

Computer/Human Interaction

Lecture 29

Overview: Prototyping

- Exploring User Requirements
- Choosing Among Alternatives
- Usability Testing
- Evolutionary Development

Fridays rest of November will be project work days.

Scenario-Based Design

- Analysis – Problem Scenarios
- Design – Activity Scenarios -> Information Scenarios -> Interaction Scenarios
- Prototype – this week's topic
- Evaluate – next week's topic

What is a Prototype?

- Any concrete, but partial implementation of a system design created to explore system issues
- In particular, a UI prototype is used to explore usability issues
- Four goals of prototyping
 - Exploring User Requirements
 - Choosing Among Alternatives
 - Usability Testing
 - Evolutionary Development

Explore User Requirements

- Illustrate current or future use
- Used in participatory design with stakeholders; record questions, reactions, and ideas for changes
- Low fidelity prototypes – storyboards, paper mockups
- High fidelity prototypes – Wizard of Oz, video, computer animation, scenario machine, rapid prototype, partial implementations

Exploring User Requirements 2

- Issues in choosing how to prototype include
 - Goals and resources of project team
 - Expertise of project team with prototype tools
 - Expectations of audience
 - Presentation context
- Want to be careful that the prototype doesn't define the final system...

Choosing Among Alternatives

- Prototypes can be built to answer specific questions regarding system function. E.g.,
 - Direct manipulation vs. command language
 - Frequency and amount of feedback
- Expensive to do high-fidelity prototypes, choose what to prototype based on the claims analysis
- Good for exploring risky or critical features that need a go/no-go decision

Usability Testing

- Usability testing is core of usability engineering practice.
- Try out ideas with target users as early as possible
- Ideally an early working version makes the best prototype, but may delay usability testing
- Often use rapid prototyping tools to build temporary, discardable prototypes; can also use low-fidelity prototypes for some parts

Evolutionary Development

- SBD UI development works very well with agile software engineering techniques like extreme programming (XP)
- Agile techniques are a reaction to overly formal, rigid software engineering processes
 - No one method fits all projects
 - Identify the "lightest" method possible
 - Particularly for web-based business applications

Extreme Programming (XP)

- Developed by Kent Beck in late 1990's
- Goal is to do extremely rapid development while avoiding defects
- Includes lots of different practices including pair programming, continuous system integration, refactoring, test first
- Key issues have been scaling the technique to very large projects, security of information, etc

SBD and XP

- Relevant XP practices related to SBD include metaphors, clients on-site and part of design team, user stories
- SBD integrates well with XP. UI design proceeds in parallel with software design
- Key is that both are inquiry-based design processes. Users participate as full members of design team. Feedback can change design at any time.

Key Tradeoffs

- Quality of prototype vs. premature commitment
- Building prototypes vs. time & resource management
- Realistic prototypes vs. early availability or discardable efforts
- Constant iteration vs. radical changes and/or refactoring of a design
- Dynamic platforms vs. organized, well-structured code base

Homework 4

- Exercises 1 & 2 on page 224 of textbook.
- **Due at beginning of class on Wednesday.** Will compare and discuss as part of class.