CS160: User Interface Design

Video, Widgets, Events

02/27/12

Apple Knowledge Navigator, 1987

http://video.google.com/videoplay?docid=100196171226719096#
Due Today: IPA 3

Control your media browser using the Kinect
(Combine IPA 1 and IPA 2)

Cancelled: IPA 4

Focus on prototyping group project instead!
## Assignment: Low Fidelity Prototype

**Due Mar 7**
Identify project mission statement

Create a **low-fidelity paper prototype** that supports 3 tasks
1. easy, 1 moderate, 1 difficult task

Create a video showing your prototype:
How it supports the 3 tasks
Context in which it will be used (back story)
Your video must include narration!

## Prototyping Theory
Prototype: ?

Prototypes: the means by which designers organically and evolutionarily learn, discover, generate, and refine designs. (Lim & Stolterman)
Prototype:
a representation of a design, made before the final solution exists.
(Moggridge, Designing Interactions)

Prototyping:
producing early working versions of the future application system and experimenting with them.
(Lichter)
Industrial Design Process, ca. 1940 (Dreyfuss)

1. We start by making the component. We analyze models and a variety of other component mechanisms, both here and from abroad.

2. We hand-render sketches with the client's manufacturing intuition. We like to build the limitations as well as the potential of the plane.

3. We learn how the product will be used. In developing Model 600, one designer used a rugged swing mount, rugged top-locating struts and all.

4. After consultation with top managers, other engineers, salesmen, we develop a variety of ideas.

5. Next, we work to closest the design in three dimensions. We start when the work makes a rough clay model.

6. Using the techniques we engaged in design engineering. We start by a master and designer will sell our machine.

7. Though each step there is close collaboration with the client's engineer. Writing proposals are made and checked against their part count.

8. A prototype model - reviewed by the production line manager, checked, and completes the job. Our designer is now into action.
The Value of Prototyping

Benefits of Prototyping

1. We know more than we can tell
2. Actions in the world outperform mental operations
3. The value of surprise
Tacit Knowledge
The Purpose of Prototyping

What questions do prototypes answer?
When and how should they be constructed?
PURPOSE

Understand Existing Experience

“Inquiring Actions”

Communicate

Explore

Experiment

Validate

Anchor Discussion

Persuade
PURPOSE

Understand Existing Experience

“Inquiring Actions”

Communicate

Explore

Experiment

Validate

Anchor Discussion

Persuade

Three Stages of Prototyping (IDEO)

# of ideas

project time

INSPIRE

EVOLVE

VALIDATE

Prototype-driven specifications

Specification-driven prototypes
Prototypes for the Microsoft mouse
From Moggridge,
Designing Interactions, Ch2
PURPOSE

Understand Existing Experience

“Inquiring Actions”

Explore Experiment Validate

Anchor Discussion Persuade

Communicate

PURPOSE

Understand Existing Experience

“Inquiring Actions”

Explore Experiment Validate

Anchor Discussion Persuade

Communicate

PURPOSE

Understand Existing Experience

“Inquiring Actions”

Explore Experiment Validate

Anchor Discussion Persuade

Communicate

PURPOSE

Understand Existing Experience

“Inquiring Actions”

Explore Experiment Validate

Anchor Discussion Persuade

Communicate
Houde and Hill – What do Prototypes Prototype?

Figure 1. A model of what prototypes prototype.

Example 1. Role prototype for 3D space-planning application [E1 Houde 1990].
Example 2. Look-and-feel prototype for 3D space-planning application [E2 Houde 1990].

Example 3. Implementation prototype for 3D space-planning application [E3 Chen 1990].
FUNCTIONALITY

Scope

Partial
Full

Method of Realization

Low-Fidelity Mockup

Horizontal slice (Breadth)

Vertical slice (Depth)

Working Implementation

Simulation / Wizard of Oz

Created With Production Tools

Created With Different Tools

Paper Prototyping
Materials

Large, heavy, white paper (11 x 17)
5x8 in. index cards
Post-it notes
Tape, stick glue, correction tape
Pens & markers (colors & sizes)
Transparencies (including colored)
Colorforms (toy stores)
Scissors, X-acto knives, etc.
Interface Elements

Constructing the Prototype

Set a deadline
Don’t think too long - build it!

Draw a window frame on large paper
**Draw at a large size, but use correct aspect ratio**

Put different screen regions on cards
Anything that moves, changes, appears/disappears
Use greeking to indicate text if necessary

Ready response for any user action
e.g., Have those pull-down menus already made

Use photocopier to make many versions
Wizard of Oz Testing

SIMS 213 Project: Telebear's redesign

User
Observer (or video camera)
“Computer”
Interface
Interface elements

A bit slow for a computer - but it works!
Conducting a Test

Three or Four testers (preferable)

Greeter - Puts users at ease & gets data
Facilitator - only team member who speaks
  Gives instructions & encourages thoughts, opinions
Computer - knows application logic & controls it
  Always simulates the response, w/o explanation
Observer(s) - Take notes & recommendations

Typical session should be approximately 1 hour
Preparation, the test, debriefing

Conducting a Test (cont.)

Greet
  Get forms filled, assure confidentiality, etc.

Test
  Facilitator explains how test will work
  Performs a simple task
  Facilitator hands written tasks to the user
  Must be clear & detailed
  Facilitator keeps getting “output” from participant
  “What are you thinking right now?” “Think aloud”
  Observers record what happens
  Avoid strong reactions, frowning, laughing, impatience – biases the test

Designers should not lead participants
  Let users figure things out themselves as much as possible
  Only answer questions if user remains stuck for a long time
Conducting a Test (cont.)

Debrief
Fill out post-evaluation questionnaire
Ask questions about parts you saw problems on
Gather impressions
Give thanks

Preparing for a Test

Select your participants
Understand background of intended users
Use a questionnaire to get the people you need
Don’t use friends or family

Prepare scenarios that are
Typical of the product during actual use
Make prototype support these (small, yet broad)

Practice running the computer to avoid “bugs”
You need every menu and dialog for the tasks
All widgets the user might press
Remember “help” and “cancel” buttons

WOZ is different from pre-built/canned functionality
Wizard of Oz Tips

Rehearse your actions
Make a flowchart which is hidden from the user
Make list of legal words for a speech interface
Stay “in role”
You are a computer, and have no common sense, or ability to understand spoken English.

Facilitator can remind user of the rules/think-aloud approach if the user gets stuck

Record Critical Incidents

Critical incidents are any unusual/interesting events
Most of them are usability problems.
They may also be moments when the user
   Got stuck
   Suddenly understood something
   Said “that’s cool” etc.
Using the Results

Update task analysis and rethink design
Rate severity & ease of fixing problems
Fix both severe problems & make the easy fixes

Will thinking aloud give the right answers?
Not always
If you ask a question, people will always give an answer, even if it is has nothing to do with the facts
Try to avoid leading questions

Prototyping in Software
Fidelity in Prototyping

Fidelity refers to the level of detail

High fidelity.
Prototypes look like the final product

Low fidelity.
Artists renditions with many details missing
Paper Prototypes are low-fidelity.
What about software?

Hi-Fi Disadvantages

Distort perceptions of the tester
Formal representation indicates “finished” nature
People comment on color, fonts, and alignment

Discourages major changes
Testers don’t want to change a “finished” design
Sunk-cost reasoning: Designers don’t want to lose effort put into creating hi-fi design
Low-fidelity prototyping in software

“Informal” design tools:
Goal is to be as rapid and flexible as physical tools
Add benefits of digital media: Undo, copy+paste, resizing, etc.
May be a good idea to design in software, then print out elements and screens for paper testing.

Examples:
DENIM (UC Berkeley)
Balsamiq Mockups
High-fidelity visual mockups

- Interface looks like the final application
- May or may not be interactive
- Often, interactivity is limited to static changes

Example Tools:
- Keynote + Powerpoint
- Adobe Flash Catalyst

“Slideware” Prototypes

- Uses standard presentation software to create screen designs
- Libraries of widgets for many platforms (e.g., iPhone, Android, Web, Facebook apps) exist
- Each slide shows one key screen
Telling vs. Testing for Touch Screens

To tell others about your design (in a presentation), visually indicate the action a user is taking.
For touch screens: include hands!
Linear: you (designer) are in control of the narrative.

Telling vs. Testing for Touch Screens

To test your design, link different screens together and let users explore
Non-linear: user is in control of the narrative
High-fidelity, fully-interactive prototypes

Look and behave like the final application.
Take a lot of effort to build – too little payoff?
Example tools:
• HTML+CSS+Javascript
• Apple DashCode
• jqTouch library
What tools do designers in industry use to prototype?

Professional Use of Prototyping Tools Reported by Myers

- Adobe Photoshop: 83%
- Adobe Dreamweaver: 66%
- Microsoft PowerPoint: 54%
- Adobe Illustrator: 52%
- Adobe Flash: 43%
- Microsoft Visio: 37%
- Adobe InDesign: 25%
- Omni Group Omnigraffle: 23%
- Microsoft Visual Studio: 19%
- Adobe Fireworks: 17%
- Adobe Director: 13%
- Microsoft Frontpage: 12%
- Adobe AfterEffects: 10%
- Axure RP: 8%
- Adobe GoLive: 6%
- Adobe Flex: 6%
- Microsoft Expression Blend: 4%

From: Myers
Video Prototyping

Video Prototypes

Narrative: You control the story!
Use existing software & images of real settings
Narration optional (but required for your assignment!)
  Explain events while others move images/illustrate interaction

With good storyboards, should be able to create video prototype in a few hours
Creating a Video Prototype

1) Review field data about users & work practices
2) Create use scenario in words
3) Develop storyboard of each action/event with annotations explaining the scene. Put each element on a card. *This will save you a lot of time later.*
4) Shoot a video clip for each storyboard card
   - Either Live action or UI Screen recording.
   - Hold last frame of a section/shot for 1s
5) Use titles to separate clips (keep it onscreen for 3s)
CLUSTER
Video Prototype

Andy Hou, Kevin Chiu
Tips & Tricks

Add structure to better explain context
Begin with a title
Follow with an “establishing shot”
Switch between showing UI and showing live shots
At the end, connect back to the original motivation

Editing: Keep it simple!
Live video is most convincing to show context,
but still photos + narration also work
Don’t obsess about transitions and composition – just tell your story.

Higher Fidelity Videos

![Higher Fidelity Videos Image]
Advantages of Low-Fi Prototyping

**Takes only a few hours**
No expensive equipment needed

**Can test multiple alternatives**
Fast iterations
Number of iterations is tied to final quality

**Can change the design as you test**
If users are trying to use the interface in a way you didn’t design it – go with what they think! Adapt!

**Especially useful for hard to implement features**
Speech and handwriting recognition
Drawbacks of Lo-Fi Prototyping

Evolving the prototype requires redrawing
Can be slow (but reprogramming usually slower)

Lack support for “design memory”

Force manual translation to electronic format

Do not allow real-time end-user interaction

Summary

Informal prototypes allow you to design (and test!) before writing code.

Rapid evolution and elimination of many problems happens in this phase.

Use informal paper + software prototypes for exploration (what should the design be?).

Once you have a compelling design, create a video for communication and persuasion.
Interactive Application Programming

In the beginning...

http://www.cryptonomicon.com/beginning.html
The Xerox Alto (1973)

Event-Driven UIs

Old model (e.g., UNIX shell, DOS)
Interaction controlled by system, user queried for input when needed by system

Event-Driven Interfaces (e.g., GUIs)
Interaction controlled by user
System waits for user actions and then reacts
More complicated programming and architecture
Console program pseudo-code

Do some work…
Prompt user for input
Wait for user input
Process user input…
Do some more work…
Exit

Console program

// Java Example:
Console console = System.console();
String name = console.readLine("Your name:");
System.out.println("You have entered : " + name);
String pass= console.readLine("Password:");
System.out.println("…");
Minimal “interactive” program

Do until a quit command: {
    wait for user input
    process it…
    (optionally) update display
}

Minimal “interactive” program

Do until a quit command: {
    wait for user input
    \textbf{switch (input-cmd)} {
        \textbf{case insert:} do-insert(…)
        \textbf{case delete:} do-delete(…)
        \textbf{case backspace:} …
    }
    (optionally) update display
}
Minimal “interactive” program

Can’t use this (global) approach for window systems, because the result of a user command depends on the active window (and the active component within that window).

Too many possible combinations of input x target window, and window structure is dynamic.

GUI Toolkits

Most user interfaces today are written using toolkits (e.g., QT, Cocoa, Java Swing, GTK, Android SDK, …) Toolkits come with libraries of interactive elements (widgets) and layouts

Frequently used interactive components

Toolkits also define an architecture:

A standard way to handle input and output

Usually wrap main() – application programmer writes pieces of code that plug into the architecture

The architecture specifies how to write new widgets for the library