Outline

• Introduction to Evaluation
  - Types of Evaluation etc.
  - Usability Specifications
• Analytic Methods
  - Metrics and Models
  - Heuristics and Inspection
• Try a Collaborative/Heuristic Inspection
• (Next week: Empirical Methods)
Introduction to Evaluation

Have we achieved our goals?
Are they the right goals?

Evaluation in Design Lifecycle

**Formative:** informs design
- *Early*
  - checking understanding of requirements
  - quick filtering of ideas
- *Middle*
  - predicting usability
  - comparing alternate designs
  - engineering towards a usability target

**Summative:** have we done well?
- *Late*
  - fine tuning of usability
  - verifying conformance to a standard
Analytic Evaluation

• Information from expert or theory
• **Usability Inspections**
  - **Heuristic**: apply guidelines
  - **Model-Based**: model simulates human behavior
• **Metric**: measurements towards some objective
• Stronger on interpretations than facts (validity)

Empirical Evaluation

• Information from user
• **Performance**: observing activity, testing
• **Subjective**: user opinions
• Stronger on facts than interpretations

Can we combine the strengths and weaknesses of Analytic and Empirical?
Mediated Evaluation

- Perform analytic evaluation early and often to inform design (formative)
- Potential problems identified analytically inform focus of empirical evaluation
- Analytic evaluation tells us how to interpret empirical results
- Supports both formative and summative evaluation

Things we might measure

- Analytical
  - Predicted number of keystrokes needed (GOMS)
  - Number of command actions that are hidden versus visible
  - Complexity and organization of interface

- Performance
  - Time to complete task
  - Number or percent of task completed per unit time
  - Number of errors per task or unit time
  - Time required to reach task criterion or error rate
  - Rate of use of help system
  - Quality of task product

- Subjective (“Psychometric”)
  - User’s attitude towards the system
  - Perception of efficiency
  - Perception of helpfulness
  - Perception of control
  - Perception of learnability

Consider also group level measurements
Usability Specifications

Evaluation for usability engineering needs measurable specifications ...

Usability Specifications in SBD
Usability Specifications (from Rosson)

- Quality objectives for final system usability
  - like any specification, must be precise
  - managed in parallel with other design specifications
- In SBD, these come from scenarios & claims
  - scenarios are analyzed as series of critical subtasks
  - reflect issues raised and tracked through claims analysis
  - each subtask has one or more measurable outcomes
  - tested repeatedly in development to assess how well project is doing (summative) as well as to direct design effort toward problem areas (formative)
- Precise specification, but in a context of use

Deriving Usability Specs in SBD

Activity, information, and interaction claims: identify key design features to be tested

Design scenarios: extract motivation and context for subtasks to be tested

Usability specifications: a list of subtasks with performance and satisfaction parameters

Estimates of behavior: published or pilot data of expected user behavior
Example: Scenario (from Rosson)

- When Mr. King meets Sally in the VSF, he can see she is already there, so he selects her name and uses Control+I to see that she is working on her slides, then Control+F to synchronize with her.
- He watches her work, and sees her uploading files from her desktop using a familiar Windows browse-file dialog.
- When he sees an Excel document, he experiments to see if it is 'live', discovers he can edit but not save.
- When he sees that she is planning to have visitors come up with their own results using her simulation, he advises her that this will crowd the display, and goes off to find a way to create a 'nested' display element.

Example: Claims (from Rosson)

- Using "Control-I" to identify activities of co-present user
  + ties info about people directly to their representation on the display
  + simplifies the screen display by hiding activity information cues
  - but conflicts with the real world strategy of just looking around
  - but this special key combination must be learned
- Exhibit components shown as miniaturized windows
  + suggests they may contain interactive content
  - but viewers may interpret them as independent applications
- File-browsing dialogs for uploading workstation documents
  + builds on familiarity with conventional client-server applications
  + emphasizes a view of exhibits as an integration of other work
  - but the status of these personal files within the VSF may be unclear.
Example: Claims to Subtasks

- (Text says use HTA, but this is informal...)
- Identifying and joining co-present users
  - key combinations are harder to learn; how distracting or difficult are they in this case?
- Recognizing and working with components
  - will users understand these as 'active' objects?
  - will they know how to activate them?
  - will they know what is possible when they have done this?
- Importing desktop files into the VSF
  - is the operation intuitive, smooth?
  - is there any resulting confusion about the status of the uploaded files?

Example: Resulting Usability Specs

<table>
<thead>
<tr>
<th>Scenario &amp; Subtasks</th>
<th>Worst Case</th>
<th>Planned</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interaction Scenario: Mr. King coaches Sally</td>
<td>2.5 on usefulness, ease of use, and satisfaction</td>
<td>4 on usefulness, ease of use, and satisfaction</td>
<td>5 on usefulness, ease of use, and satisfaction</td>
</tr>
<tr>
<td>1. Identify Sally's view and synchronize</td>
<td>1 minutes, 1 error 3 on confusion</td>
<td>30 seconds, 0 error 2 on confusion</td>
<td>10 seconds, 0 error 1 on confusion</td>
</tr>
<tr>
<td>2. Upload desktop file from the PC</td>
<td>3 minutes, 2 errors 3 on familiarity</td>
<td>1 minute, 1 error 4 on familiarity</td>
<td>30 seconds, 0 error 5 on familiarity</td>
</tr>
<tr>
<td>3. Open, modify, try to save Excel file</td>
<td>2 minutes, 1 error 3 on confidence</td>
<td>1 minute, 0 errors 4.5 on confidence</td>
<td>30 seconds, 0 error 5 on confidence</td>
</tr>
<tr>
<td>4. Create nested exhibit component</td>
<td>5 minutes, 3 errors 3 on complexity</td>
<td>1 minute, 1 error 2 on complexity</td>
<td>30 seconds, 0 error 1 on complexity</td>
</tr>
</tbody>
</table>

- Precise measures
  - Derived from published & pilot data
  - Time to perform task, Error rates, Likert scale ratings
Generality of SBD Usability Specs

- Salient risk in focusing only on design scenarios
  - may optimize for these usage situations
  - the “successful” quality measures then reflect this
- When possible, add contrasting scenarios
  - overlapping subtasks, but different user situations (user category, background, motivation)
  - assess performance satisfaction across scenarios
- Construct functional prototypes as early as feasible in development cycle (unlike UCD)
- Mediated evaluation may also help identify tasks for which you need specs.

Analytic Evaluation

Metrics and Models
Why Analytic Evaluation

- Performance testing is expensive and time consuming, and requires a prototype (although I encourage you to always do at least some of it)
- Analytic techniques use expertise of human-computer interaction specialists (in person or via heuristics or models they develop) to predict usability problems without testing or (in some cases) prototypes
  - Can be done early in process
  - Can also yield metrics to compare designs or track progress

Metrics and Models

- **Structural:** surface properties
  - Tend not to be correlated with usability
- **Semantic:** content sensitive
  - How users might make sense of relationships between components
- **Procedural:** task sensitive
  - How content and organization fits specific task scenarios or use cases
- **Note:** C&L like these; Nielsen and R&C don’t think they are worthwhile
**Fitts' Law** (Fitts, 1954)

- The time $T$ to point at an object using a device is a function of the distance $D$ from the target object & the object's size $S$:
  \[ T = k \log_2(D/S + 0.5), \quad k \sim 100 \text{ msec} \]
- The further away ($T \sim D$) and the smaller ($T \sim 1/S$) the object, the longer the time to locate it and point.
- What does this say about
  - Pie menus?
  - Objects on the edge of screen or corners?
  [http://www.asktog.com/columns/022DesignedToGiveFitts.html](http://www.asktog.com/columns/022DesignedToGiveFitts.html)

**Keystroke Level Modeling**

- Simulates *expert* behavior
- No users or prototype needed!
- Input:
  - Specification of functionality
  - Task analysis
- Add time for physical & mental acts
  - Keystroking, pointing, homing, drawing
  - Mental operator
  - System response
Keystroke Modeling Example

Suppose we determined these parameters ...

- Keystroking (K): average of 0.35
- Pointing (P): 1.10 (but see Fitt’s Law)
- Clicking on mouse (P1): 0.20
- Homing (H) hands over device: 0.40
- Drawing (D) a line: variable
- Mental operator (M): 1.35 to make a decision
- System response (R): variable

Keystroke Modeling Example

Save a file using mouse and pull down menu

1. Mentally prepare: \( M = 1.35 \)
2. Initial homing (reaching) to mouse: \( H = 0.40 \)
3. Move cursor to file menu: \( P = 1.10 \)
4. Select “save as” in file menu (click, decide, move, click): \( P1 + M + P + P1 = 0.20 + 1.35 + 1.10 + 0.20 = 2.85 \)
5. Application builds dialog and prompts for file name: \( R = 1.2 \)
6. User chooses name and types 8 characters: \( M + K^8 + K \text{ for return} = 1.35 + 0.35*8 \text{ for } 0.35 = 4.5 \)
Total = 12.65
GOMS (Card, Moran & Newell, 1983)

Influential family of models for predicting performance

- **Goals** - the state the user wants to achieve e.g., find a website
- **Operators** - the cognitive processes & physical actions performed to attain those goals, e.g., decide which search engine to use
- **Methods** - the procedures for accomplishing the goals, e.g., drag mouse over field, type in keywords, press the go button
- **Selection rules** - determine which method to select when there is more than one available

Essential Efficiency (C&L)

- How efficient is the design?
- Assumes that essential use cases express the minimum number of user steps for a task
- Enacted steps are “what users experience as discrete actions, such as selecting, moving, entering, deleting”
- Ratio of essential steps (from EUC) to enacted steps (from analysis of what the user actually has to do) gives efficiency of design:
  \[ EE = 100 \left( \frac{S_{\text{essential}}}{S_{\text{enacted}}} \right) \]
- Can compute weighted sum of EE for N tasks
Task Concordance (C&L)

- Are frequently executed tasks easier?
- Rank tasks by expected frequencies (ranking is easier to judge than absolute frequencies)
- Compare this to ranking by difficulty with Kendall’s tau (rank order correlation)
  \[ TC = 100 \frac{D}{P} \]
  \[ D = \# \text{ pairs in order} - \# \text{ out of order} \]
  \[ P = \text{number of possible pairs} \]
- -100 to 100

Task Visibility (C&L)

- Is what you need visible when you need it?
- Determined enacted steps for use case
- Visibility Score \( V_i \) for each enacted step \( i \) is:
  - 0 if recall is needed (Hidden);
  - .5 if it is available but needs to be exposed (Exposing), or
  - if the interaction context is changed (Suspending)
  - 1 if visible (Direct)
- Then find portion of steps that are visible:
  \[ TV = 100 \left( \frac{\text{Sum of } V_i \text{ for } N \text{ steps}}{N} \right) \]
- Thus, if all steps are visible (1) the score is 100
Structural Metrics

- Layout Uniformity
  - How similar are the interface elements (widgets)?
  - Measures height, width, and alignments.
  - Perfect uniformity is not desirable

- Visual Coherence
  - Are semantically related elements grouped together?
  - Recursive sum of ratio of related to total pairs grouped together

- These are probably the most useful for “the designer who lacks an eye for layout”

Analytic Evaluation

Inspection and Heuristics
Expert Evaluation

• **Studied Ignorance**
  - Pretend you are a novice user; identify usability problems

• **Stress testing**
  - Violate task sequence, click and type a lot, enter every special character on the keyboard, etc.

• **Exhaustive Exploration**
  - Examine the entire interface, looking for consistency, things that don’t work

• **Can be fast and cost effective**

---

Expert Evaluation: Issues

• **Requires**
  - Expertise in HCI
  - Expertise in the application area
  - Ability to role play the novice
  - Objectivity (not a developer)

• **Problems**
  - Experts are biased
  - Hard to find experts
  - Does not increase skill of development team
  - Novices do the weirdest things! (which experts may not anticipate)
Walkthroughs

- **Structured form of usage simulation**
  - Identify task, context, and user population
  - Walk through task, predicting user behavior

- **Variations:**
  - **Cognitive walkthrough:**
    - simulate cognitive processing of user ... tedious!
  - **Pluralistic walkthrough:**
    - multiple types of experts (designers, users, usability experts)
    - Each decides on action and assessment at each step, and then discuss

Cognitive Walkthroughs

- Designer presents an aspect of the design and usage scenarios
- **One of more experts walk through the design prototype with the scenario**
- Expert is told the assumptions about user population, context of use, task details
- Experts are guided by 3 questions
  - Will the correct action be sufficiently evident?
  - Will the user notice that the correct action is available?
  - Will the user associate and interpret the response from the action correctly?
Pluralistic walkthrough

- Variation on the cognitive walkthrough
- Performed by a carefully managed team that includes developers and users
- For each screen
  - Each panelist writes down what they would do
  - They compare their responses (users going first) and discuss
  - Then there is managed discussion that leads to agreed decisions
- Works well for participatory design
- Slow

Heuristic Evaluation (Nielsen)

- Conducted by experts
  - Expertise in both usability and domain
- Inspection guided by usability heuristics
  - Based on Design Guidelines
- Two passes
  - Inspect flow of interface from screen
  - Inspect each screen one at a time against heuristics
- 50% of the problems with two evaluators
- 75% of the problems with 5 evaluators
Nielsen's heuristics

- Visibility of system status
- Match between system and real world
- User control and freedom
- Consistency and standards
- Help users recognize, diagnose, recover from errors
- Error prevention
- Recognition rather than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help and documentation

Doing Heuristic Evaluation

- Briefing session to tell experts what to do
- Evaluation period of 1-2 hours in which:
  - Each expert works separately
  - Take one pass to get a feel for the product
  - Take a second pass to focus on specific features
- Debriefing session in which experts work together to prioritize problems
What about UCD?

Operational Model (environmental and contextual factors)

User Roles

Use Cases

Content Model

Visualization and Interaction Design

Domain Model (glossary, data model, or object class model)

They show how to use analytic techniques on EUC and Content Model (as well as visual design and prototypes)

Collaborative Usability Inspection

• Constantine & Lockwood’s hybrid of Pluralistic and Heuristic
• Team of developers, end users, domain experts, usability experts
• Allows transfer of expertise
• Focus on finding defects: no other debate allowed
• Roles: Lead reviewer, inspection recorder, continuity reviewer
Additional Methods (Next Week)

- Subjective
- Performance
  - Usability Testing
  - Experiments