CMPS 115 Winter 04

Class #6 (2004/01/22)

- Changes/Review
- Intro to UI design
- Some bloopers
- Paper prototyping
- User Manual Expectations

<break>
- EXAM 1 Handout; Q&A in-class
Changes & Lecture 5 Takeaway

- Changes/Notices
- Guest:
  - SW dev is embedded in business process
  - Investment curves affect development process
- Requirements:
  - Three key techniques for finding ambiguities (emphasis, dictionary mix-n-match, memorization)
  - Verify (cross-check) and Validate (analysis, inspect, demo, test)
- Use Cases
  - Three forms (narrative, scenario, conversational)
  - Format not standardized, templates available, varies with canned process
UI Design Principles

*GUI Bloopers*, Morgan-Kaufmann, Jeff Johnson

- Focus on users/tasks - not technology
- Function first, presentation later
- Conform to user’s view of task
- Don’t complicate user’s task
- Promote learning
- Deliver information, not just data
- Design for responsiveness
- Try it out on users, then fix it
“Blooper” categories

- GUI component - using component in surprising ways
- Layout & appearance - complicated; confusing groups; bad fonts
- Textual - bad writing, bad titles (same on 2 windows, etc)
- Interaction (more later)
- Web-specific (“back” broken; frames; hidden links, etc)
- Responsiveness (more later)
- Management (treat UI as low priority, discounting testing, no domain expert, giving programmer fastest computer)
Interaction Bloopers

- Allowing implementation to dictate GUI
  
  SCREEN BELONGS TO USER
  
  - example: reusing a Form without restoring state
  - example: asking user for random number seed

- Presenting information poorly
  
  - overwhelming detail, too many decisions
  - unexpected rearrangement
  - instructions that go away too soon

- Failure to consider lifecycle (install, de-install)
Responsiveness Bloopers

- Cursor/keystrokes not keeping up with user
- Buttons not acknowledge a click, or long delay
- No busy indicator/feedback
- Long-duration continuous operations without feedback (such as rubberband or card-drag)
- Periodic/unpredictable UI freeze (Java risk)
- Ignoring user input (including Abort or Cancel) until finishing lengthy task
Responsiveness Parameters

- 0.1 sec user action -> user-visible change
  - button-push, pointer movement
  - animation frame intervals

- 1 second max comfortable gap in “conversation”
  - showing progress bar change
  - autosaving pause
  - “New game” to game ready

- 10 seconds task duration/planning “unit task”
  - complete user input (entering name in high-score table)
Responsiveness Design Principles

- Responsiveness != Performance
- Resource are *always* limited
- UI is a real-time interface
- Not all delays are equal; the s/w needn’t do everything immediately
- S/W need not do tasks in the order requested
- S/W need not do everything it’s asked to do
- Human != computer
Client Responsiveness Design

- Local/Remote awareness
  - Server may balk; client should keep running smoothly
  - Optimistic response (correct if server updates otherwise)

- Threading
  - User thread services UI (collects events, changes control state, posts events) … not to be used for game action!
  - Game thread runs engine, calculates local world based on local events and server data
  - I/O thread handles network (critical robustness point)
Paper Prototyping

Lo-fidelity values compared to hi-fi

– faster
– induce more experimentation
– avoid raising “almost done” expectations
– better comments, more to interaction less to “finish”
– robust (people recover quickly during UI test)
– lots more fun :)}
User Manual Expectations

- Formal doc - use template
- Traceability matrix
  - Trace to satisfying application element(s)
  - If multiple, show decomposition
  - Good practice is to indicate how validated, but this isn’t required (due to time constraints)
EXAM #1

- Available in Word97, PDF, OpenOffice SXW
- Q&A must all be public
  - DForge site (Lecture-centered Discussion Forum)
  - Don’t ask friend “what do you think this means” ...