

Workflow Using the Discount Usability Engineering Method

Cynthia Shehata

This is not meant as a comprehensive study

I. Concept

Develop clear vision of the product

Identify project goals

1. Functional Requirements
What the system must be able to do
 - a. High level (abstract) - what the system should do
 - b. Low level (concrete) - ex.: operation of particular commands
2. Usability goals for the interface design
 - a. Determine who the user will be
 1. Task analysis - determine the cognitive characteristics system must meet
 - a. Search strategies currently used by users
 - b. Prerequisite knowledge required by users
 2. User analysis
Identify the scope of the user population using the system
 - a. Previous experience
 - b. Physical capabilities
 3. Environmental analysis of the situation in which the system is to be used
 - a. Physical environment
 - b. User support environment
 - b. Determine
 1. Most important features
 2. Tasks performed most often
3. Usability Requirements
 - a. Learnability: the time and effort required to reach a specified level of performance (ease of learning)
 - b. Throughput: the tasks accomplished by experienced users, the speed of task execution and the errors made (ease of use)
 - c. Flexibility: the extent to which the system can accommodate changes to the tasks and environments beyond those first specified
 - d. Attitude: the positive attitude engendered in users by the system
4. Create a "UI Roadmap" to document the preliminary analysis and concepts

II. Iterative Design Process

A. Scenarios

Get quick & frequent feedback from users

1. Proto-typing
 - a. Horizontal Proto-types
Reduce functionality to user interface surface layer
 - b. Vertical Proto-types
Reduce number of features but implement full functionality of those chosen
2. Implementation
 - a. Paper mock-ups
 - b. Simple proto-typing environment

B. Simplified Thinking Aloud

1. Ask real users to think out loud while performing some typical tasks
Maximum benefit reached between 3-5 users per test
2. Take notes (not video)

C. Heuristic Evaluation

User & Task Analysis

1. Several different people
 - a. Evaluate usability of GUI
 1. average correct
 2. task time
 3. subjective satisfaction
 - b. Judge compliance with recognized usability principles (see Usability Heuristics)
 - c. Identify problems
 - d. Novice vs. Expert User
 1. Learnability (novice)
 2. Speed of Use (expert) - responsiveness of system, shortcuts
2. Usability Heuristics
 - a. Visibility
 1. System status
Keep users informed about what is going on with progress indicator when longer than 7-10 seconds
 2. Controls
 - a. High contrast - most important elements
 - b. Other visual cues
 - c. Grouping
 - b. Match between system and the real world
 1. Speak the users' language, with words, phrases and concepts familiar to the user.
 2. Follow real-world conventions, making information appear in a natural and logical order
 3. Affordance - design should suggest functionality
 - c. User control and freedom
 1. Provide clearly marked "emergency exit"

2. Support undo and redo
- d. Consistency and standards
 1. Different words, situations, or actions should not mean the same thing
 2. Follow platform conventions
 3. Build upon user's prior knowledge of other applications
 4. On the basis of prior experience, user should have sense of how to interact with the next screen or control

See "List of Reserved Words," Figure 2 & "Guidelines for Using Controls," Figure 4, both at http://axp16.iie.org.mx/Monitor/v01n03/ar_ihc2.html)
- e. Error prevention
 1. Provide good error messages, but
 2. Careful design should prevent problems
- f. Rely on recognition rather than recall
 1. Make objects, actions, and options visible
 2. The user should not have to remember information from one part of the dialogue to another
 3. Instructions for use of the system should be visible or easily retrievable
- g. Flexibility and efficiency of use
 1. Accelerators can speed up interaction for the expert user such that the system can cater to both inexperienced and experienced users
 2. Allow users to tailor frequent actions
- h. Aesthetic and minimalist design

Dialogues should only contain relevant information. Every extra unit of information diminishes the relative visibility of relevant information.
- i. Help users recognize, diagnose, and recover from errors
 1. Express error messages in plain language
 2. Provide error codes for support desk
 3. Precisely indicate the problem
 4. Suggest a solution
- j. Help and documentation
 1. Easy to search
 2. Focused on the user's task
 3. List concrete steps to be carried out
 4. Not be too large
- k. Information Flow

If intuitive, users learn GUI faster

 1. To organize information in logical order, use
 - a. Position
 - b. Alignment
 - c. Grouping
 2. Position controls according to frequency of use
 - a. Most important field - upper left
 - b. Least important field - lower right
- l. Good Writing Style

Usability increases with concise, scannable text

 1. Highlighted keywords

2. Meaningful sub-headings
 3. Bulleted lists
 4. One idea per paragraph
 5. Inverted pyramid style - start with the conclusion
 6. Half the word count (or less) than conventional writing (assign to tech writers)
 - m. Avoid too many functions at the top level
 - but Features used frequently should be readily available
 - n. Provide traceable paths
 - o. Provide Keyboard & mouse support
- See Top Ten Mistakes in Web Design at <http://www.useit.com/alertbox/9605.html>

D. Design & Prototype

Create

1. Task-based design concept
2. Key screen prototype to illustrate it - conceptual model

E. Evaluate & Refine

1. Evaluate the prototype for usability
 - Get feedback from users who are like the ones who will be using the product
2. Iteratively refine and expand the design
3. Develop rules for repeatedly successful design

III. Build

A. Complete Detailed Design & Production

1. Complete the detailed screen design for the full program
2. Support late-stage changes

B. Evaluate & Refine

Statistical significance - indicate the probability of making the right decision

Even tests that are not statistically significant will improve probability of making correct decision

Any empirical study will help non-human factors people evaluate user interfaces (strive for excellence, not perfection)

1. Evaluate the complete prototype or early versions of the program for usability
 - Usability Metrics - requirements expressed in terms of performance measures, which are in turn detailed in the usability specifications
 - a. Time to complete a task
 - b. Percentage of task completed
 - c. Percentage of task completed per unit time (speed metric)
 - d. Ratio of success to failures
 - e. Time spent on errors
 - f. Percentage number of errors
2. Iteratively refine the design

IV. Release & Follow Up

- A. Plan and implement the introduction of the product to users, including final usability evaluations to ensure that the product has met the goals established at the beginning of the process
- B. Create and monitor feedback mechanisms to gather data for future releases
- C. Conduct Cost-Benefit Analysis
 - Estimate costs and benefits of
 - 1. Time spent on evaluation
 - 2. Increased usability
 - a. Reduction of learning time
 - b. Speed increase of expert performance

References:

http://www.useit.com/papers/guerrilla_hci.html
<http://www.useit.com/alertbox/9710a.html>
<http://www.cognetics.com/lucid/lucid.html>
http://axp16.iie.org.mx/Monitor/v01n03/ar_ihc2.html
<http://www.baddesigns.com/faq.html>
http://www.dcs.napier.ac.uk/~michael/hci_ohps/hci41n.html
http://www.dcs.napier.ac.uk/~michael/hci_ohps/hci444r.html
<http://www.wordfixers.no/process.html>
Internet Hall of Shame