Course Overview

- Introduction
- Understanding Users and Their Tasks
- Principles and Guidelines
- Interacting With Devices
- Interaction Styles
- UI Design Elements
- Visual Design Guidelines
- UI Development Tools
- Iterative Design and Usability Testing
- Project Presentations and Selected Topics
- Case Studies
- Recent Developments in HCID
- Conclusions

Chapter Overview Testing and Evaluation

- Motivation
- Objectives
- Prototyping
  - Prototypes
  - Prototyping Techniques
  - Benefits and Drawbacks
- Evaluation
  - Methods, Techniques and Tools
  - Comparison
- Important Concepts and Terms
- Chapter Summary

Motivation

- testing and evaluation of user interfaces is critical for the acceptance of products
- evaluations should be done as early as possible
  - mock-ups, scenarios, prototypes, …
- testing and evaluation can be expensive
  - correcting errors late in the development process is even more expensive
  - for many software systems, modifications based on dissatisfied users are a very large part of the overall costs
- a careful selection of the test and evaluation methods is important
  - not all methods are suitable for all purposes

Objectives

- to know the important methods for testing and evaluating user interfaces
- to understand the importance of early evaluation
- to be able to select the right test and evaluation methods for the respective phase in the development

Prototypes

- simulate the structure, functionality, or operations of another system
  - represent a model of the application, service, or product to be built
  - may or may not have any real functionality
  - can be either paper based or computer based

Paper-based Prototypes

- cheap
- low fidelity
- can often be useful to demonstrate a concept
  - e.g., a back-of-the-envelope sketch
- can not show functionality so that users can actually interact with them

Computer-based Prototypes

- higher fidelity than paper based
- can demonstrate some aspect with varying degrees of functionality
- can offer valuable insights into how the final product or application may look like

Why Prototype?

- part of the iterative nature of UI design
- 20%-40% of all system problems can be traced to problems in the design process
- 60%-80% can be traced to inaccurate requirements definitions
- cost of correcting a problem increases dramatically as the software life cycle progresses

Prototyping Techniques

- low-fidelity prototypes
- high-fidelity prototypes

Low-fidelity Prototypes

- low-fidelity prototypes
  - cheap, rapid versions of the final system
  - limited functionality and/or interactivity
  - depict concepts, designs, alternatives, and screen layouts rather than model user interaction with a system
    - e.g. storyboard presentations, proof-of-concept prototypes
  - demonstrate the general 'feel and look' of the UI
    - their purpose is not to show in detail how the application operates
  - are often used early in the design cycle
    - to show general conceptual approaches without investing too much time or effort

High-fidelity Prototypes

- high-fidelity prototypes
  - fully interactive
    - users can enter data into entry fields, respond to messages, select icons to open windows, and interact with the UI
  - represent the core functionality of the product's UI
  - typically built with 4GLs such as Smalltalk or Visual Basic
    - can simulate much of the functionality of the final system
  - trade off speed for accuracy
    - not as quick and easy to create as low-fidelity prototypes
    - faithfully represent the UI to be implemented in the product
    - can be almost identical in appearance to the actual product

Comparison

Type	***Advantages***	***Disadvantages***
Low-Fidelity Prototyping	Lower development cost Evaluate different design concepts Useful communication vehicle Addresses screen layout issues Useful for identifying market requirements Proof of concept	Limited error checking Poor detailed specification for coding Facilitator driven Limited usefulness after requirements established Limitations in usability testing Navigational & flow limitations
High-Fidelity Prototyping	High degree of functionality Fully interactive User driven Defines navigational scheme Useful for exploration & testing Look and feel of final product Serves as a living specification Marketing and sales tool	More expensive to develop Time consuming to build Inefficient for proof of concept designs Not effective for requirements gathering

Fidelity Requirements

- recent study by Cantani and Biers (1998) investigated the effect of prototype fidelity on the information obtained from performance test
  - 3 levels of prototypes:
    - paper - low fidelity
    - screen shots - medium fidelity
    - interactive Visual Basic - high fidelity

Case Study (cont.)

- 30 university students performed 4 typical library search tasks using one of the prototypes
  - total of 99 usability problems were uncovered
- no significant difference in the number and severity of problems identified, and a high degree of commonality in the specific problems uncovered by users using the 3 prototypes
  - Catani, M.B., And Biers, D.W. (1998). Usability Evaluation and Prototype Fidelity: Users and Usability Professionals. Proceedings of the Human Factors and Ergonomic Society, 42nd Annual Meeting, 1331-1336.

Low-fidelity Prototyping

- identify key market and user requirements
- provide a very high-level view of the proposed UI and service concept
- provide a common language or vision
- develop a common understanding with others
- investigate early concepts and ideas independently of platform, technology, and other issues
- evaluate design alternatives
- get customer support during requirements gathering
- elicit user input prior to selecting a design

High-fidelity Prototyping

- create a living specification for programmers and customers
- make an impression with customers to show how well the product, service, or application will operate
  - prior to the code being fully developed
- test UI issues prior to committing to a final development plan
  - e.g., error handling, instructions

Software Prototypes

- actually work to some degree
  - not an idea or drawing
- must be built quickly and cheaply
  - throw-away - thrown away or discarded immediately after use
  - incremental - separate components, added to the system
  - evolutionary - may eventually evolve into the final system
- may serve many different purposes
  - elicit user reactions, serve as a test bed
- integral part of an iterative process
  - includes modification and evaluation

Levels of Prototyping

- full prototype
- horizontal prototype
- vertical prototype
- scenarios

Full Prototype

- contains complete functionality
- lower performance than the final system
  - e.g. trial system with a limited number of simultaneous users
  - may be non-networked, not fully scalable, ...

Horizontal Prototype

- demonstrate the operational aspects of a system
  - do not provide full functionality
  - e.g. users can execute all navigation and search commands, but without retrieving any real information as a result of their commands
- reduced level of functionality
  - all of the features present

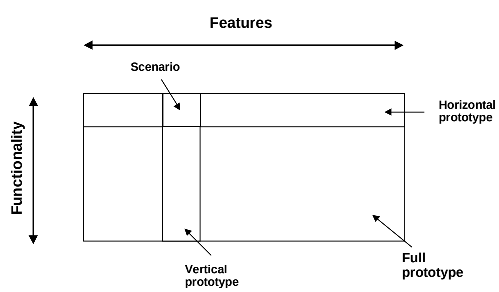
Vertical Prototype

- contain full functionality, but only for a restricted part of the system
  - e.g., full functionality in one or two modules, but not entire system
  - e.g. in an airline flight information system, users can access a database with some real data from the information providers, but not the entire data
    - in other words, they can play with a part of the system
- reduced number of features, but with full functionality

Scenarios

- both the level of functionality and the number of features are reduced
- very cheap to design and implement
- but, only able to simulate the UI as long as the test user follows a previously plan test
- small, can be changed frequently and re-tested
- reduced level of functionality and reduced number of features

Diagram Levels



Levels of prototyping.

Chauffeured Prototyping

- involves the user watching while another person ‘drives’ the system
  - usually a member of the development team
- the system may not yet be complete enough for the user to test it
  - it is nevertheless important to establish whether a sequence of actions is correct

Wizard of Oz

- a person hidden to the user provides feedback for the system
- user is unaware that he/she is interacting with another user who is acting as the system
- usually conducted very early in development
  - to gain an understanding of the user's expectations

Testing of Prototypes

- structured observation
  - observe typical users attempting to execute typical tasks on a prototype system
  - note number of errors and where they occur, confusions, frustrations, and complaints
- benchmarking
  - oriented toward testing the prototype UI or system against any pre-established performance goals
    - example: error-free performance in less than 30 min

Testing of Prototypes (cont.)

- experimentation
  - two or more UI design (prototype) alternatives with the same functionality are directly compared
  - the one that leads to the best results is selected for the final product

Benefits of Prototyping

- integral part of the iterative design process
- permits proof of concept/design validation
- raises issues not usually considered until development
- provides a means for testing product- or application-specific questions that cannot be answered by generic research or existing guidelines
- permits valuable user feedback to be obtained early in the design process

Benefits of Prototyping (con t.)

- qualitative and quantitative human performance data can be collected within the context of the specific application
- provides a relatively cheap and easy way to test designs early in the design cycle
- permits iterative evaluation and evolving understanding of a system, from design to the final product
- improves the quality and completeness of a system's functional specification
- substantially reduces the total development cost for the product or system

Drawbacks

- inadequate analysis
  - inadequate understanding of the underlying problem
  - the lack of a thorough understanding of the application, service, or product being developed
- the prototype may look like a completed system
  - customers may get the mistaken idea that the system is almost finished, even when they are told very clearly that it is only a prototype
- unattainable expectations
  - unrealistic expectations with respect to actual product performance
- ignoring reality
  - limitations and constraints that apply to the real product may often be ignored within the prototyping process
    - e.g., network constraints

Drawbacks (Cont.)

- users that are never satisfied
  - users can ask for things that are beyond the scope of the project
- viewing the prototype as an exercise
  - developers may develop the wrong thing
  - at great effort and expense
- the trap of over-design or under-design
  - “just one more feature ...”
  - “this is just the prototype, we’ll fix it when we develop the product”

User Interface Evaluation

- terminology
- evaluation and UI design
- time and location
- evaluation methods
- usability

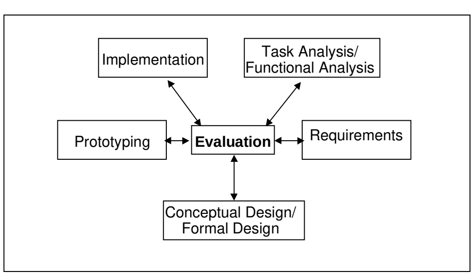
Evaluation

- gathering information about the usability of an interactive system
  - in order to improve features within a UI
  - to assess a completed interface
- assessment of designs
- test systems to ensure that they actually behave as expected, and meet user requirements

Evaluation Goals

- to improve system usability, thereby increasing user satisfaction and productivity
- to evaluate a system or prototype before costly implementation
- to identify potential problem areas, and perhaps suggest possible solutions

Evaluation and UI Design



The star life cycle (adapted from Hix & Hartson, 1993).

Hix, D., & Hartson, H.R. (1993). *Developing User Interfaces: Ensuring Usability through Product & Process*. New York: John Wiley.

Evaluation Time

- not a single phase in the design process
- ideally, evaluation should occur throughout the design life cycle
  - feedback of results into modifications to the UI design
- close link between evaluation and prototyping techniques
  - help to ensure that the design is assessed continuously

Types of Evaluation

- formative evaluation
  - takes place before implementation in order to influence the product or application that will be produced
  - are usability goals met?
- summative evaluation
  - takes place after implementation with the aim of testing the proper functioning of the final system
  - improve the interface, find good/bad parts
  - examples
    - quality control
      - a product is reviewed to check that it meets its specifications
    - testing to check whether a product meets International Standards Organization (ISO) standards

Evaluation Location

- laboratory studies
  - controlled setting
  - experimental paradigm
- field studies
  - natural settings
  - unobtrusive, non-invasive if possible
- with or without users
  - in the lab with users
    - participatory design
  - in the lab without users
    - brainstorming sessions, storyboarding, workshops, pencil-and-paper exercises

Evaluation Methods

- analytic evaluation
- observational evaluation
- interviews
- surveys and questionnaires
- experimental evaluation
- expert evaluation

Analytic Evaluation

- uses formal or semi-formal interface descriptions
  - e.g. GOMS
  - to predict user performance
  - to analyze how complex a UI is and how easy it should be to learn
  - can start early in the design cycle
    - an interface is represented only by a formal or semi-formal specification
  - doesn't require costly prototypes or user testing
    - not all users are experts, and not all users learn at the same rate or make the same number or same types of errors
    - not all evaluators have the necessary expertise to conduct these analyses

Analytic Evaluation (cont.)

- enables designers to analyze and predict expert performance of error-free tasks in terms of the physical and cognitive operations that must be carried out
  - examples:
    - how many keystrokes will the user need to do task A?
    - how many branches in a hierarchical menu must a user cross before completing task B?
    - in the absence of errors, how many errors should we expect users to make, and how long should it take them?

Observational Evaluation

  - involves observing or monitoring users' behavior while they are using/interacting with a UI
    - applies equally well to listening to users interacting with a speech user interface
  - can be carried out in a location specially designed for observation such as a usability lab, or informally in a user's normal environment with minimal interference
- Hawthorne effect
  - users can alter their behavior and their level of performance if they aware that they are being observed, monitored, or recorded

Observational Evaluation Techniques

- direct observation
  - but, beware of the Hawthorne effect
- video/audio recording
  - video/audio taping user activity
- software logging
  - time-stamped logs of user input and output
  - monitoring and recording user actions, and corresponding system behavior
- Wizard of Oz
  - person behind the curtain
- verbal protocols
  - thinking aloud

Interviews

- structured
  - pre-determined set of questions, fixed format
  - e. g. public opinion surveys
- unstructured
  - set topic, but no set sequence
  - free flowing and flexible
  - e.g. talk show

Surveys and Questionnaires

- seek to elicit users' subjective opinions about a UI
  - types of questions
    - open-ended questions - "what do you think about this course?"
    - closed-ended questions - select an answer from a choice of alternative replies, e.g., yes/no/don't know; true/false).
    - rating scales (thurstone scale (1-10 with 1 being worst), likert scale (strongly disagree to strongly agree with a neutral point)
    - semantic differential (bipolar adjectives e.g., easy-difficult, clear-confusing at the end points)
    - multiple choice (a, b, c, d, or none of the above)
    - value (with range or percentage) - "How many hours per day do you spend watching TV?"
    - multiple answer/free form - "Name the five top grossing films of the year."

Experimental Evaluation

- uses experimental methods to test hypotheses about the use of an interface
  - also known as usability testing
- controlled environments, hypothesis testing, statistical evaluation and analysis
- typically carried out in a specially equipped and designed laboratory

Expert Evaluation

- involves experts in assessing an interface
- informal diagnostic method
  - somewhere between the theoretical approach taken in analytic evaluation, and more empirical methods such as observational and experimental
- expert evaluation that is guided by general "rules of thumb" is known as heuristic evaluation

Usability

- definitions
- measurements
- justification
- considerations
- system acceptability
- usability and evaluation
- usability goals
- usability testing
- usability testing methods
  - focus groups
  - contextual inquiry
  - co-discovery
  - active intervention
- usability inspection methods
  - walkthroughs
  - heuristic evaluation

Definitions of Usability

- usability is a fuzzy, global term, and is defined in many ways
- some common definitions
  - “the effectiveness, efficiency, and satisfaction with which users are able to get results with the software”
  - “usability is being able to find that you want and understand what you find”
  - “usability refers to those qualities of a product that affect how well its users meet their goals”

Definitions (cont.)

- “the capability of the software to be understood, learned, used, and liked by the user when used under specified conditions” (ISO 9126-1)
- “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use” (ISO 9241-11)
- “usability means that people who use a [system or] product can do so quickly and easily to accomplish their own tasks” (Dumas and Redish, 1994)

Usability Aspects

- usability means focusing on users
- people use products to be productive
  - the time it takes them to do what they want
  - the number of steps they must go through
  - the success that they have in predicting the right action to take
- users are busy people trying to accomplish tasks
  - people connect usability with productivity
- users decide when a product is easy to use
- incorporates attributes of ease of use, usefulness, and satisfaction

Usability (Cont.)

- grounded in data from and about a product’s or system’s intended users
  - a usable product empowers users
  - a usable product provides functionality designed from the users’ perspective
- measure of quality
  - major factor in the user’s overall perception of system quality
    - becomes even more important as the number and types of users increase

Usability Justification

- some statistics of cost justifying usability
  - 80% of software lifecycle costs occur after the product is released, in the maintenance phase
    - of that work, 80% is due to unmet or unseen user requirements
    - only 20% is due to bugs or reliability problems
  - 40-100x more expensive to fix problems in the maintenance phase than in the design phase
  - systems designed with usability principles in mind typically reduce the time needed for training by 25%
  - user-centered design typically cuts errors in user-system interaction from 5% to 1%.
    - Tom Landauer. The Trouble With Computers. 1995.

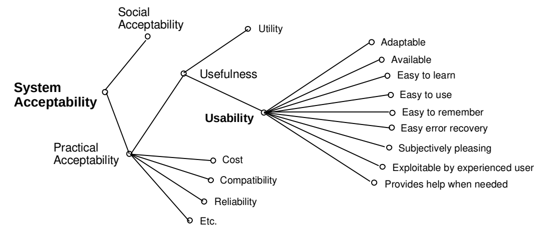
Usability Considerations

- functionality
  - can the user do the required tasks?
- understanding
  - does the user understand the system?
- timing
  - are the tasks accomplished within a reasonable time?
- environment
  - do the tasks fit in with other parts of the environment?
- satisfaction
  - is the user satisfied with the system?
  - does it meet expectations?

Considerations (cont.)

- safety
  - will the system harm the user, either psychologically or physically?
- errors
  - does the user make too many errors?
- comparisons
  - is the system comparable with other ways that the user might have of doing the same task?
- standards
  - is the system similar to other that the user might use?

System Acceptability



(Adapted from Nielsen, 1993)

Usability and Design

- usability and design
  - usability is not something that can be applied at the last minute, it has to be built in from the beginning
- engineer usability into products
  - focus early and continuously on users
  - integrate consideration of all aspects of usability
  - test versions with users early and continuously
  - iterate the design

Usability and Design (cont.)

- involve users throughout the process
- allow usability and users' needs to drive design decisions
- work in teams that include skilled usability specialists, UI designers, and technical communicators
  - because users expect more today
  - because developing products is a more complex job today
- set quantitative usability goals early in the process

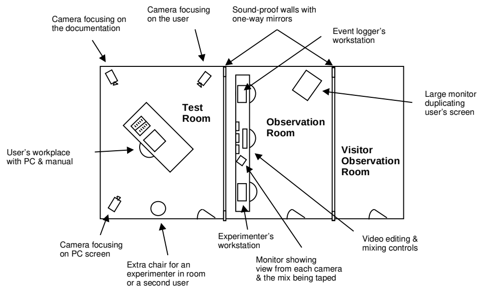
Usability Engineering

- primary goals
  - to improve the usability of the system being tested
  - improve the process by which products are designed and developed
    - the same problems are avoided in other products
- the participants represent real users, do real tasks
- observe and record what the participants do and say
- analyze the data, diagnose the real problems, and recommend changes to fix those problems
- Microsoft invested nearly 3 years of development and 25k hours of usability testing in Office 97

Usability Goals

- performance or satisfaction metrics
  - time to complete, errors, confusions
  - user opinions
  - problem severity levels
- benefits
  - guide and focus development efforts
  - measurable evidence of commitment to customers
  - e.g. user opinions
    - 80% of users will rate ease of use and usefulness at 5.5 or greater on a 7-point scale
    - target = 80%, minimally acceptable value = 75%

Usability Testing Lab



Floor plan of a hypothetical, but typical usability lab

Usability Testing Methods

- focus groups
- contextual inquiry
- co-discovery
- active intervention
- usability inspection methods
- walkthroughs
- heuristic evaluation

Focus Groups

- highly structured discussion about specific topics
  - moderated by a trained group leader
- typically held prior to beginning a project
  - in order to uncover usability needs before any actual design is started
- to probe users' attitudes, beliefs, and desires
  - they do not provide information about what users would actually do with the product
- can be combined with a performance test
  - e.g. hand out a user guide; ask whether they understand it, what they would like to see, what works for them, what doesn't, etc.

Contextual Inquiry

- technique for interviewing and observing users individually at their regular places of work as they do their own work
- contextual inquiry leads to contextual design
- very labor intensive
  - requires a trained, experienced contextual interviewer
- observation should be as non-invasive as possible. not always practical
- can be used at the earliest pre-design phase
  - then iteratively throughout product design and development

Co-discovery

- technique in which two participants work together to perform tasks
- participants are encouraged to talk to each other as they work
  - yields more information about what the participants are thinking and what strategies they are using to solve their problem than by asking individual participants to think out aloud
- more expensive than single participant testing
  - two people have to be paid for each session
  - more difficult to watch two people working with each other and the product

Active Intervention

- a member of the test team sits in the room with the participant
  - actively probes the participant's understanding of whatever is being tested
- particularly useful in early design
  - excellent technique to use with prototypes, because it provides a wealth of diagnostic information
- not so good if the primary concern is to measure time to complete tasks or to find out how often users will request help

Usability Inspection Methods

- evaluators inspect or examine usability-related aspects of a UI
  - usability inspectors can be usability specialists, software development consultants, or other types of professionals
- formal:
  - usability inspections - UI is checked against quantitative usability goals and objectives

Usability Inspection Methods (cont.)

- informal
  - guideline reviews - interface is checked against a comprehensive list of usability guidelines
  - consistency - evaluate cross-product consistency “look and feel”
  - standards inspections - check for compliance with applicable standards
  - cognitive walkthroughs (more later)
  - feature inspections - focus on the function delivered in a software system
  - heuristic evaluation (more later)

Structured Walkthroughs

- peers or experts walk through the design
- very common in software development
  - code inspection and review
- called a cognitive walkthrough in UI design
- aim is to evaluate the design in terms of how well it supports the user as s(he) learns how to perform the required tasks
- a cognitive walkthrough considers:
  - what impact will the interaction have on the user?
  - what cognitive processes are required?
  - what learning problems may occur?

Usability Walkthrough

- systematic group evaluation
  - conducted to find errors, omissions, and ambiguities in the proposed design, and to ensure conformance to standards.
- advantages
  - early feedback, relatively informal
    - can be called on short notice
    - can focus on critical areas
- disadvantages
  - feedback may be taken personally
  - focus on finding errors, not solutions
  - generally does not involve end users

Heuristic Evaluation

- getting experts to review the design
- informal inspection technique where a small number of evaluators examine a user interface and look for problems that violate some of the general heuristics of user interface design.
  - Nielsen, J., And Molich, R. (1990). Heuristic Evaluation of User Interfaces. CHI ’90 Proceedings. New York: ACM Press.

UI Heuristics

- use simple and natural language
- speak the user’s language (match between the system and the real world)
- minimize memory load (recognition rather than recall)
- be consistent (consistency and standards)
- provide feedback (visibility of system status)
- provide clearly marked exits (user control and freedom)
- provide shortcuts (flexibility and efficiency of use)
- provide good error messages
- prevent errors

Heuristic Evaluation (cont.)

- basic questions explored by heuristic evaluation
  - are the necessary capabilities present to do the users' tasks?
  - how easily can users find or access these capabilities?
  - how successful can users do their tasks with the capabilities?

Outcome Heuristic Evaluation

- types of problems uncovered by heuristic evaluation
  - hard-to-find functionality
    - menu choices and icon labels don't match user's terminology
    - important choices are buried too deep in menus or window sequences
    - choices located are far away from the user's focus
    - choices don't seem related to menu title
  - limited or inaccurate task flow
    - screen sequences and/or menus don't reflect user tasks
    - unclear what user should do next
    - unclear how to end task

Heuristic Evaluation (cont.)

- clutter
  - too many choices in menus
  - too many icons or buttons
  - too many fields
  - too many windows
  - misuse of shading and color to set off elements
- cumbersome operation
  - too much scrolling is needed to accomplish tasks
  - long-distance mouse movement is required
  - actions required by the software are not related to the user's task
  - focus area is too small for easy selection

Heuristic Evaluation (cont.)

- lack of navigational signposts
  - task sequence is not clear
  - no labeling of the current position
  - no way to see the overall structure (index or map)
- lack of feedback
  - not clear when the user has reached the end
  - no indication that the operation is in progress
  - "beep" with a message, or a message stating a problem but not the solution
  - messages are in hard-to-find locations

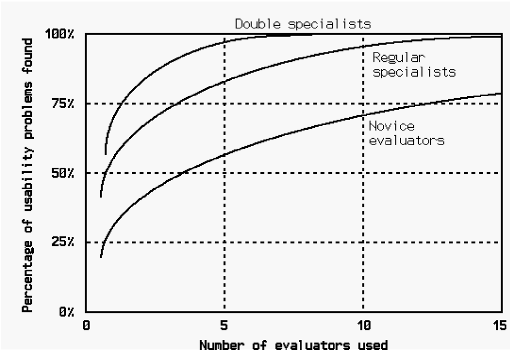
Practical Aspects

- how many evaluators are enough?
  - 2 evaluators at a minimum
  - usability specialists and domain experts
  - more evaluators find more problems
  - more evaluators provide a better indication of the seriousness of problems
  - but, more evaluators require more time to coordinate findings and develop recommendations

Practical Aspects (cont.)

- should the focus of the evaluation be on first use, continued use, or both?
  - first use: how "learnable" and usable is the system on the first look? what prerequisite training should be provided?
  - continued use: how convenient is the system for expert users? what efficiencies must be provided?
- how deep should the investigation be?
  - usability and usefulness
  - identifying problems only or solutions too?
  - number of user audiences, and usage scenarios to consider
  - time constraints?

Evaluators



Strengths Heuristic Evaluation

- skilled evaluators can produce high-quality results
- key usability problems can be found in a limited amount of time
- provides a focus for follow-up usability studies

Weaknesses Heuristic Evaluation

- not based on primary user data
  - heuristic evaluation does not replace studying actual users
  - heuristic evaluation does not necessarily indicate which problems will be most frequently experienced
  - heuristic evaluation does not represent all user groups
- limited by evaluators' experience and expertise
  - domain specialists normally lack user modeling expertise
  - usability specialists may lack domain expertise
  - "double" experts produce the best results
  - usability specialists are better than novice evaluators
  - better to concentrate on usability expertise, because developers can usually fill domain gaps

Selection of Evaluation Methods

- factors to consider
  - stage in the cycle at which the evaluation is carried out
    - design vs. implementation stage
  - style of evaluation
    - laboratory or field studies?
  - level of subjectivity or objectivity
  - type of measures needed
    - qualitative or quantitative?
  - type of information needed
  - immediacy of the response
  - level of interference implied
  - resources required

Comparison of Evaluation Methods

Hints

- don't rely on a single evaluation method
  - use multiple evaluation methods to supplement each other
- use both formal and informal methods where applicable, but recognize the tradeoffs
- do feature inspection early in the design process
- perform heuristic evaluations of paper-based mock-ups and of functioning prototype designs
- perform standards and consistency checks
- test and re-test often until ...
  - usability goals are met
  - customers, users, and developers are satisfied

Selection of Evaluation Methods

Method	Lifecycle Stage	No. users needed	Advantages	Disadvantages
Heuristic evaluation	Early design	None	Finds individual usability problems. Can address expert user issues.	Does not involve real users, so does not find surprises relating to their needs.
Performance measures	Competitive analysis, final testing	At least 10	Hard numbers. Results are easy to compare.	Does not find individual usability problems.
Thinking aloud	Iterative design, formative evaluation	3-5	Pinpoint user misconceptions. Cheap.	Unnatural for users. Hard for experts to verbalize.
Observation	Task analysis, follow-up studies	3 or more	Ecological validity - reveals users' real tasks. Suggests functions & features.	Appointments hard to set up. No experimenter control.
Questionnaires	Task analysis, follow-up studies	at least 30	Finds subjective user preferences. Easy to repeat.	Pilot work needed (to prevent misunderstandings).
Interviews	Task analysis	5	Flexible, in-depth probing of attitudes & experience.	Time consuming. Hard to analyze & compare.
Focus groups	Task analysis, user involvement	6-9 per group	Spontaneous reactions & group dynamics.	Hard to analyze. Low validity.
Logging actual use	Final testing, follow-up studies	at least 20	Finds highly used (or unused) features. Can be run continuously.	Analysis programs needed for huge mass of data. Violation of users' privacy.
User feedback	Follow-up studies	100s	Tracks changes in use, requirements, & views.	Special organization needed to handle replies.

Comparison Evaluation Methods

Method	Advantages	Disadvantages
Analytic	Usable early in design. Few resources required. Cheap.	Narrow focus. Lack of diagnostic value for redesign. Makes broad assumptions of users' cognitive operations. Requires experts.
Observational	Quickly pinpoints difficulties. Verbal protocols are valuable source of information. Provides rich qualitative data.	Observation can affect users' activity & performance levels. Analysis can be both time & resource consuming.
Survey	Addresses users' opinions & understanding of the interface. Can be used for diagnosis. Can provide qualitative data. Can be used with many users.	Low response rates (especially for mailed questionnaires). Possible interviewer bias. Possible response bias. Analysis can be complicated & lengthy. Interviews are very time consuming.
Experimental	Powerful. Provides quantitative data for statistical analysis's. Provides replicable results.	High resource demands. Evaluators require specialized skills & knowledge of experimental design. Takes a long time to do properly. Tasks may be artificial & restricted. Data cannot always be generalized.
Expert	Strongly diagnostic. Provides a snapshot of entire interface. Few resources needed (apart from paying experts). Therefore, cheap. Can yield valuable results.	Subject to bias. Problems locating experts. Cannot capture real user behavior.

Important Concepts and Terms

Chapter Summary

- testing and evaluation are important activities to be performed as early as possible, and throughout the development cycle
- the emphasis should be on the user
  - user-centered design and evaluation
- testing and evaluation can be expensive, but fixing design flaws is much more expensive
- test and evaluation methods must be matched carefully with the specific situation