TU/e technische universiteit eindhoven

Graphical User Interfaces

Prof. dr. Paul De Bra
Technische Universiteit Eindhoven
Universiteit Antwerpen

TU/e technische universiteit eindhoven

Part 3: User Interface Evaluation

- Predictive evaluation, without users:
  - cognitive walkthrough
  - action analysis
  - heuristic evaluation
- Evaluation with users:
  - user studies (in user's natural environment)
  - qualitative analysis, thinking aloud method
  - quantitative analysis, logging

TU/e technische universiteit eindhoven

What, Why, When to evaluate

- What: early design, simulations (scenario's), prototypes, finished product
- Why: to fix problems before product is shipped; to reduce costs by identifying problems early in the design process
- When: *throughout* the design and development process

TU/e technische universiteit eindhoven

Example: Evaluating the 1984 OMS

- Early tests of printed scenarios & user guides
- Early simulations of telephone keypad
- An Olympian joined team to provide feedback
- Interviews & demos with Olympians outside US
- Overseas interface tests with friends and family.
- Free coffee and donut tests
- Usability tests with 100 participants.
- A 'try to destroy it' test
- Pre-Olympic field-test at an international event
- Reliability of the system with heavy traffic

TU/e technische universiteit eindhoven

Evaluation without users

- Cognitive walkthrough:
  - experts play scenarios and verify they are believable
- Action analysis:
  - calculate timing of scenarios
- Heuristic evaluation:
  - check usability criteria

TU/e technische universiteit eindhoven

Cognitive walkthrough

- Focus on ease of learning
- Designer presents an aspect of the design & usage scenarios
- One of more experts walk through the design prototype with the scenario
- Expert is told the assumptions about user population, context of use, task details
- Experts are guided by 3 questions

The three questions

- Will the correct action be sufficiently evident to the user?
- Will the user notice that the correct action is available?
- Will the user associate and interpret the response from the action correctly?

As the experts work through the scenario they note problems

Who should do the walkthrough, when?

- For small piece you design: do a short walkthrough *in your head.*
- For larger scenario, for complete task, get a group of people together (including designers and users).
- Use a group of peers (same hierarchical level), not afraid to give criticism.

How to prepare for a walkthrough

- need a detailed description of how the interface will work
- need a task description of a whole task or a large part
- need a complete list of actions to complete the task with the interface
- need knowledge about the target end-users

What to do with results of walkthrough

- Fix the interface design!
  - make controls more visible
  - move controls
  - add or change labels
  - provide better feedback
- Difficult:
  - what if the user may not know which action to perform next (although it is visible)?

Action analysis

- Estimate the time needed to perform a task (by executing the correct series of actions using the interface)
  - **formal** action analysis: takes every atomic action of the user into account
  - **back of the envelope** action analysis: done quickly using rough estimates
- careful analysis has error margin of ±20%

Formal a a: physical movement

pressing a single key on keyboard (skilled typist 0.22, novel 1.2)	0.28 sec
move mouse pointer to point at an object on the screen	1.10 sec
clicking the mouse (or similar device)	0.20 sec
move hand from keyboard to mouse or to a function key	0.30 sec

Formal a a: visual perception

respond to a brief light (varies from 0.05 to 0.2 sec depending on intensity of the light)	0.10 sec
recognize a 6 letter word	0.34 sec
move eyes to new location on screen	0.23 sec

Formal a a: mental actions

retrieve simple item from long-term memory (command, password)	1.20 sec
learn a single "step" in a procedure (very bright people 10 to 15 sec)	25.0 sec
execute a mental step (like logical *and*, *or* or *implication*, varies a lot!)	0.075 sec
choosing among methods (varies a lot depending on complexity, 0.6 to 1.8)	1.20 sec

Back of the envelope analysis

- Rough analysis using *compound* actions (e.g. "select *cut* from the *edit* menu")
- Every compound item take 2 or 3 seconds
- Questions:
  - Can a "simple" task be done with a simple action sequence?
  - Can frequent tasks be done quickly?
  - How many facts and steps does the user have to learn?

Heuristics (Jakob Nielsen)

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

Strong points of heuristic evaluation

- Heuristic evaluation is *cheap* (when few evaluators participate).
- Evidence suggests that 5 evaluators identify 75-80% of all usability problems.
- Few ethical & practical issues to consider (no users struggling with bad prototypes)

Drawbacks of heuristic evaluation

- Good experts are hard to find and expensive (need expertise in interfaces and application domain).
- Trivial problems are identified but important problems may be missed.

Three stages of heuristic evaluation

- Briefing session to tell experts what to do
- Evaluation period of 1-2 hours in which:
  - Each expert works separately
  - Take one pass to get a feel for the product
  - Take a second pass to focus on specific features
- Debriefing session in which experts work together to prioritize problems

Evaluation with users

- Observation (in user's natural environment)
  - field tests, ethnographic research
- Thinking aloud method
  - let users explain what they are thinking and doing and why they are doing it
- Usability testing in controlled environment
  - quantitative evaluation (also qualitative but less)

Observation / Field studies

- Field studies are done in natural settings
- The aim is to understand what users do naturally and how technology impacts them.
- In product design field studies can be used to:
  - identify opportunities for new technology
  - determine design requirements
  - decide how best to introduce new technology
  - evaluate technology in use.

Observing, in the field

- Observing without interacting with the subjects, but they know you are there:
  - Notes & still camera
  - Audio & still camera
  - Video
  - Tracking users:
    - diaries
    - interaction logging

What to observe? (Goetz and LeCompte)

- *Who* is present?
- What is their role?
- *What* is happening?
- *When* does the activity occur?
- *Where* is it happening?
- *Why* is it happening?
- *How* is the activity organized?

What to observe? (Robinson)

- *Space*. What is the physical space like?
- *Actors*. Who is involved?
- *Activities*. What are they doing?
- *Objects*. What objects are present?
- *Acts*. What are individuals doing?
- *Events*. What kind of event is it?
- *Goals*. What do they to accomplish?
- *Feelings*. What is the mood of the group and of individuals?

Key points about observation methods

- Observe from outside or as a participant (can include interviews).
- Analyzing video and data logs can be time-consuming.
- In participant observation collections of comments, incidents, and artifacts are made.
- Ethnographers immerse themselves in the culture that they study.

Selecting users for evaluating prototypes

- Need representative for real users.
  - must be real volunteers
  - convince them the *system* is being tested, not the *users*
  - protect users' privacy and inform them
  - provide training to the level you expect from real users
  - first do a *pilot study* yourself or with colleagues

Select a task to perform

- Select a task that real users will perform with the final product. Modifications are allowed:
  - the real task may take too long, so you simplify something or take a part of the task
  - the real task may require knowledge the test users don't have (careful here!)
  - do not select tasks because you think the design supports best

Provide a test system

- Should be reasonably close to final product.
  - the functions needed for the selected task should be implemented
  - some restrictions are acceptable (e.g. choice of file names)
  - errors by the user may result in a message instead of performing the wrong action
  - use throw-away prototype for missing parts

Choose what to evaluate

- Thinking aloud method:
  - let users say what they think and do.
  - qualitative analysis: find out "why" they do things
- Usability testing (logging):
  - let users work and measure performance
  - also qualitative: find errors they make

Thinking aloud method

- Ask users: "Tell me what you are thinking about as you work".
- Remind users when they stop talking.
- Videotape or take notes (fast).
- Make a list of difficulties users encountered.
- Compare how you *thought* the interface would work with test results; *rethink* the design.
- Users will think more carefully than normal.
- Users will say things that make no sense or are wrong when they don't know what to say.

Measuring "bottom line" usability

- Involves recording typical users' performance on typical tasks in natural or controlled settings.
- Record on video & log key presses.
- Calculate performance times, identify errors & help explain why the users did what they did.
- User satisfaction questionnaires & interviews are used to elicit users' opinions.
  - Careful: after the experiment users may not remember

What to measure?

- Time to complete a task
- Time to complete a task after a specified time away from the product
- Number and type of errors per task
- Number of errors per unit of time
- Number of navigations to online help or manuals
- Number of users making a particular error
- Number of users completing task successfully

Comparing prototypes

- *Between-groups*: two groups, one test prototype A, the other tests B
  - problems with differences between groups
- *Within-groups*: one group first tests prototype A, then B
  - problem: users learn from prototype A
- Perform statistical analysis to verify whether observed differences are relevant.

Problems with user testing

- In principle the designer/evaluator must not interfere with the experiment (video)
  - do not let users suffer indefinitely: help them overcome a difficult step or let them stop
  - take the failure into account in bottom-line data
  - ask questions in a debriefing interview (but no suggestive questions)
  - take users back to the interface to explain

Evaluation assignment

- Evaluate the Dutch Universities' UFO system, using the following scenario:
  - You are manager of a small group of people: a teacher of simple html and Word courses, a teacher of complex software courses (matematica, spss, etc.), a programmer and a secretary. You should determine the function and corresponding salary level using the UFO system. You should produce a report for each of them (by transferring data from UFO to a word processor).
  - We provide a more detailed description of the four people later (in Dutch).

Evaluation assignment (cont.)

- First observe (informally) how UFO works. Then do a *cognitive walkthrough* of the whole task. Write this down.
- Do a formal and back of the envelope action analysis for the evaluation process of a single employee. Add the time the client and server need to process the requests! (UFO can be slow)
- Perform a thinking-aloud experiment. Take notes and report the problems encountered.
  (Note: expect to encounter a lot of problems.)
- Measure the total time for the whole task (for all four the employees).

Evaluation assignment (cont.)

- The UFO site is available at: http://web14.e-office.com/vsnu/fws.nsf/wwwlogin?openform For this project a password is required. It will be issued during the lecture.
- Tip: have one person study the interface carefully and act as expert who can provide help when needed; one is end-user, one is observer ⇒ group of 3 needed.
- **deadline: December 22, 2006, 2.30pm**
- **format: paper document**
- **place: secretary building G 1st floor**

De Secretaresse (m/v)

- Ze beheert de agenda van de medewerkers. Cursusdata en – tijden worden in overleg met de groepsleider gepland, maar alle overige afspraken regelt ze zelfstandig, met mensen van binnen en buiten de organisatie.
- Ze communiceert via email met medewerkers over in- en uitgaande correspondentie, en stelt zelf brieven op volgens aanwijzigingen van de medewerkers.
- Ze regelt zalen, catering, inschrijvingen, betalingen en andere zaken voor de verzorgde cursussen en voor vergaderingen.
- Daarnaast doet ze ook zaken als koffie zetten, plantjes water geven en telefoon beantwoorden.

Docent A

- A ontwikkelt cursussen over eenvoudige ICT onderwerpen als HTML en Word. Hij raadpleegt boeken en on-line materiaal.
- Hij overlegt met de afnemers over de behoeften en het niveau, en ook de achtergrond van de afnemers om gepaste voorbeelden te bedenken.
- Hij verzorgt de cursussen op het in overleg bepaalde tijdstip en plaats.
- Hij houdt enquêtes met als doel de cursussen te verbeteren.
- Een deel van de productie van het cursusmateriaal besteedt hij uit aan de secretaresse.

Docent B

- A ontwikkelt cursussen over moeilijke ICT onderwerpen als Matematica en SPSS. Hij raadpleegt boeken en on-line materiaal.
- Hij overlegt met de afnemers over de behoeften en het niveau, en ook de achtergrond van de afnemers om gepaste voorbeelden te bedenken.
- Hij verzorgt de cursussen op het in overleg bepaalde tijdstip en plaats.
- Hij houdt enquêtes met als doel de cursussen te verbeteren.
- Een deel van de productie van het cursusmateriaal besteedt hij uit aan de secretaresse.

De Programmeur

- Hij ontwikkelt software in opdracht van de groepsleider en de docenten, bedoelt om het verzorgen van de cursussen te ondersteunen.
- Hij stelt in overleg met de docenten de specificaties van de software op.
- Hij implementeert de software in de volgens hem meest geschikte programmeertaal.
- Hij komt ook zelf met nieuwe ideeën voor voorbeelden en ondersteunende software.
- Hij is nauw betrokken bij het bepalen van de benodigde ondersteuning bij nieuw te ontwikkelen cursussen.