User Centered Design and Prototyping

Why User Centered Design is important
How Prototyping helps User Centered Design

System Centered Design

—we could design the product with a simple point-and-click interface...
or we could require the user to choose among thousands of poorly documented commands, each of which must be typed exactly right on the first try.

bear in mind, i will never meet a customer ourselves.
make it so they have to reboot after every typo.
**System Centered Design**

What can be built easily on this platform?
What can I create from the available tools?
What do I as a programmer find interesting to work on?

Ideally we AVOID doing this!

---

**User Centered System Design**

Design is based upon a user’s
• abilities and real needs
• context
• work
• tasks

Golden rule of interface design:
“Know The User”
**User Diversity**

It is important to note that the users that you think will benefit might only be a subset of the actual set of users that will benefit from or make use of your technology.

Some assistive technologies that were “meant” for users with hearing or vision impairments are used by “everyday” users today.

---

**User Centered System Design**

... is based on understanding the domain of work or play in which people are engaged and in which they interact with computers, and programming computers to facilitate human action...

Three assumptions

- The result of a good design is a satisfied customer.
- The process of design is a collaboration between designers and customers. The design evolves and adapts to their changing concerns, and the process produces a specification as an important byproduct.
- The customer and designer are in constant communication during the entire process.

---

Participatory Design

Problem
• intuitions wrong
• interviews etc. not precise
• designer cannot know the user sufficiently well to answer all issues that come up during the design

Solution
Designers should have access to pool of representative users. These are the “end-users” themselves, not their managers or union reps, etc.

Participatory Design

Users become first class members in the design process
– active collaborators vs passive participants

Users considered subject matter experts
– know all about the work context

Iterative process
– all design stages subject to revision
**Participatory Design**

**Participatory Design:**

**Up side**
- users are excellent at reacting to suggested system designs
  - designs must be concrete and visible
- users bring in important “folk” knowledge of work context
  - knowledge may be otherwise inaccessible to design team
- greater buy-in for the system often results

**Down side**
- hard to get a good pool of end users
  - expensive, reluctance ...
- users are not expert designers
  - don’t expect them to come up with design ideas from scratch
- the user is not always right
  - don’t expect them to know what they want

---

**Methods for involving the user**

**User**
- Uses system after deployment.

**Tester**
- Tests system after development, before deployment

**Informant**
- Helps during development – perhaps by critiquing designs, participating in interviews, observations of current practices, etc.

**Design Partner (full PD)**
- Equal partner

- Allison Druin, UMD (CHI 2000)
Methods for involving the user

At the very least, talk to users
• surprising how many designers don’t!

Interviews
• used to discover user’s culture, requirements, expectations, etc.
• contextual inquiry:
  – interview users in their workplace, as they are doing their job

Explain designs
• describe what you’re going to do
• get input at all design stages
  – all designs subject to revision
• important to have visuals and/or demos
  – people react far differently with verbal explanations

Prototyping

<table>
<thead>
<tr>
<th>Early design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brainstorm different representations</td>
</tr>
<tr>
<td>Choose a representation</td>
</tr>
<tr>
<td>Rough out interface style</td>
</tr>
<tr>
<td>Task centered walkthrough and redesign</td>
</tr>
<tr>
<td>Fine tune interface, screen design</td>
</tr>
<tr>
<td>Heuristic evaluation and redesign</td>
</tr>
<tr>
<td>Usability testing and redesign</td>
</tr>
<tr>
<td>Limited field testing</td>
</tr>
<tr>
<td>Alpha/Beta tests</td>
</tr>
<tr>
<td>Low fidelity paper prototypes</td>
</tr>
<tr>
<td>Medium fidelity prototypes</td>
</tr>
<tr>
<td>High fidelity prototypes / restricted systems</td>
</tr>
<tr>
<td>Working systems</td>
</tr>
</tbody>
</table>

| Late design |
Low fidelity prototypes

Paper-based prototypes
- a paper mock-up of the interface look, feel, functionality
- “quick and cheap” to prepare and modify

Purpose
- brainstorm competing representations
- elicit user reactions
- elicit user modifications / suggestions

Sketches
- drawing of the outward appearance of the intended system
- crudity means people concentrate on high level concepts
- but hard to envision a dialog’s progression

Generally not good!
Should really be hand-drawn on paper.
Low fidelity prototypes

Iterate

• “To get a good idea, get lots of ideas”

The speed of lo-fi prototypes makes it fundamentally easier to go through several iterations – each with feedback from users.

Low fidelity prototypes

Storyboarding

• a series of key frames
  – originally from film; used to get the idea of a scene
  – snapshots of the interface at particular points in the interaction

• users can evaluate quickly the direction the interface is heading
**Storyboard of a computer based telephone**

**PICTIVE prototypes**

- “Plastic Interface for Collaborative Technology Initiatives through Video Exploration” - Muller, CHI 91

- Design is multiple layers of sticky notes and plastic overlays
  - different sized stickies represent icons, menus, windows etc.

- Interaction demonstrated by manipulating notes
  - contents changed quickly by user/designer with pen and note repositioning

- Session can be videotaped for later analysis
  - usually end up with mess of paper and plastic!
PICTIVE prototypes

Can create pre-made interface components on paper (though this can lock users into a certain initial mindset).

e.g., these empty widgets were created in visual basic and can be printed out:

```
buttons
combo box
list box

menu

alert box

entries
tabs
```

*I would argue it is still better to hand-draw them…*

Other uses of low fidelity prototypes

Tutorials and manuals

- write them in advance of the system
- what are they?
  - tutorial for step by step description of an interaction
    - an interface “walk-through” with directions
  - manual for reference of key concepts
    - in-depth technical description
- if highly visual, then storyboard is set within textual explanations
- does this work?
  - people often read manuals of competing products to check:
    - interface
    - functionality
    - match to task
A directory title shows you the name of the folder you're presently working in—in this case, the TeachText Folder. The box beneath it shows you all the other items in the TeachText Folder that you can open with this application—in this case, only the Memos Folder.

1908s - From Apple’s Tutorial Guide to the Macintosh Finder

Photoshop 7 – Toolbox
Overview from help
1980s - From Apple’s Tutorial Guide to the Macintosh Finder

As you open the Memos Folder, you move down through the hierarchy. The directory title changes to remind you where you are in the hierarchy, and the box shows you what's on the new level you just moved to—in this case, the two documents in the Memos Folder. The selected document is the one that will open when you click the Open button. If you want to open the other document, click anywhere on the other document's name to highlight it, and then click the Open button.

From the 2007 TurningPoint User Guide

The following descriptions define the buttons that make up the tools in the TurningPoint toolbar. The numbered labels in the previous image show you the location of each tool.

1 Activate/Update

Allows you to submit the license information for TurningPoint software and upgrades. Find information about installation and customer support in Chapter 1: About TurningPoint.
**Medium fidelity prototypes**

**Prototyping with a computer**

• simulate or animate some but not all features of the intended system
  – engaging for end users

**Purpose**

• provides a sophisticated but limited scenario to the user to try out
• provides a development path (from crude screens to functional system)
• can test more subtle design issues

---

**Medium fidelity prototypes**

**Danger**

• Medium fidelity prototypes take too long to build and change
  – Reduces number of iterations
• User’s reactions are usually “in the small”
  – blinds people to major representational flaws
• Developers resist changes
  – “but it is already working…”
• A single bug can halt testing
• Management may think its real!
**Medium fidelity prototypes**

**Approaches to limiting prototype functionality**

- **vertical prototypes**
  - includes in-depth functionality for only a few selected features
  - common design ideas can be tested in depth

- **horizontal prototypes**
  - surface layers includes the entire user interface with no underlying functionality
  - a simulation; no real work can be performed

- **scenario**
  - scripts of particular fixed uses of the system; no deviation allowed

---

**Medium fidelity prototypes**

**Wizard of Oz** - A method of testing a system, or a part of a system, that does not yet exist.

- human simulates the system’s intelligence and interacts with user

- uses real or mock interface
  - “Pay no attention to the man behind the curtain!”

- user uses computer as expected

- “wizard” (preferably hidden):
  - interprets subjects input according to an algorithm
  - has computer/screen behave in appropriate manner
  - might have errors artificially introduced

- good for:
  - adding simulated and complex vertical functionality
  - testing futuristic ideas
**Wizard of Oz Examples (I)**

**IBM: an imperfect listening typewriter using continuous speech recognition**
- secretary trained to:
  - understand key words as “commands”
  - to type responses on screen as the system would
  - manipulating graphic images through gesture and speech

**Intelligent Agents / Programming by demonstration**
- person trained to mimic “learning agent”
  - user provides examples of task they are trying to do
  - computer learns from them
- shows how people specify their tasks

**In both cases, system very hard to implement, even harder to change!**

---

**Wizard of Oz Examples (II)**

**Imagining scenarios where you aren’t sure whether the investment is worth the ‘payout’ or you want to develop the technology while exploring interface ideas.**

- You want to build a map system that shows where the user is in real-time. Rather than needing to install tracking systems before being able to do the UI testing, you could have a wizard watching the users and updating their location manually on the system.

- You want to have location-aware directional cues such as blinking lights or arrows or sound effects turn on and off as appropriate to guide a user to a destination. Again, you could have a wizard instruct the system to turn things on and off without having the proximity sensors installed or heuristics to determine the user’s directional orientation.
**What you now know**

**User centered design**  
- design is based upon a user’s real needs, tasks, and work context

**Participatory design**  
- brings *end-user* in as a first class citizen into the design process

**Prototyping**  
- allows users to react to the design and suggest changes  
- low-fidelity prototypes best for brainstorming and choosing representations  
- medium-fidelity prototypes best for fine-tuning the design

**Prototyping methods**  
- vertical, horizontal and scenario prototyping  
- storyboarding  
- Pictive  
- scripted simulations  
- Wizard of Oz

**Reading**  