The Design of Interactive Computational Media

Class 4: 2 Oct. 2002

User-centred, Iterative Design (Hour 1)

Learning from Design (Hour 2)

User-centred, Iterative Design

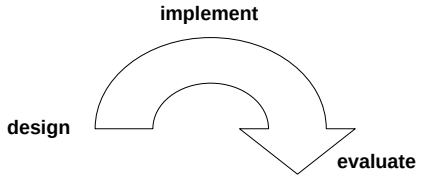
- User-centred design
- Iterative design
- Design example: Movie Authoring and Design system
  - Design process (relate to Rosson and Carroll)
  - History
  - Demo
  - Evolution of the design

User-centred design

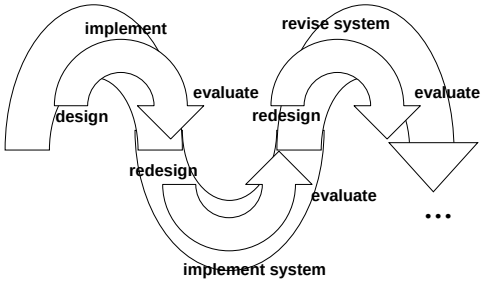
- Requires an early focus on users and their tasks prior to system design and *throughout entire design process*
- Need to understand potential users
  - Not just identifying and characterizing them
- Do this through direct contact with users
  - Questionnaires and interviews
  - Observations of current work *and work with new artifacts*
  - Task analysis
  - User(s) on design team

Iterative design

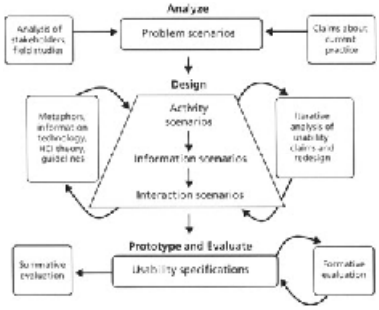
- Need to iteratively design, prototype, analyze/evaluate
- Design —>Prototype- —>Evaluate —>Redesign —>Implement —>Evaluate —>Redesign —>Revise implementation —>Evaluate —>etc..



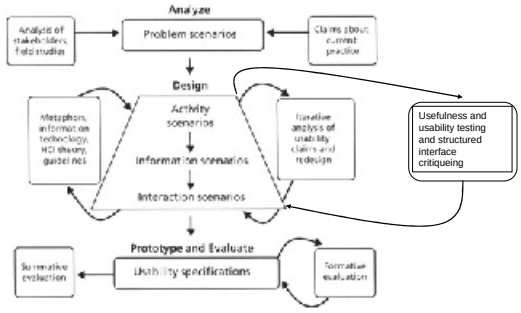
The iterative design process



Rosson & Carroll's Design Process



Augmented Design Process



Design process in tabular form

	DESIGN	IMPLEMENT	ANALYZE AND EVALUATE
Information collection and requirements analysis	Reflections, studies, classes on filmmaking	No "Problem Scenario" developed (weakness)	Some contact with real filmmakers (should have had more contact)
Activity, information & interaction design	Initial design concepts	Design sketches, Director prototypes, small C programs	Feedback only from research group (weakness)
Prototyping and prototype system	System functionality and look-&-feel	Critical mass C prototype	Demos, first real projects, observations, filmmaker interviews
Production prototype and its evolution	Complete system, incorporating evaluation insights	Implementation of significantly useable C++ system	More demos, real projects, observations, interviews, multimedia summer camps
Production system and its evolution	Deliverable system, incorporating evaluation insights	Java implementation	Intensive internal use, beta testing, client use

Movie Authoring and Design (MAD) System: The Origin

- Roughly 1990... when I first saw Apple's QuickTime
- Making movies is hard, insufficient technology support
- Contrast to word processing and document creation

- Will sketch history of the project over a decade, how the design has evolved, and how users have helped

MAD concept

- Hierarchic, outline-based script and movie organization
- Top-down design, bottom-up implementation of movie
- Integration of multiple media, multiple views and representations
  - Text (represented in a script)
  - Voice (dialogue and narration), music, sound effects
  - Sketches (represented in a storyboard)
  - Recorded video
- Playback of "approximate" movie at any time

Sample movies

QuickTime™ and a PNG decompressor are needed to see this picture.

QuickTime™ and a Animation decompressor are needed to see this picture.

QuickTime™ and a Animation decompressor are needed to see this picture.

Sample movies

QuickTime™ and a Animation decompressor are needed to see this picture.

QuickTime™ and a Animation decompressor are needed to see this picture.

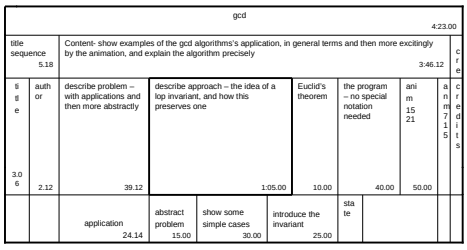
Live demo

MAD system information collection, requirements analysis

- Reflections on my experiences as a moviemaker
- Analysis of contrasts between producing a film and:
  - Outlining a report
  - Writing a document
  - Creating electronic music
  - Writing a large program
- Student Alan J Rosenthal takes OCA film production class, creates movie called *How To Hyoop* (1991)

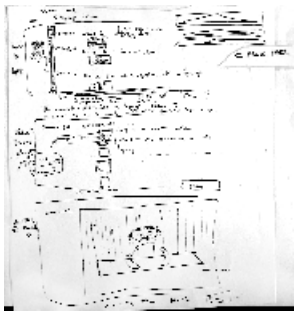
MAD concept design

- First "sketch": Diagram of hierarchic structure for "greatest common divisor" film (Nov. 91)



MAD concept design

- Second "sketch": Script view, expanded script view, playback view (Mar. 92)

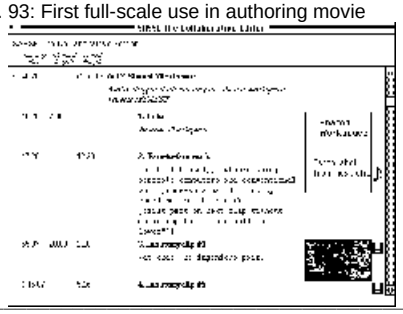


MAD concept design (continued)

- Apr. 92: Canned demo, Macromedia Director prototype
- May 92: Outline processor partial implementation in C
- June 92: Outliner control strategies Director prototype

MAD prototype system

- 92-93: Further development of C implementation, first demos, interviews with these informants
- Sep. 93: First full-scale use in authoring movie



MAD prototype system

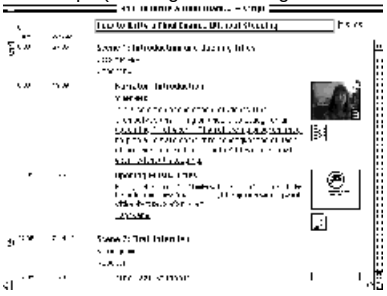
- First half of 1994: Use on other small projects including with a few school children, evaluation of use
- Summer 1994: Some redesign of interface, added functionality
- Winter 1994-5: First demos to film and video community

MAD production prototype (first "deliverable" system)

- Fall 1994: Major redesign of new interface incorporating what was learned, e.g.,
  - (Start time, duration, end time) --> (Start time, duration)
  - Storyboard view in addition to script view
  - Full-screen playback, ability to record movie onto videotape
  - Need for titles
- 1995: Reimplementation in C++ of most significant system classes to allow malleability and further expansion
- 1995: First rethinking of markets and customers --> kids, amateur filmmakers, education

MAD production prototype

- 1996-7: Continued usage for projects by youngsters and children, demos, design discussions, multimedia summer camps (first large scale usage with children)



MAD production system

- 1996-9: Redesign, with help of a graphic designer; reimplementation of MAD v. 3 in Java

MAD production system

- Incorporation of what was learned...
  - Simplification of hierarchy to {acts, scenes, shots}
  - Improvements to the film editor --> {clip, in-out, selection}
  - Easier and safer management of media resources
- Also design of beta user feedback and evaluation system for the commercial product (Expresto Software's CineKit)
  - A Web community
  - Documentation and training materials
  - The Users Forum
  - The Users Electronic Newsletter
  - The human support system

CineKit —> Creator

- General-purpose tool for horizontal market —> Special-purpose tool for vertical (eCRM) market
- Features for
  - Web publishing
  - Media management

Summing up

- Issues and opportunities
  - Need for evaluation of sustained real use
  - Extensions for more metaphors and viewpoints
  - Extensions to collaboration at a distance
  - Other vertical markets, e.g., multimedia messaging
  - General-purpose vs. special-purpose?
- Conclusions
  - Iterative design
  - User-centred design
    - We did some, BUT SHOULD HAVE DONE MORE!!!
  - Multidisciplinary design

Questions and Discussion

Break

Learning from Design

- Design, interfaces, and everyday things
- Problems with the interfaces to everyday things
- Key concepts (from Don Norman)
- Computers (embedded) as everyday things
- Visual examples
  - Staterooms of ships
  - Things for seniors
- The bottom line

Class Discussion

- *What is good design?*
- *Where do we see it in our day-to-day lives?*
- *Consider Bahen and Sandford Fleming buildings as examples ...*

Design, interfaces, and everyday things

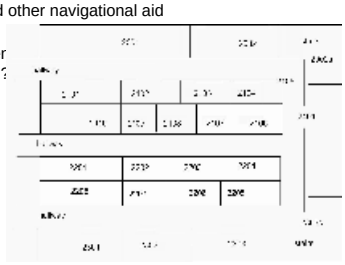
- Interfaces are everywhere
  - All “everyday things” (all 20,000 of them) have interfaces
  - See Norman's *The Psychology of Everyday Things*
- We learn by observing & reflecting on these interfaces
  - We can become more sensitive and insightful observers
  - This will help us become better designers

Everyday thing interface problems

- *SST Death Flight* (Airplane Disaster Film)
  - Sabotage: detergent in hydraulic fluid --> Can't steer or change altitude
  - Shimmering pressure gauge, visual, not auditory
    —> Delayed discovery of problem
  - Identical sabotage to backup system —
    > Inadequate security, robustness
  - Electrical bypass procedure, but cable brushes a connection, causing short and explosion —
    > Poor juxtaposition of controls, insufficient system simulation
  - Happy ending: substitution of gallery water for hydraulic fluid —
    > Some robustness
- Pilot error on airplanes (New York Time Magazine, 27/3/88)
  - 65% of all jet transport accidents between 1959 and 1969 attributed to errors by flight crews
  - Automatic pilots may exacerbate the problem

Everyday thing interface problems

- Navigation through Sandford Fleming Bldg.
  - Conceptual model underlying numbering scheme
  - Need for maps and other navigational aid
  - Where am I,
    where have I beer
    where am I going?

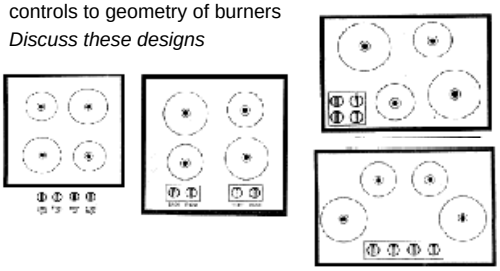


Everyday thing interface problems

- Doors
  - Do you pull or do you push?
  - Need for *affordances* for choosing pull or push and doing so at the proper place
  - Constraints to prevent errors
- Faucets
  - Usually controls over hot and cold water flow
  - But the need is to vary overall flow and temperature
  - Need for *mappings* from given control(s) to these variables
  - The need for *visibility*
  - The need for rapid *feedback*

Stove interface problems

- Four burners, four controls (Norman)
- The mapping problem: Relationship of geometry of controls to geometry of burners
- *Discuss these designs*



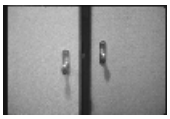
The stateroom of a ship (slides)

- Marvelous design under severe space limitations

Stateroom (slides 2nd set)

Stateroom (slides 3rd set)





Stateroom (slides 4th set)



Stateroom (slides 5th set)


Key concepts (see Norman)

- Affordances
- Constraints
- Mappings
- Conceptual models
- Visibility
- Feedback

- *In each case, students please volunteer examples!!!*

Affordance

- "The perceived and actual properties of a thing, primarily those fundamental properties that determine just how the thing could possibly be used"
- Properties need to be visible
- Method of usage needs to be "natural"
- E.g., buttons are for pushing, menus are for choosing

- *Another example?*

Constraints

- Physical, semantic, cultural, and logical factors that limit the set of all possible actions
- Constraint encourage proper actions, prevent errors
- E.g., training wheels interface

- *Another example?*

Mappings

- For example, the relationships between controls and their effects on a system
- E.g., stove problem is a mapping problem
- E.g., if input goes “up,” output should go “up”, as in control for side view mirrors of car
- *Another example?*

Conceptual models

- Mental models of a system which allow a user to understand the system, to predict the effects of their actions, and to interpret the results
- E.g., tree structure of menus, state diagram of commands
- *Another example?*

Visibility

- Including “the conceptual model of the system, the alternative actions, and the results of actions”
- E.g., lack of visibility in digital phone system
- E.g., lack of visibility, and partial solutions, in VCR and television controls
- *Another example?*

Feedback

- “Sending back to the user information about what action has actually been done and what result has been accomplished”
- E.g., control panel of digital thermostat
- E.g., some feedback with VCR programming
- *Another example?*

Computers (embedded) as everyday things

- Examples
  - Thermostats
  - VCRs
  - Microwaves
  - Watches
  - Notebook computers
  - Cruise controls
  - Home multi-media entertainment consoles
  - etc. etc. etc.
- Ubiquitous computers, invisible computers!!! —> Will be in canes, hearing aids, memory prostheses, pill dispensers, etc.

The bottom line

- Look, observe, and think!!!
- Observe interfaces in *all* everyday things
  - Cars
  - Administrative procedures
  - Instruction manuals
  - etc. etc. etc.

Examples from things for seniors



More examples: things for seniors



From www.independentneeds.com

From www.independentneeds.com

Closing words (Norman, pp. 216-217)

- "Now you are on your own. If you are a designer, help fight the battle for usability. If you are a user, then join your voice with those who cry for usable products. Write to manufacturers. Boycott unusable designs. Support good designs by purchasing them, even if it means going out of your way, even if it means spending a bit more. And voice your concerns to the stores that carry the products; manufacturers listen to their customers.

  When you visit museums of science and technology, ask questions if you have trouble understanding. Provide feedback about the exhibits and whether they work well or poorly. Encourage museums to move toward better usability and understandability.

  And enjoy yourself. Walk around the world examining the details of design. Take pride in the little things that help; think kindly of the person who so thoughtfully put them in. Realize that even details matter, that the designers may have had to fight to include something helpful. Give mental prizes to those who practice good design: send flowers. Jeer those who don't: send weeds."

But, a caution ... a tale of two cookbooks

- The Blender Cookbook
  - Affordances for holding the book up and open
  - Affordances for keeping the pages clean
  - Affordances for adjusting the contents
  - Great human factors, but terrible recipes!
- Interfaces are vital, but functionality is even more vital

Summary

- Design is everywhere!
- Look, observe, think, and learn!!!

Questions and Discussion