items as titles for sub trees
    - Group items meaningfully
    - Use brief items, begin with the keyword
    - Use consistent grammar, layout, terminology
    - Allow type ahead, jump ahead, or other shortcuts
    - Consider: online help, optimal response time, display rate, screen size

10.1.3. Fill-in forms

- Advantages of fill-in forms
    - Self-explanatory: reduces need for manuals, requires little or no training, makes both semantics and syntax explicit
    - Requires little memory: recognition vs. recall
    - Efficient use of screen “real-estate”
    - Accommodates parameters with many possible values
    - Provide context
- Disadvantages of fill-in forms
    - Assumes knowledge of valid inputs
    - Assumes typing skills and knowledge of special keys (e.g. TAB, RETURN, BACKSPACE)
    - Creates opportunities for user error
- When to use fill-in forms?
    - Fill-in form is most appropriate for
        - Knowledge and experience: moderate to high typing skill, little to moderate system experience, moderate to high task experience, low to moderate application experience, moderate to frequent use of other systems, moderate to high computer literacy
- Guidelines for fill-in forms
    - Meaningful title
    - Comprehensible instructions
    - Logical grouping and sequencing of fields
    - Familiar field labels
    - Consistent terminology and abbreviations
    - Visible space and boundaries for data-entry fields
    - Convenient cursor movement
    - Error correction for individual characters and entire fields
    - Error prevention where possible
    - Error messages for unacceptable values
    - Marking of optional fields

    - Explanatory messages for fields

10.1.4. Direct manipulation

- Visual representation of the “world of actions”
    - Objects and actions are shown.
    - Taps analogical reasoning.
- Rapid, incremental, and reversible actions.
- Replace typing with pointing/selecting.
- Immediate visibility of results of actions.
- Examples: Flight simulator, Display-based text editor, Personnel system, Database query-by-example, Video games, CAD, Programming of industrial robots, Office automation systems, Windowing systems, Visual programming, Touch-screen kiosk, Touch-screen phones
- Advantages of direct manipulation
    - Easy to learn and remember
    - Direct, intuitive, WYSIWYG
    - Flexible, easily reversible actions
    - Provides context and instant visual feedback
    - Exploits human use of visual and spatial cues
    - Low typing requirements and visual feedback
    - Less opportunity for user error
- Disadvantages of direct manipulation
    - Inefficient for high frequency expert users
    - May be difficult to design recognizable icons for many objects and actions
    - Icons take may more screen real estate than words
- When to use direct manipulation?
    - Knowledge and experience: low typing skill, low system experience, low task experience, high frequency of use of other systems, low computer literacy
    - Job and task characteristics: low frequency of use, little or no training, discretionary use, high turnover rate, low task importance, low task structure
- Direct manipulation design guidelines:
    - Provide alternative interface for high frequency and expert users
    - Choose a consistent icon design scheme: depict “before and after”, tool, action
    - Accompany icons with names
    - Provide visual feedback for position selection and movement, and physical feedback for modes

10.1.5. Command language

- Interact with a computer using text or voice commands. Rely on naming and syntax
- Examples: Commands on DOS, Commands on UNIX
- Advantages
    - Flexibility
    - Supports user initiative
    - Appeals to “power users”
    - Potentially rapid for complex tasks
    - Supports macro capability

- Disadvantages
    - Requires training and memorization
    - Difficult to retain
    - Poor error handling
- Command language guidelines
    - Choose meaningful, specific, distinctive names
    - Support consistent abbreviation rules, prefer truncation to one letter
    - Offer frequent users the capability to create macros
    - Limit number of commands and ways of accomplishing a task
    - Consider command menus on high-speed displays

10.1.6. Function keys

- Dedicated function keys: F1, Esc, Window key, etc.
- Soft function keys (labels on screen)
    - Self-explanatory. Easy to use. Flexible. Requires little human memory.
    - Little or no on screen real estate needed. Limited typing requirement
- Concerns
    - Limited number of function keys exist
    - Application-specific
    - Inconsistency among applications: Ctrl + F on Office and Outlook
- Guidelines
    - Gray-out non-applicable functions
    - Combination of keys. E.g. Ctrl + Alt + Del, Ctrl + C
        - Keys should be easy to reach
        - Consistent grammar. E.g., Ctrl for special, Alt for alternative pointing methods

10.1.7. Question and answer style

- Combines some features of menus and fill-in forms
- User is posed with a single question, e.g., Wizard dialog, Prompt for missing parameters
- Appropriate for lowly-motivated, less-experienced users. Requires little training

10.1.8. Natural language interaction style

- Interact with computer using natural spoken or written language
- Examples
    - Voice command for GPS to find gas stations, food, directions, etc.
    - Google search voice command box
    - Chatbot, Amazon’s Alexa, Google’s Home
- When to use?
    - NLI may work best for
        - Users who are knowledgeable about the task domain
        - Users with moderate computer skills
        - Limited access to other interaction styles (voice used while driving), disabled people (those cannot type)

10.2. Comparison of interaction styles

User Psychology	Menu	Fill-in Forms	Question & Answer	Command Language
Attitude	Negative	Negative Neutral	Negative	Positive
Motivation	Low	Low Moderate	Low	High

Knowledge & Experience	Menu	Fill-in Forms	Question & Answer	Command Language
Typing Skill	Low	Moderate High	Moderate High	Moderate High
System Experience	Low	Low Moderate	Low Moderate	High
Task Experience	Low	Moderate High	Low	High
Application Experience	Low	Low Moderate	Moderate	High
Use of Other Systems	Frequent	Moderate Frequent	Moderate Frequent	Infrequent
Computer Literacy	Low	Moderate High	Low	High

User Psychology	Function Keys	Direct Manipulation	Natural Language
Attitude	Negative	Negative	Negative
Motivation	Low	Low	Low

Knowledge & Experience	Function Keys	Direct Manipulation	Natural Language
Typing Skill	Low	Low	High
System Experience	Low	Low	Low
Task Experience	Moderate High	Low	High
Application Experience	Moderate	Low	Low
Use of Other Systems	Low	High	High
Computer Literacy	Moderate High	Low	Low

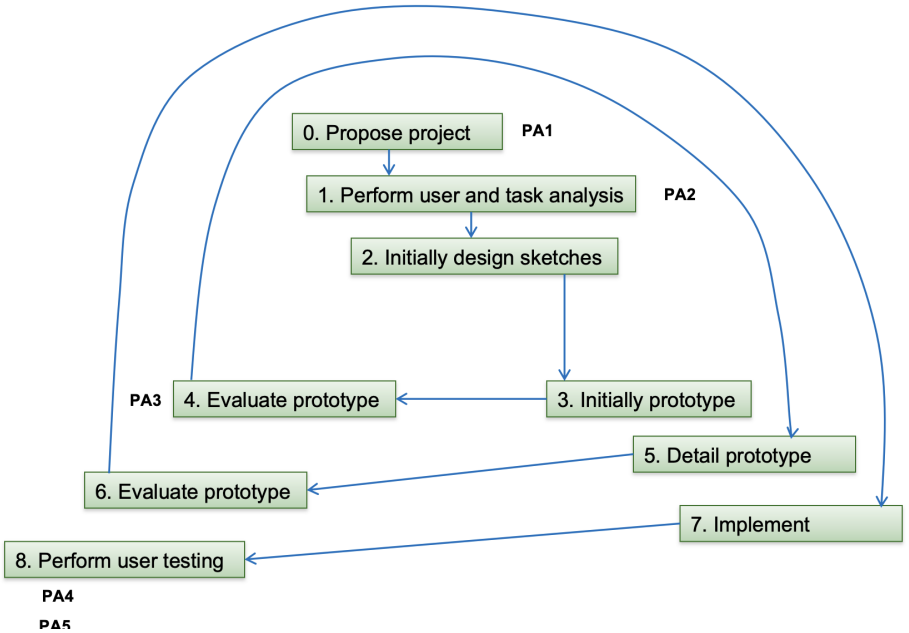
Task Characteristics	Menu	Fill-in Forms	Question & Answer	Command Language
Frequency of Use	Low	Moderate High	Low	High
Primary Training	Little or none	Little or None	Little or None	Formal
System Use	Discretionary	Discretionary	Discretionary	Mandatory
Turnover Rate	High	Low Moderate	High	Low
Other Systems		Paper forms		
Task Importance	Low	Moderate	Low	High
Task Structure	High	High	High	Low

Task Characteristics	Function Keys	Direct Manipulation	Natural Language
Frequency of Use	Low	Low	Low
Primary Training	Little or none	Little or none	Little or none
System Use	Discretionary	Discretionary	Discretionary
Turnover Rate	Moderate	High	High
Other Systems			
Task Importance	Moderate	Low	Low
Task Structure	Low Moderate	Low	Low

11. UI Evaluation

11.1. Overview

11.1.1. Projects Process



11.1.2. Why, what, where, when to evaluate

- Iterative design and evaluation is a continuous process that examines
- Why: check that users can use the product and that they like it
- What: a conceptual design, early prototypes and later, more complete prototypes
- Where: in natural and team settings
- When: throughout design process; finished products can be evaluated to collect information to inform new products

11.1.3. Throughout the Usability Engineering Lifecycle

- Conceptual design: evaluating initial concept design ideas with the users
- Early prototype design: evaluating with the users
    - Does system behavior match the user's task requirements?
    - Are there specific problems with the design?
- Complete prototype design: acceptance testing to verify the system meets expected user performance criteria

11.1.4. Purpose of evaluation

- Identify specific problems in UI: mismatches between design and final product, between design and real users' behaviors
- Assess the effect of interaction: usability and user satisfaction
- Explore and understand system's functionality: understand real users' interactions

11.1.5. Methods of evaluation

- Heuristic evaluation: Done by experts
- User tests: done by users. Field studies, Controlled experiment, Formative evaluation

11.2. Heuristic evaluation

11.2.1. Usability heuristics/guidelines

- Many heuristics to base on
    - Nielsen's 10 heuristics
    - Norman's rules from Design of Everyday Things
    - Shneiderman's Eight Golden Rules
- Heuristics help
    - Designers to choose design alternatives
    - Help evaluators find problems and assess interfaces (hence Heuristics Evaluation)
- UI design principles we've studied
    - Learnability/Memorability – L. Visibility – V. Simplicity – S
    - User control – UC. Error handling – ER. Efficiency – E. Graphic design – GD
- Nielsen's heuristics
    - Match the real world (L). Consistency and standards (L). Help and documentation (L)
    - User control and freedom (UC). Visibility of system status (V). Flexible and efficiency (EF)
    - Error prevention (ER). Recognition, not recall (ER). Error reporting, diagnosis, and recovery (ER)
    - Aesthetic and minimalist design (GD)
- Norman's rules: Affordances (L). Natural mapping (L). Visibility (V). Feedback (S)
- Shneiderman's eight golden rules
    - Consistency (L). Shortcuts (EF). Feedback (V). Dialog closure (V). Simple error handling (ER)
    - Reversible actions (UC). Put users in control (UC). Reduce short-term memory load (ER)
- Heuristic evaluation
    - Invented and evaluated by Jakob Nielsen. Cheaper and more effective than alternative methods
    - An inspection technique often done by experts
    - Basic steps
        - Review/inspect UI
        - Compare it against certain heuristics
        - Document usability problems
        - Explain and justify problems against heuristics

11.2.2. Guidelines for heuristic evaluation

- Justify every problem with heuristics (one or many)
    - "Dialog A has button 'Hủy' while the same button on dialog B is labeled 'Đóng'" (Consistency)
    - Don't just say "I don't like the layout"
- List every problem, go through the interface multiple times
- Don't have to limit to the 10 Nielsen's heuristics
- Not all heuristics are applicable for a user interface
- Use multiple evaluators

    - More people find more and different problems than one person
    - Problems can be duplicated
    - Nielsen recommends 3-5 evaluators
- Use heuristic evaluation in addition to user testing
    - Each method finds different problems
    - Heuristic evaluation is **cheaper**

11.2.3. Evaluating prototypes

- Heuristic evaluation can also be used to evaluate
    - Sketches
    - Paper/computer prototypes
    - Demo code
- Certain heuristics are not applicable: Feedback, “Missing-element” problems, Help & documentation

11.2.4. Formal evaluation process

- Training
    - Meeting to introduce the application and its user interface
    - Explain user population, domain, scenarios
- Evaluation: evaluators work independent to record problems and generate report
- Severity rating: evaluators provide severity levels for problems found
- Briefing: evaluators and design team discuss results and suggest solutions

11.2.5. Severity level

- Different scales can be provided, such as
    - Cosmetic or trivial: not important, low priority
    - Minor: not serious, low priority
    - Major: quite serious, high priority
    - Critical: serious, high priority
- Factors to consider for rating
    - Frequency: how common the problem is found?
    - Impact: hard to resolve?
    - Persistence: repeated?

11.2.6. Writing heuristic evaluations

- Evaluators must provide clear descriptions of problems to reduce time for both designers and evaluators
- Include both positive comments and criticisms, emphasize more on criticisms and recommendations
- Don’t be unnecessarily tough
- Be specific. Don’t “text is unreadable”. Be “text has poor contrast”
- Evaluation report can include: problem title, problem description, heuristic, severity, recommendations and screenshots (if any)

11.2.7. Cognitive walkthrough

- Is an inspection technique that focuses on learnability

- Inputs: prototype, task, sequence of actions, user analysis
- Evaluator asks questions while following sequence of actions
    - Will users find the action on the interface?
    - Will users know what to do?

11.3. Field Studies

11.3.1. Overview

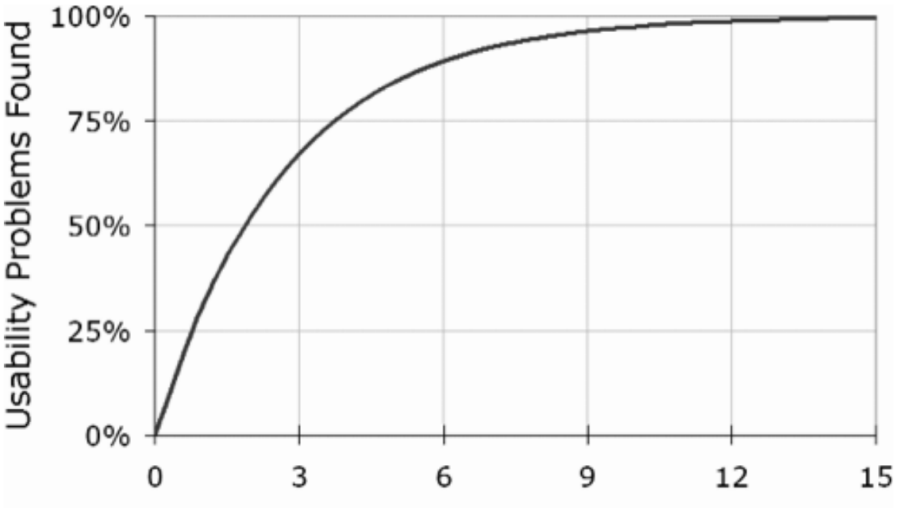
- Real use environments
    - Observe effect occurs in realistic setting
    - Can't tell how good UI is until the users use the products and that they satisfy
    - Hard to predict what real users will do. Problems identified are those that are really raised when the user uses the software
    - Usability could be measured during the test by calculating the user performance
- Problems: cost/time consuming, may not generalize
- Procedure
    - Prepare test proposal
    - Choose participants
    - Select properties to be tested
    - Perform the tests
    - Measure the test results

11.3.1.1. Prepare test proposal

- Test objective. Test participants. Test environment and materials
- Test methodology to be used: detailed descriptions of systems/tasks/properties being tested
- Test measures: test data to be collected and analyzed

11.3.1.2. Choose participants

- Representative of target users
    - The users must be those targeted by the evaluated product
    - Relevant tasks, job-specific knowledge
- Approximate if needed:
    - System intended for doctors → get medical students
    - System intended for engineers? → get engineering students
- How many users are enough?
    - Too many users? Cost issue - observing many users is time/cost consuming
    - Too few users: High variation, Subjective issue - individual differences matter
- Partial solution: reasonable number of users
    - Big problems usually detected with handful of users
    - Small problems and fine measures need many users
    - J. Nielsen has shown that tests performed with 5 users raise at least 80% of the usability problems. 3 tests with 5 users is better than 1 test with 15 users



- Ethical considerations
    - Users are human beings!
    - Testing can be a distressing experience
        - Pressure to perform, errors inevitable
        - Feelings of inadequacy
    - You have a responsibility to alleviate:
        - Make voluntary testers with informed consent
        - Make them feel comfortable, avoid pressure
        - Let them know they can stop at any time
        - Stress that you are testing the system, not them
        - Make collected data as anonymous as possible
    - Law often requires to get human subjects approval
    - The objective of test is to identify usability problem, not to measure user's capability

11.3.1.3. Select properties to be tested

- Define a precise objective for each test. Well-defined test instructions and objectives → test results are easy to interpret
- Test objective should reflect real tasks
- Avoid bending tasks in direction of what your design best supports
- Don't choose tasks that are too fragmented
- The objective is to evaluate the software, not the user

11.3.1.4. Perform the tests

- Evaluation methods
    - Inspection
    - Direct observation
    - Query techniques: Interview, Questionnaire/Surveys
- Inspection
    - Try system/prototype: does the system "feel right"?
    - Benefits: can catch some major problems in early versions, often help the task centered walkthroughs

    - Problems: not reliable as completely subjective
    - Analytics: detailed in next class. Record user events automatically
- Direct observation
    - Evaluator observes/records users interacting with systems
    - In usability laboratory: user completes set of predetermined tasks
    - In fields: user goes through normal duties
    - Benefits: excellent at identifying gross design/interface problems, validity depends on how the situation is controlled
- Query techniques: evaluator asks users about their opinion of the system
- Interview:
    - Evaluator questions user (on 1-to-1 basis) based on prepared questions (structured, unstructured, or both)
    - Informal, subjective and relatively cheap, good for pursuing specific issues with users, flexible as evaluator can probe more deeply into interesting issues that arise
    - Prepared a set of central questions, a few good questions gets things started
    - Could be based on results of user observations, post-observation interview
    - Let user responses lead follow-up questions, avoid user leading the test
    - Post-observation Interview
        - Have the users perform an observational test
        - Create a video record of the test
        - Interview the users with the video record
            - Users comment on the events that occurred during system use. E.g. I used this button because…
            - Users often offer concrete suggestions. E.g. you should make this button smaller…
    - Advantages
        - Can be varied to suit context
        - Issues can be explored more fully
        - Can elicit user views and identify unanticipated problems
    - Disadvantages: very subjective, time consuming
- Survey
    - Prepare a set of fixed questions given to users. Need careful design
    - Expensive preparation, but cheap administration. Use cheap methods: e.g. phone, mail, email, online form
    - Does not require the presence of an evaluator. Results can be quantified
    - Survey question types: Open-ended, Rating scalar, Multichoice, Ranked
    - Open-ended
        - Asks for unprompted opinions: can you suggest any improvements to the systems?
        - Don’t restrict answers to evaluators’ expectations
        - Good for general subjective information, but difficult to analyze rigorously
    - Rating scalar questions
        - Ask user to judge a specific statement on a numeric scale “The system is easy to recover from mistakes” Disagree 1 2 3 4 5 Agree
        - Scale usually corresponds with agreement or disagreement with a statement, typically 5 or 7 scale measuring agreement/disagreement with statements
        - Easily to analysis, but needs careful design

    - Multi-choice questions
        - Respondents offered a choice of explicit responses
        - “Is there a clear way to return to a starting point or main menu?” A. Yes, it’s very clear B. Maybe, but it’s hard to navigate C. No, I cannot find
    - Ranked questions
        - Respondent places an ordering on items in a list
        - Useful to indicate a user’s preferences
        - Forced choice
        - Example: Please rank the usefulness of these methods for text input on mobile devices (1. most useful, 2. next most useful,... 0 if not useful) using keypad, QWERTY keyboard, touch screen with virtual keyboard
    - Advantages
        - Quick and reaches large user group
        - Does not require presence of evaluator
        - Potential to automate administration and analysis
        - Can be analyzed more rigorously n results can be quantified
    - Disadvantages
        - Preparation can be expensive
        - Limited quality of feedback n Depends on having right questions

11.4. Formative evaluation

- Basic steps similar to field studies, but different
    - Users are assigned to certain tasks rather than natural
    - Evaluators
        - Choose participants
        - Assign each participant some tasks
        - Watch participants to perform tasks
- “Think aloud”
    - The goal is to know what the users are thinking, not just what they are doing
    - First, ask the users to talk their thoughts while doing the task, example:
        - What they are trying to do
        - Why they took an action
        - How they interpret what the system did
    - Prompt the user to keep talking “tell me what you are thinking”
    - Make a recording the thoughts using notepad, audio or video tape
- Widely used evaluation method in industry
- Advantages
    - Simplicity - requires little expertise
    - Can provide useful insight into the user’s thinking
    - Can show how system is actually use
- Disadvantages
    - Subjective
    - selective
    - Act of describing may alter task performance

11.5. Controlled experiment

- A preferred traditional scientific method
- Clear convincing result on specific issues
- Benchmark tests
- Allows system comparison, fine-tuning of details
- Can deal with a practical problem and use the scientific method and theoretical framework
- Theory-driven, hypothesis testing
    - Study relations by manipulating independent variables
    - Observe effect on one or more dependent variables. Modify independent variables → measure dependent variables
- Procedure
    - Prepare proposal
    - Prepare environments
    - Choose participants
    - Determine forms and properties to be used
    - Determine evaluation strategies

11.5.1. Prepare test proposal

- Test objective
- Test participants
- Test environment and materials
- Test methodology to be used: detailed descriptions of systems/tasks/properties being tested
- Test measures: test data to be collected and analyzed

11.5.2. Prepare environments

- Usability testing labs
- Equipment: Computers, Video cameras, Internet access

11.5.3. Choose participants

- Participants should be chosen to represent the intended user communities, considering: background, experience with the task, motivation, education, etc.
- Participation should always be voluntary and informed

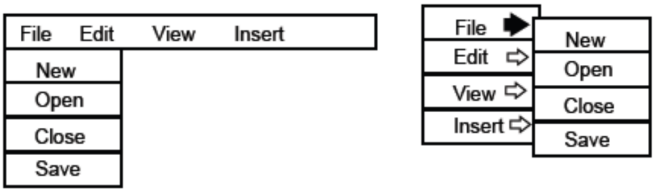
11.5.4. Determine forms and properties

- Different forms
    - Paper mockups
    - Special usability testing
    - Remote usability testing, etc.
- Videotaping participants performing tasks is often valuable

11.5.5. Evaluation strategies

- The scientific method applied to HCI include major tasks:
    - A. state lucid and testable hypothesis

    - B. identify independent variables that are to be manipulated
    - C. choose dependent variables that will be measured
    - D. select subjects and assign subjects to groups
    - E. control for biasing factors
    - F. apply statistical methods to data analysis
    - G. interpret experimental results
- Hypothesis
    - Example, the testing menu. There is no difference in user performance (time and error rate) when selecting a single item from a pop-up or a pull down menu of 4 items, regardless of the subject's previous expertise in using a mouse or using the different menu types



- Independent variables
    - Things you control and are independent of a subject's behavior
    - Example, in the menu testing example above, there are independent variables:
        - Menu type: pop-up and full-down
        - Subject: expert and novice
- Dependent variables
    - Variables dependent on the subject' behavior/reaction to the independent variable
    - The specific things you set out to quantitatively measure / observe
    - Example, in the menu testing example above, dependent variables are:
        - Speed: time to select an item
        - Error rate: number of errors made
        - Easy to learn: time to learn to use menu thoroughly
- Subjects
    - Those who perform tasks
    - Carefully select subjects and assign subjects to groups
- Statistical data analysis
    - Applying statistical methods to data analysis
    - Simple analysis
        - Determine means and medians
        - Compare values of means and medians between methods
        - Using hypothesis testing methods (e.g., student t-test, ANOVA)
- Statistical data analysis: presentation using charts, tables, graphs
- Interpreting experimental results
    - What you believe the results really mean
    - Their implications to your design
    - How generalizable the results are
    - Limitations and critique

12. Google Analytics and UI Evaluation Tools

12.1. Google Analytics

12.1.1. Overview

- Google Analytics can help answer these questions
  - Who is visiting your site?
  - How many?
  - Where do they go?
  - How much time do they spend on your site?
  - Where are they from?, etc.

12.1.2. Purpose

- Understand users
- Evaluate user interface design and experience (UI/UX)
- Improve UI/UX
- Validate business objectives and goals
- Improve business objectives and goals
- Increase conversion rate and revenue
- Support marketing and sales

12.1.3. How to use Google Analytics

- Simple steps
  - Create a property ID
  - Add JavaScript snippet to a webpage
  - Use only two API functions: ga('create',...), ga('send',...)

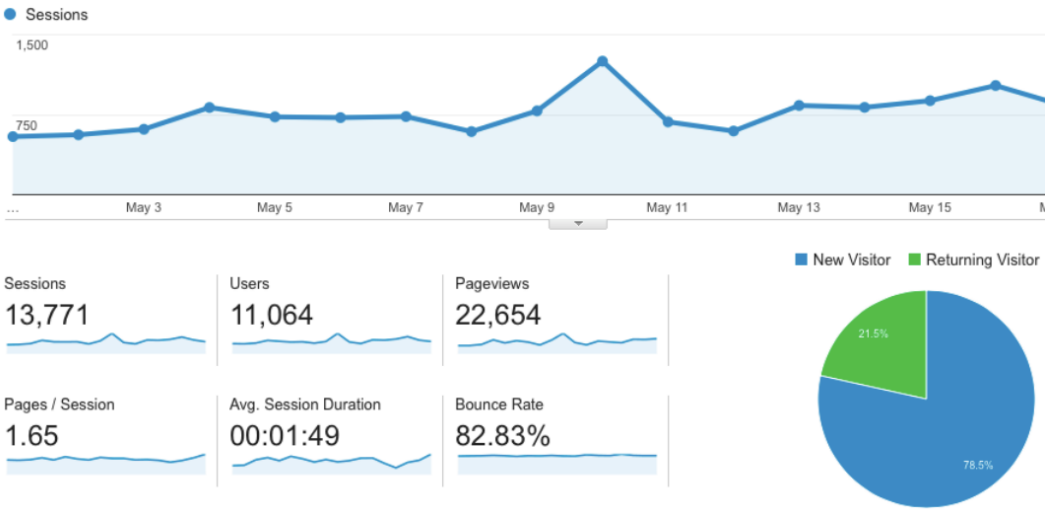
```
<!-- Google Analytics -->
<script>
(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
})(window,document,'script','https://www.google-analytics.com/analytics.js','ga');

ga('create', 'UA-XXXXX-Y', 'auto');
ga('send', 'pageview');
</script>
<!-- End Google Analytics -->
```

- Use the functions for
  - Sending page views: everytime users view a page
  - Sending events: any event a website needs to record. E.g., click on a link, click on a button, play video, etc.

12.1.4. Key metrics

- Bounce rate
- Number of users
- Number of sessions
- Session duration
- New visitors
- Returning visitors
- Number of pageviews
- Conversion rate



12.1.5. Who visits the site (audience)

- Demographics
- Interests
- Geography
- Behavior
- Technology
- Mobile

12.1.6. Technologies

- Browser, OS, Screen Resolution, Flash version, etc.

12.1.7. Where are they from (acquisition)

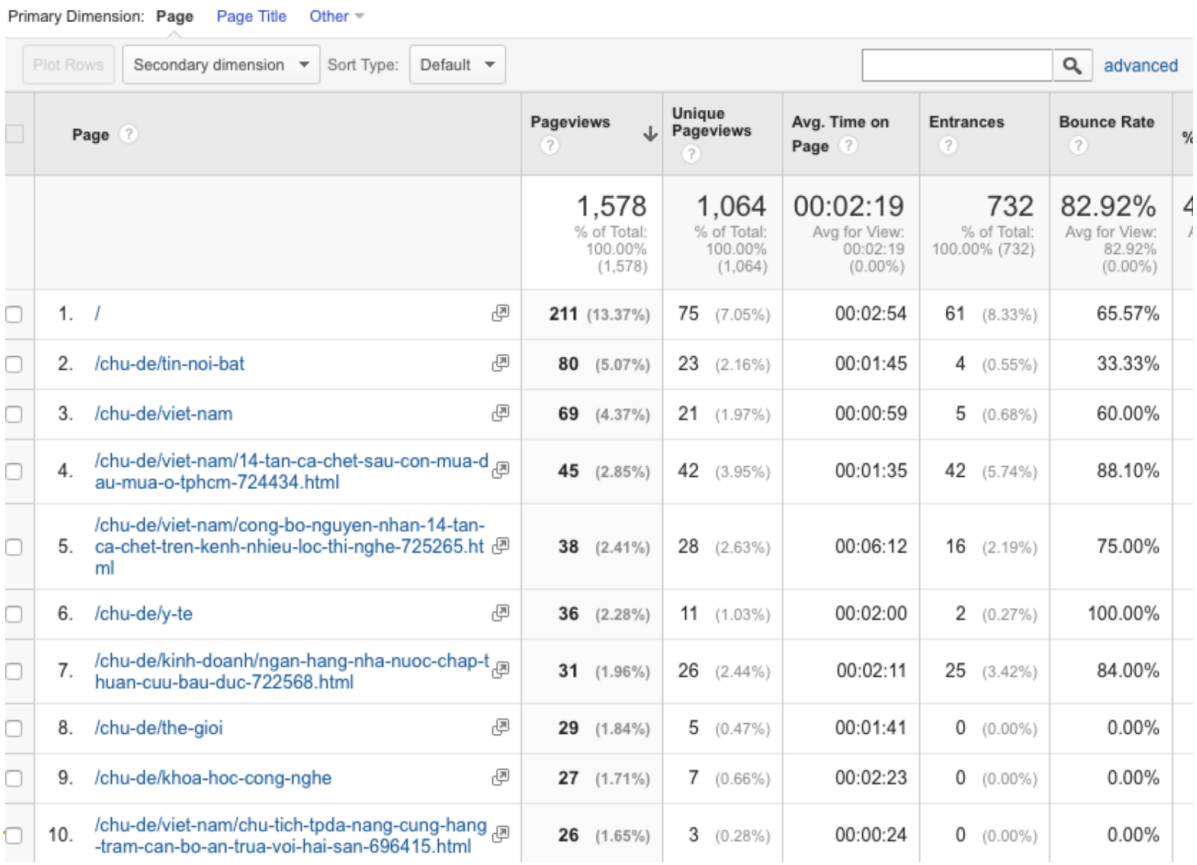
- Channels
- Treemaps
- Source/Medium
- Referrals
- Social
- Campaigns

	Default Channel Grouping	Acquisition			Behavior			Conversions
		Sessions ↓	% New Sessions	New Users	Bounce Rate	Pages / Session	Avg. Session Duration	Goal Conversion Rate
		732 % of Total: 100.00% (732)	77.05% Avg for View: 77.05% (0.00%)	564 % of Total: 100.00% (564)	82.92% Avg for View: 82.92% (0.00%)	2.16 Avg for View: 2.16 (0.00%)	00:02:41 Avg for View: 00:02:41 (0.00%)	0.00% Avg for View: 0.00% (0.00%)
☐	1. Organic Search	510 (69.67%)	83.92%	428 (75.89%)	87.25%	1.24	00:01:03	0.00%
☐	2. Social	111 (15.16%)	48.65%	54 (9.57%)	79.28%	3.55	00:04:50	0.00%
☐	3. Direct	73 (9.97%)	63.01%	46 (8.16%)	54.79%	6.84	00:11:57	0.00%
☐	4. Referral	38 (5.19%)	94.74%	36 (6.38%)	89.47%	1.34	00:00:30	0.00%

12.1.8. What do users do (behavior)?

- Content visited
- Landing pages
- Site speed
- Events

12.1.9. Top Pages



Primary Dimension: **Page** Page Title Other

Plot Rows | Secondary dimension | Sort Type: Default | advanced

	Page	Pageviews	Unique Pageviews	Avg. Time on Page	Entrances	Bounce Rate
		1,578 % of Total: 100.00% (1,578)	1,064 % of Total: 100.00% (1,064)	00:02:19 Avg for View: 00:02:19 (0.00%)	732 % of Total: 100.00% (732)	82.92% Avg for View: 82.92% (0.00%)
1.	/	211 (13.37%)	75 (7.05%)	00:02:54	61 (8.33%)	65.57%
2.	/chu-de/tin-noi-bat	80 (5.07%)	23 (2.16%)	00:01:45	4 (0.55%)	33.33%
3.	/chu-de/viet-nam	69 (4.37%)	21 (1.97%)	00:00:59	5 (0.68%)	60.00%
4.	/chu-de/viet-nam/14-tan-ca-chet-sau-con-mua-dau-mua-o-tphcm-724434.html	45 (2.85%)	42 (3.95%)	00:01:35	42 (5.74%)	88.10%
5.	/chu-de/viet-nam/cong-bo-nguyen-nhan-14-tan-ca-chet-tren-kenh-nhieu-loc-thi-nghe-725265.html	38 (2.41%)	28 (2.63%)	00:06:12	16 (2.19%)	75.00%
6.	/chu-de/y-te	36 (2.28%)	11 (1.03%)	00:02:00	2 (0.27%)	100.00%
7.	/chu-de/kinh-doanh/ngan-hang-nha-nuoc-chap-thuan-cuu-bau-duc-722568.html	31 (1.96%)	26 (2.44%)	00:02:11	25 (3.42%)	84.00%
8.	/chu-de/the-gioi	29 (1.84%)	5 (0.47%)	00:01:41	0 (0.00%)	0.00%
9.	/chu-de/khoa-hoc-cong-nghe	27 (1.71%)	7 (0.66%)	00:02:23	0 (0.00%)	0.00%
10.	/chu-de/viet-nam/chu-tich-tpda-nang-cung-hang-tram-can-bo-an-trua-voi-hai-san-696415.html	26 (1.65%)	3 (0.28%)	00:00:24	0 (0.00%)	0.00%

12.2. Other tools

- Optimizely: A/B Testing
- Inspectlet: screen capture, heatmap, etc.
- Crazy Egg: heatmap, points of interest
- Mixpanel: individual actions on a per-user level to process usage patterns
- www.feng-gui.com/

13. Internationalization (i18n)

13.1. Internationalization and localization

- Internationalization

    - The process of making software ready for translation or adaptation into different languages and cultures
    - Often called as i18n – 18 characters between ‘i’ and ‘n’ in the word “internationalization’
- Localization
    - The process making software to a specific language and culture
    - More than just translate, find, and replace words

13.2. Text translation

- All user-visible text has to be translated, including error messages
    - Text in code
    - Text in images
    - Text in data files
- Translation results in the change of text length

13.2.1. Different scripts

- English: This is the class on user interface design
- Vietnamese: Đây là lớp học về thiết kế giao diện người dùng
- Russian: это класс дизайна пользовательского интерфейса
- Chinese (traditional): 這是用戶界面設計類
- Arabic مدختسملا ةهجاو ميمصت يف ةقبطلا يه هذه

13.2.2. Text direction

- Left-to-right: English, Vietnamese, French, etc.
- Right-to-left: Arabic, Hebrew
- To-down, left-to-right, right-to-left: Chinese, Korean, Japanese
- Text direction affects drawing, screen layout, editing, etc.

13.3. Sort order

- Unicode is often used
- It is not always ordered according to a language
- Upper/lowercase affects order
- Java’s String.compareTo() does not work as expected all the times

13.4. Formatting

- Numbers
    - US/UK: 1,000.5 (one thousand point five)
    - Vietnam, Germany: 1.000,5
    - France: 1 000,5
- Date
    - US: 5/10/2021 (M/D/Y)
    - VN and others: 10/5/2021 (D/M/Y)
- Time
    - US: 12:30 PM (H:M)

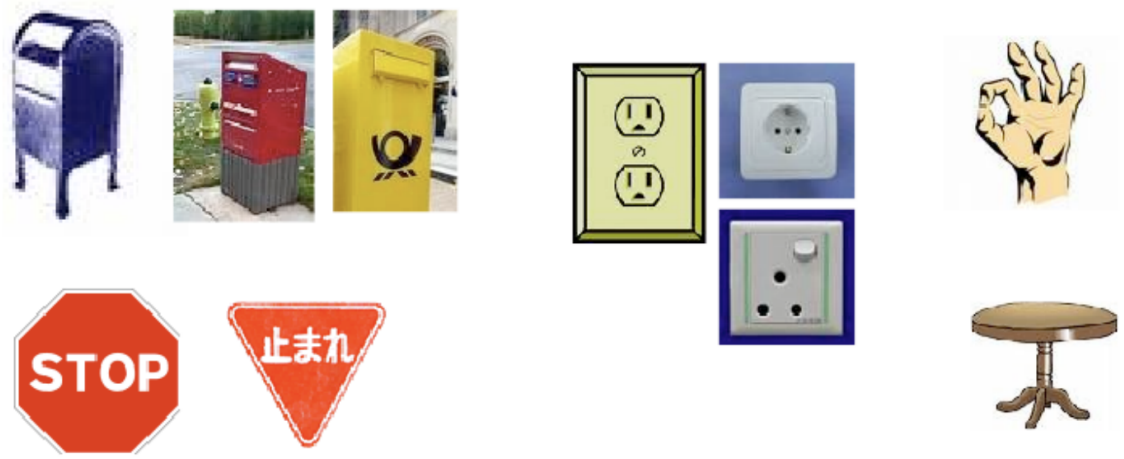
- VN: 12.30 chiều

13.5. Color conventions

- White
  - East Asia (China, Vietnam, etc.): death, used in funeral
  - US: purity, used in wedding
- Red
  - East Asia (China, Vietnam, etc.): luck, on new year celebrations
  - US: sometimes red refers to danger, e.g. stop sign

13.6. Icons

- Common icons in one culture may not be common



13.7. Guidelines on supporting I18N

- Message and resource files
- Unicode
- Bidirectionality
- Formatting libraries
- Separating processing and structure from presentation

13.7.1. Message and resource files

- Message/resource files are separated from source code
  - Storing messages, images, icons
  - Can be translated and created without changing code
- Pay attention to messages with dynamic parts
  - E.g., “<N> users have visited since <Date>”
  - Above is the structure in English, but in Vietnamese, it’s different “Số người đến thăm từ ngày <Date> là <N>”

13.7.2. Unicode

- Many character sets have been used: ASCII, ASCII extension, Unicode

- Use Unicode for better i18n support

13.7.3. Bi-directionality

- Bi-directional text display and editing
    - Storing in memory may be different from displaying
    - String in memory: “This is arabic text”
    - Shown on screen:
        - English: “This is txet icbara”
        - Arabic: “txet icbara This is”
- Use UI toolkits provided in the language

13.7.4. Separating processing from presentation

- Use and store replaceable images and icons, separate them from code
- Separate and allow configuring fonts and colors without changing code

13.8. Summary

- Benefits of supporting internationalization
    - Writing once, deploying in many languages and cultures
    - Avoid changes in code → reduce risks and cost
- Internationalization makes a user interface ready for localizing into different locales
- We should consider differences in
    - Languages
    - Scripts
    - Cultures
    - Formatting conventions