Evaluation of Interactive Systems
Inspection methods
Inspection Methods

Walkthroughs

Norman and Nielsen’s usability heuristics

Green and Blackwell’s cognitive dimensions

Predictive models (we will get back to it later in this course)

e.g., GOMS models
Walkthrough

A step-by-step review of a computer program or system during its design to search for errors and problems.

- source of a program - to find the 'bugs'
- design of a system - to understand the structure
- text (scientific papers) - to verify the structure and comprehensibility
- experiments - to verify the details of the method used
Cognitive Walkthrough

A group of evaluators “walkthrough” an interface to list as many problems of usability as possible using some evaluation criteria.

Choose a small group with different roles and expertise.

Establish a duration time, up to 1 hour.

Choose a presenter that explains the scenario of use, each action at a time.

Choose the level of critique (system, interface, specific component).
Usability problems

Examples of critics / comments

Specific
   e.g., It takes three steps to make a simple search

Missing functionality
   e.g., No help, need to search outside

Bugs
   e.g., The import functionality of X does not work

Suggestions
   e.g., An overview of all data created is needed

General (the least useful type)
   e.g., Difficult to use too many icons
Usability Heuristics

Source Nielsen Norman Group http://www.nngroup.com/
Evaluation criteria
Norman & Nielsen’s 10 Usability Heuristics

1. Visibility of system status

   The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

2. Match between system and the real world

   The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
Evaluation criteria
Norman & Nielsen’s 10 Usability Heuristics

3. User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

4. Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
5. Error prevention
   Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

6. Recognition rather than recall
   Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
Evaluation criteria
Norman & Nielsen’s 10 Usability Heuristics

7. Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can fit both inexperienced and experienced users. Allow users to tailor frequent actions.

8. Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
Evaluation criteria
Norman & Nielsen’s 10 Usability Heuristics

9. Help users recognize, diagnose, and recover from errors

   Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

10. Help and documentation

   Any help information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.
“Iterating through at least three versions of the interface is recommended, since some usability measures often decrease in some versions if the usability engineering process has focused on improving other parameters.” [2]

Heuristic evaluation evaluators

Who are the evaluators?
usability engineers, end users, experts in evaluation

How many evaluators?
5 evaluators find around 75% of the usability problems

Recruiting more evaluators reduces the ratio costs/benefits
Heuristic evaluation evaluators

A single evaluator identifies (Average over six case studies)

- 35% of all usability problems
- 42% of the major problems
- 32% of the minor problems

Not great, but finding some problems with one evaluator is much better than finding no problems with no evaluators.
Heuristic evaluation evaluators

Evaluators miss both easy and hard problems

‘best’ evaluators can miss easy problems

‘worse’ evaluators can discover hard problems
Nielsen’s recommendation: 5 evaluators but working individually on the same usability test

evaluation is not influenced by others

independent and unbiased (unlike the focus group technique)
greater variability in the kinds of errors found

no overhead required to organize group meetings
Usability Heuristic #1

Visibility of system status

Continuously inform the user about

what the system is doing

how the system is interpreting the user’s input

Be as specific as possible, based on user’s input
Usability Heuristic #1
Visibility of system status

Especially important when the interactive system features several modes

What did I select?
What mode am I in now?
How is the system interpreting my actions?
Usability Heuristic #1
Visibility of system status

System Response time (time to give feedback)

- < 0.1s  perceived as “instantaneous”
- 1s  user’s flow of thought stays uninterrupted, but delay noticed
- 10s  limit for keeping user’s attention focused on the dialog
- > 10s  user will want to perform other tasks while waiting
Usability Heuristic #2
Match between system and the real world

Speak the user language

Apple iTunes uses the library metaphor for media files as opposed to a collection of computer files browsed through the Apple Finder.

Folded page effect
Usability Heuristic #3
User control and freedom

Users don’t like to feel trapped by the computer!

Strategies:

- Cancel button (for dialogs waiting for user input)
- Universal Undo (can get back to previous state)
- Interrupt (especially for lengthy operations)
- Quit (for leaving the program at any time)
- Defaults (for restoring a property sheet)
Usability Heuristic #3
User control and freedom

Undo support

Bad example: window management (the user can not restore the size or the location of a window he has just manipulated), view navigation (the user can not navigate back to a portion of a document he was looking at)
Usability Heuristic #3
User control and freedom

Generic undo support for drag-and-drop

Dwell and Spring: Undo for Direct Manipulation

Caroline Appert    Olivier Chapuis    Emmanuel Pietriga

CHI 2012
Usability Heuristic #4
Consistency and Standards

Use meaningful mnemonics, icons & abbreviations

e.g., File / Save   (terminology)

Ctrl + S     (keyboard shortcut)

(icon)
Usability Heuristic #4
Consistency and Standards

Consistent effects

commands, actions have same effect in equivalent situations

Consistent language and graphics

same visual appearance across the system (e.g. widgets)

same information/controls in same location on all windows
Usability Heuristic #5
Error prevention

Constrain input (disabling widgets on right time, auto-completion...)

Good example: Google Auto Recommend cuts down on mis-spellings
Usability Heuristic #6
Recognition rather than recall

Computers are good at remembering, people are not.

Promote recognition over recall

rely on visibility of objects to the user (but less is more...)

Good example: font preview
Usability Heuristic #7
Flexibility and efficiency of use

Experienced users want to perform frequent operations quickly

keyboard and mouse accelerators

command completion

navigation jumps and search

(e.g. bookmarking)

history...
Usability Heuristic #8
Aesthetic and minimalist design

Less is more...

Google Scholar new look
http://googlescholar.blogspot.fr/2012/05/our-new-modern-look.html

De-clutter Your Interface With Hover Controls
http://usabilitypost.com/2008/11/19/de-clutter-your-interface-with-hover-controls/
Usability Heuristic #9
Help users recognize, diagnose, and recover from errors

Provide immediate feedback with specific instructions
Usability Heuristic #10
Help and documentation

Help is not a replacement for bad design

Users do not read manuals, they prefer to pursue their task

Simple systems
  walk up and use; minimal instructions

Most other systems
  simple things should be simple
  learning path for advanced features
Usability Heuristic #10
Help and documentation

Help when users are in some kind of panic

online documentation specific to current context with good search/lookup tools

Help for getting started

short guides demonstrating very basic principles through quick working examples (videos, exercises, “tours”...)

Help for quick reference

completion for syntax of actions, list of shortcuts...
Heuristic evaluation
pros and cons

Advantages

- cheap and fast way to inspect a system
- can be done by usability experts, double experts, and end users

Problems

- principles can’t be treated as a simple checklist
- subtleties involved in their use
  - for example, satisfying “less is more” and “recognition rather than recall” at the same time may be challenging…

Source of illustration http://www.nngroup.com/articles/form-design-placeholders/
Cognitive Dimensions

What for

Check-list approach for evaluating usability of information-based artefacts:

not only *interactive systems* (word-processors, graphics packages, mobile telephones, …)

but also *static systems* (graphs, music notation, programming languages, …)

focus on usability aspects that make learning and doing hard for mental (not physical) reasons
Approach

Cognitive Dimensions (CDs) are ‘discussion tools’

CDs name concepts (lexicalisation)

CDs are not powerful mathematical tools for deep analysis (≠ modeling approaches like GOMS)

Complementary to heuristic evaluation

  looks at other usability aspects

  probably easier to learn

  takes the type of activity into account
Approach

The information artefact is evaluated along different dimensions, giving a profile

The profile determines the suitability for one of the four types of activity

incrementation: adding a new card to a cardfile; adding a formula to a spreadsheet

transcription: copying book details to an index card; converting a formula into spreadsheet terms

modification: changing the index terms in a library catalogue; changing the layout of a spreadsheet; modifying the spreadsheet to compute a different problem

exploratory design typographic design; sketching
Dependence between CDs

CDs are conceptually independent for real entities, some dimensions are only ‘pairwise’ independent (trade-off/compromise between two dimensions)

Take a constant mass of gas, with 3 ‘dimensions’: temperature, pressure and volume. If you heat the gas, the temperature rises, and it tries to expand. If the volume is held constant, it can’t expand, so the pressure increases. If the pressure is to be kept constant, the volume must increase.

So although pressure, temperature, and volume are conceptually independent, for physical objects they are only pairwise independent.
## CDs list

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstraction</td>
<td>types and availability of abstraction mechanisms</td>
</tr>
<tr>
<td>Hidden dependencies</td>
<td>important links between entities are not visible</td>
</tr>
<tr>
<td>Premature commitment</td>
<td>constraints on the order of doing things</td>
</tr>
<tr>
<td>Secondary notation</td>
<td>extra information in means other than formal syntax</td>
</tr>
<tr>
<td>Viscosity</td>
<td>resistance to change</td>
</tr>
<tr>
<td>Visibility</td>
<td>ability to view components easily</td>
</tr>
<tr>
<td>Closeness of mapping</td>
<td>closeness of representation to domain</td>
</tr>
<tr>
<td>Consistency</td>
<td>similar semantics are expressed in similar syntactic forms</td>
</tr>
<tr>
<td>Diffuseness</td>
<td>verbosity of language</td>
</tr>
<tr>
<td>Error-proneness</td>
<td>notation invites mistakes</td>
</tr>
<tr>
<td>Hard mental operations</td>
<td>high demand on cognitive resources</td>
</tr>
<tr>
<td>Progressive evaluation</td>
<td>work-to-date can be checked at any time</td>
</tr>
<tr>
<td>Provisionality</td>
<td>degree of commitment to actions or marks</td>
</tr>
<tr>
<td>Role-expressiveness</td>
<td>the purpose of a component is readily inferred</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
Viscosity

Resistance to change: the cost of making small changes

*Repetition* viscosity: a single goal-related operation on the information structure (one change 'in the head') requires a high number of individual actions

*Knock-on* viscosity: one change 'in the head' entails further actions to restore consistency
Repetition viscosity

Typically when a structure exists in the user’s mind but is not recognized by the system

A collection of files making up one document in the user’s mind may need to be edited to bring their typography into conformance, usually by editing each file individually.

Manually changing the headers format in a word processor without the notion of editable style.
Knock-on viscosity

Typically in structures that have high inter-dependencies

Moving one slot in a set of interconnected schedules will require many changes

<table>
<thead>
<tr>
<th>Year</th>
<th>9.00-10.00</th>
<th>10.00-11.00</th>
<th>11.00-12.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year 1</td>
<td>Mr Adams</td>
<td>Ms Burke</td>
<td>Ms Cooke</td>
</tr>
<tr>
<td>Year 2</td>
<td>Ms Cooke</td>
<td>Mr Davis</td>
<td>Mr Adams</td>
</tr>
<tr>
<td>Year 3</td>
<td>Ms Burke</td>
<td>Mr Adams</td>
<td>Mr Davis</td>
</tr>
</tbody>
</table>
Viscosity and activity type

<table>
<thead>
<tr>
<th>viscosity</th>
<th>transcription</th>
<th>incrementation</th>
<th>modification</th>
<th>exploration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>acceptable</td>
<td>acceptable</td>
<td>harmful</td>
<td>harmful</td>
</tr>
<tr>
<td></td>
<td>(because one</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>shouldn’t need</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>to make changes)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Remark: Viscosity is not always harmful: increased viscosity may encourage deeper reflection and better learning (e.g., good for safety-critical systems)
Lowering viscosity

Introduce a new abstraction

Change the notation, usually by shifting to a relative information structure instead of an absolute one

...
Lower viscosity
Powerful selection mechanisms
Lower viscosity

Styles
Lower viscosity
Connectors
Premature commitment

Constraints on the order of doing things force the user to make a decision before the proper information is available.

Anticipating the right letter size to sign in a limited space box.

The amateur signwriter’s target notation contains a constraint relating the width of the sign-board and the width of the wording, but until the sign was written he or she did not know how wide the wording would be – so they made a guess (premature commitment).

Voice menu systems: the user has to decide whether to select an option before knowing what other choices are available.

“If you wish to make a booking, press 1 ...if you wish to reserve a seat, press 2 ... etc”
Premature commitment and activity type

<table>
<thead>
<tr>
<th></th>
<th>transcription</th>
<th>incrementation</th>
<th>modification</th>
<th>exploration</th>
</tr>
</thead>
<tbody>
<tr>
<td>premature commitment</td>
<td>harmful</td>
<td>harmful</td>
<td>harmful</td>
<td>harmful</td>
</tr>
</tbody>
</table>

Problematic in all contexts, except where the lookahead is not extensive.
Lowering Premature commitment

Decoupling: e.g., sign somewhere with an arbitrary size and scale/translate to fit a given box

Ameliorating: a low viscosity reduces the cost of premature commitment, since bad guesses can be easily corrected

“Deconstraining”: removing constraints on the order of actions
Lower premature commitment
Edit container once content added
Lower premature commitment

Editable histories
Hidden dependencies

A relationship between two components, but that is not fully visible: the cost of searching it

- HTML links: may become fossils when they point at deleted websites

- Rendering of a document depending on the version of the editing software
Hidden dependencies and activity type

<table>
<thead>
<tr>
<th>Hidden dependencies</th>
<th>transcription</th>
<th>incrementation</th>
<th>modification</th>
<th>exploration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>acceptable</td>
<td>acceptable</td>
<td>harmful</td>
<td>acceptable for small tasks</td>
</tr>
</tbody>
</table>
Reducing cost of hidden dependencies

Add cues to the notation

Highlight different information

If they have been hidden once, keep them always hidden (e.g., ensure backward compatibility)
Reduce cost of hidden dependencies
make them predictable by the user

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

Values of C cells depend on values of A and B cells
Reduce cost of hidden dependencies
Warn users in case of repercussions on other objects
Exercise
Walkthrough

You want to:

Schedule interviews (2 groups of 5) - [http://www.youcanbookme.com/](http://www.youcanbookme.com/)

Task: create a schedule for booking time slots for interviews

Distribute a questionnaire - (2 groups of 5) [https://docs.google.com/forms/](https://docs.google.com/forms/) or (2 groups of 5) [https://fr.surveymonkey.com/](https://fr.surveymonkey.com/)

Task: create a form to collect demographic data for each participant (name, age, gender) and include a question of each type (a - closed question with multiple choice, b - closed question with Likert-scale, c - closed question where the interviewee has to rank different choices, d - closed question with multiple choice and an open comment)
For Wednesday Oct. 17

• Prepare a series of slides that you are able to present in 5 minutes

• a typical slide is a screenshot illustrating the usability issue, with the name of the usability heuristics / cognitive dimension used to identify this problem and a short text description detailing why

• Send me your slides (appert@lri.fr) in PDF, with the names of all students in the group