## Evaluation and Testing

- as HCI is not hard science, it has to rely on evaluation and testing to improve itself, and try and learn more
- evaluation is to HCI as testing is to software engineering: the interface is tested against the need and practices of the user
- it is important to have in mind a precise goal to test ("this interface is nice" is not a clear parameter!)
- common problems:
  - rely on common sense
  - systems are for people, and designers are people!
  - not consulting the real, end user
  - "it is easier to build it this way"
  - new is cool!

## Evaluation

WHAT IS IT?

- *Evaluation is concerned with gathering information about the usability or potential usability of a system, in order either to improve features within an interface and its supporting material, or to assess a completed interface.*

WHAT SHOULD IT ACCOUNT FOR?

- the experience level of users
- the types of tasks to be undertaken
- the system being used
- the environment in which the study takes place

WHEN TO DO IT?

- *Formative evaluation:* before implementation, to influence the final product
- *Summative evaluation:* after implementation, as a test

## Evaluation methods

1. Analytic evaluation
   - using formal or semi-formal interface description to predict user performance
2. Expert evaluation
   - involving experts in the evaluation
3. Empirical evaluation
   - make experiments letting a sample of people use the system and observe them solving particular tasks
   - requires an hypothesis to test
4. Survey evaluation
   - ask directly the users what they think
   - less formal, but helpful to collect opinions rather than facts
5. Experimental evaluation
   - uses scientific experimental practice to test hypotheses

## Analytic evaluation

- based on formal description of:
  - the task (use task analysis techniques, e.g. GOMS model)
  - the interface (use a user interface definition, e.g. StateCharts)
- the interface is analysed "on paper", by predicting performances and errors in terms of the physical and cognitive operations that have to be performed
- ADVANTAGES:
  - it can start before actually implementing the interface
  - few resource required (cheap!)
- DISADVANTAGES:
  - narrow focus
  - broad assumptions of users' cognitive operations
  - limited guidance on how to use methods, therefore can be difficult for the evaluator

## Expert evaluation

- "experts" (i.e. people experienced in interface design, or human factors research, or both) take the role of less experienced users and describe potential problems they foresee with the interface
- ADVANTAGES:
  - few resource needed: a small number of experts can usually identify a whole range of problems
  - effective: often little prompting is needed to experts in order to identify problems (real users can suggest solutions only to some of the problems encountered)
- DISADVANTAGES:
  - restrictions in role playing: experts need to know lots of information on the knowledge level of the potential users
  - subject to bias
  - problems locating experts in both HCI and the system application
  - cannot capture real users' behaviour

## Empirical evaluation

- collecting data that provide information about what users do when they interact with the interface
- main techniques:
  - observations (direct or video recording)
  - software logging
- ADVANTAGES:
  - quickly highlights problems
  - verbal protocols valuable source of information (e.g. for user models)
  - can be used for rapid iterative development
- DISADVANTAGES:
  - environment different from typical working situation
  - analysis of data can be time and resource consuming

## Observations

- various techniques:
  - think aloud: users are asked to elaborate what they are doing, describing what's happening, if they found any problem etc
  - post-task walkthrough: to collect more critical information by a subsequent interpretation of what the user did and why
- better to prepare a set of task that the user should perform
- do not interfere
- Hawthorne effect: people can act differently when they are being watched

## Survey evaluation

- ask users subjective opinions
- techniques:
  - interviews
  - questionnaires
- ADVANTAGES:
  - addresses users' opinions and understanding
  - can be applied to users and designers
  - questions can be tailored to the individual
  - provide quantitative results
  - can be used on a large group of users
- DISADVANTAGES:
  - user experience is important
  - low response rates (esp. mailed questionnaires)
  - possible response bias
  - analysis can be complicated and lengthy
  - interviews very time consuming

## Questionnaires

- can help collect lots of information, but it's time consuming both to prepare and to analyze
- can use open questions ("what did you like of the interface most?") and/or closed questions ("what did you like of the interface most? (a) the colours, (b) the sound, (c) the ease of moving from area to area")
- not too long (use filter questions to allow users ignore some parts)
- ranked questions:

| Place the following ways of issuing a command in order of easiness | | |
|---|---|---|
| Menu selection ◯ | Command line ◯ | Control key accelerator ◯ |

- scalar questions:

| It is easy to recover from mistakes | | | | |
|---|---|---|---|---|
| Strongly agree | Agree | Neutral | Disagree | Strongly disagree |

## Experimental evaluation

- a number of factors associated with the interface are manipulated and the effect of this manipulation on the user is studied
- usually, the number of factors is limited, so that causal relationships can be established
- ADVANTAGES:
  - powerful method
  - quantitative data for statistical analysis
  - can compare different groups of users
  - reliability and validity good (theory is sound)
  - replicable
- DISADVANTAGES:
  - high resource demands
  - requires knowledge of experimental methods
  - tasks can be artificial and restricted
  - cannot always generalize to full system in typical working situation

## Experiments

- choose subjects: try and match the expected user population
- choose the variables: there are three kind of variables to consider
  1. independent variables (what we use): interface style, types of help, icon design, etc.
  2. dependent variables (what we want to measure): speed of menu selection, number of errors, how many times help is invoked etc.
  3. confounding variables (what we try to avoid): expertise of users
- formulate an hypothesis:
  - the independent variable is affected by the dependent variable ("users of a word processing system need less help with an iconic interface")
  - there will be no relation between the two variables ("there will be no difference between users using an iconic interface and users using a standard, menu interface")

## Key Differences among Evaluation Methods

| **Method** | **Stage** | **User** | **Task** | **Data type(*)** |
|---|---|---|---|---|
| Analytic | specification | no users | Specified | quantitative |
| Expert | specification or prototype | no users: experts play role | no restriction | qualitative |
| Empirical | simulation or prototype | real users | no restriction | quantitative and qualitative |
| Survey | simulation or prototype | real users | no restriction | quantitative and qualitative |
| Experimental | full prototype | real users | no restriction | quantitative and qualitative |

(*) Quantitative data deal with either user performances or attitudes that can be recorded in a numerical form. Qualitative data focus on reports and opinions that may be categorized in some way but are not reduced to numerical values.

## Usability engineering

**Usability:** *Usability is a measure of the ease with which a system can be learned or used, its safety, effectiveness and efficiency, and the attitude of its users towards it.*

**Usability engineering:** an approach to system design in which the usability level of a system is specified quantitatively in advance, using metrics

## Aspects to consider

Key aspects of interest in usability testing are:

**learnability:** the time and effort required to reach a specified level of user performance (“ease of learning”)

**throughput:** the task accomplished by experienced users, the speed of task execution and the errors made (“ease of use”)

**flexibility:** the extent to which users can adapt a system to new ways of interaction as they become more experienced

**user satisfaction:** the positive attitude the users have towards the system

## More specifically

- learnability
  - predictability: determine the effects of future actions based on previous experience
  - synthesizability: assess the effect of past operations on the current state
  - familiarity: how user’s knowledge of the real world or other, more familiar systems can be applied when interacting with the new system
  - consistency: likeness in I/O behaviour arising from similar situations
- throughput
  - robustness: recoverability, responsiveness (communication between system and user), task adequacy (how the system supports all the user wishes to do)
- flexibility
  - dialogue initiative (maximize user’s, while avoiding too much freedom), multi-task, task migratability (cooperation user/system), customizability

## Some parameters to measure learnability

- the time required to learn the system,
- the time required to achieve a stated performance criterion,
- the difficulties observed in acquiring the necessary skill,
- the frequency of error messages,
- the frequency of particular error messages,
- the frequency of use of on-line help,
- the number of times the user needed help on a specific problem,
- user comments, suggestions and preferences.

## Some parameters to measure throughput

- the time requested to perform selected tasks,
- per cent of task completed
- the success or failure rate in completing tasks,
- the frequency of use of various commands or of particular features/functions,
- the time spent looking for information in the documentation,
- the time spent using on-line help,
- the time spent (or number of actions performed) to undo an erroneous command
- number of repetitions of failed commands,
- the number of commands not used,
- the measurements of user problems as for learning, and user comments, suggestions and preferences.

## The process of usability engineering

1. define the attribute to measure
2. establish a metric to measure it
3. identify the subset of users it applies to
4. define the pre-conditions for measurement
5. identify the usability specifications:
    - worst case
    - lowest acceptable level
    - planned case
    - best case
    - current level

## An Example of a Usability Specification Table

| | |
|---|---|
| Attribute | Throughput |
| Metric | Number of errors |
| User set | All users |
| Pre-conditions | After 1 week of use |
| Worst case | More error than at present |
| Lowest acceptable level | Same errors as at present |
| Planned case | 2 errors every hour |
| Best case | Error free |
| Current level | Six errors per hour |

## Comparing usability principles

Usability of a system can be measured *per se* or with respect to:

- another existing system, or previous versions of the system
- a competitive system
- carrying out the task without use of a computer system
- a prototype
- user's own earlier performances
- another component of the same system
- another test performed previously