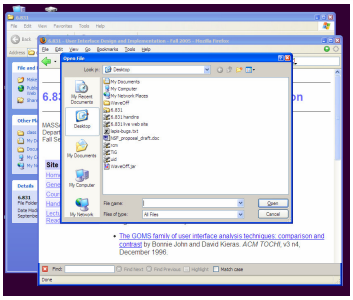
Lecture 18: Predictive Evaluation

UI Hall of Fame or Shame?



Suggested by Vikki Chou

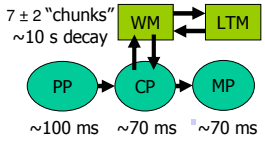
UI Hall of Fame or Shame?

Today's Topics

- Keystroke-level models
- GOMS
- CPM-GOMS

Predictive Evaluation

- Predictive evaluation uses an engineering model of human cognition to predict usability
- Model is
  - abstract
  - quantitative
  - approximate
  - estimated from user experiments



Advantages of Predictive Evaluation

- Don't have to build UI prototype
  - Can compare design alternatives with no implementation whatsoever
- Don't have to test real live users
- Theory provides explanations of UI problems
  - So it points to the areas where design can be improved
  - User testing may only reveal problems, not explain them

Keystroke-Level Model (KLM)

- **K**eystroke or mouse button press
- **P**oint with mouse
- **D**raw straight line with mouse
- **H**ome hands between mouse and keyboard
- **M**entally prepare

KLM Analysis

- Encode a method as a sequence of physical operators (KPHD)
- Use heuristic rules to insert mental operators (M)
- Add up times for each operator to get total time for method

Estimated Operator Times

- **K**eystroke determined by typing speed
  - 0.28 s average typist (40 wpm)
  - 0.08 s best typist (155 wpm)
  - 1.20 s worst typist
- **P**ointing determined by Fitts's Law
  - T = a + b log(d/s + 1) = a + b ID
  - 0.8 + 0.1 ID [Card 1978]
  - 0.1 + 0.4 ID [Epps 1986]
  - -0.1 + 0.2 ID [MacKenzie 1990, mouse selection]
  - 0.14 + 0.25 ID [MacKenzie 1990, mouse dragging]
  - OR
  - T ~ 1.1 s for all pointing tasks
- **D**rawing determined by steering law

Estimated Operator Times

- **H**oming estimated by measurement
  - 0.36 s (between keyboard and mouse)
- **M**ental preparation estimated by measurement
  - 1.35 s [1.08 − 1.35]

Heuristic Rules for adding M's

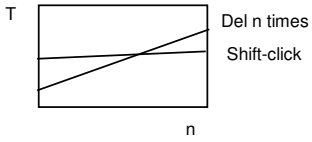
- Basic idea: M before every chunk in the method that must be recalled from long-term memory
- Insert M's before each K & P
  - K => MK
  - P => MP (if P points at a command, not an argument)
- Delete M's in typed chunks
  - MK MK ... MK => M KK .. K if K's form a command name, single text string, or number
- Delete anticipated M's
  - x M y => x y if x fully anticipates y
  - e.g., point-and-click is a chunk, so PMK => PK

Example: Deleting a Word

- Shift-click selection
  - M
  - P [start of word]
  - K [click]
  - M
  - P [end of word]
  - K [shift]
  - K [click]
  - H [to keyboard]
  - M
  - K [Del]
- Total: 3M + 2P + 4K
  - = 7.37 sec
- Del key N times
  - M
  - P [start of word]
  - K [click]
  - M
  - K [Del]
    - x n [length of word]
- Total: 2M + P + (n+1)K
  - = 4.08 + 0.28n sec

Applications of KLM

- Comparing designs & methods
- Parametric analysis



Limitations of KLM

- Only expert users doing routine (well-learned) tasks
- Only measures efficiency
  - Not learnability, memorability, errors, etc.
- Ignores
  - errors (methods must be error-free)
  - parallel action (shift-click)
  - mental workload (e.g. attention & WM limits)
  - planning & problem solving (how does user select the method?)
  - fatigue

GOMS

- Goals
- Operators
- Methods
- Selection rules

- GOMS offers a language for task analysis and high-level design description
  - can be abstract or detailed
  - can be qualitative or quantitative

Example

- Goal: delete text (n chars long)
  - Select: method 1 if n > 10
    method 2 if n < 10
  - Method 1: Goal: highlight text & delete
    - Goal: highlight text
      - Point
      - Click
      - Point
      - Shift
      - Click
  - Method 2: Goal: delete n chars
    ...

NGOMSL

- "Natural GOMS language"
  - formal language with restricted English syntax
- Addresses gaps in KLM modeling
  - learning time measured by the # of NGOMSL statements
  - working memory use modeled by Retain and Recall statements
  - no errors or problem solving

NGOMSL

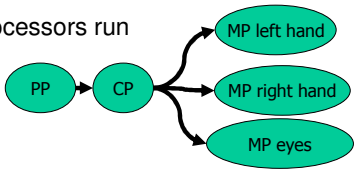
```
Method for goal: Move text
  Step 1.  Accomplish goal: Cut text.
  Step 2.  Accomplish goal: Paste text.
  Step 3.  Return with goal accomplished.

Method for goal: Cut text
  Step 1.  Accomplish goal: Highlight text.
  Step 2.  Retain that the command is CUT, and
             accomplish goal: Issue a command.
  Step 3.  Return with goal accomplished.

Method for goal: Paste text
  Step 1.  Accomplish goal: Position cursor at insertion point.
  Step 2.  Retain that the command is PASTE,
             and accomplish goal: Issue a command.
  Step 3.  Return with goal accomplished.
```


CPM-GOMS

- CPM-GOMS models parallel operations
  - e.g. point & shift-click
- Uses parallel cognitive model
  - each processor is serial
  - different processors run in parallel



Critical Path Determines Time