User Modeling –
Cognitive & Physical Models

John Stasko
Spring 2007

This material has been developed by Georgia Tech HCI faculty, and continues to evolve. Contributors include Gregory Abowd, Al Badre, Jim Foley, Elizabeth Mynatt, Jeff Pierce, Colin Potts, Chris Shaw, John Stasko, and Bruce Walker. Permission is granted to use with acknowledgement for non-profit purposes. Last revision: January 2007.

Agenda

• User modeling – cognitive models
  – Model Human Processor
  – GOMS
  – Cognitive Complexity Theory
  – Keystroke-level models

• Physical modeling
  – Fitt’s Law
User/Cognitive Modeling

- Idea: If we can build a model of how a user works, then we can predict how s/he will interact with the interface
  - Predictive modeling, predictive evaluation

Modeling Goals

- Goals (Salvendy, 1997):
  1. Predict performance of design alternatives
  2. Evaluate suitability of designs to support and enhance human abilities and limitations
  3. Generate design guidelines that enhance performance and overcome human limitations

Note: Not even a mockup is required
Components

- Model some aspects of user’s understanding, knowledge, intentions and processing
- Vary in representation levels: high level plans and problem-solving to low level motor actions such as keypresses

Differing Approaches

- Human as information processing machine
  - “Procedural models”
  - Many subfamilies and related models
- Human as biomechanical machine
- Human as a social actor in context
  - Situation action
  - Activity theory
  - Distributed cognition
Cognitive Models

1. Model Human Processor

- Consider humans as information processing systems
  - Predicting performance
  - Not deciding how one would act
  - A “procedural” model
    - People learn to use products by generating rules for their use and “running” their mental model while interacting with system

- From Card, Moran, and Newell (1980’s)
MHP Components

- Set of memories and processors together
- Set of “principles of operation”
- Discrete, sequential model
- Each stage has timing characteristics (add the stage times to get overall performance times)

3 Subsystems

- Perceptual, cognitive and motor
  - Cycle times ($\tau$):
    - $\tau_p \approx 100$ ms ("middle man" values)
    - $\tau_c \approx 70$ ms
    - $\tau_m \approx 70$ ms
- Perceptual and cognitive have memories
- Fundamental recognize-act cycle of behavior
  - Contents of working memory trigger actions held in long-term memory
Perceptual System

- Consists of sensors and associated buffer memories
  - Most important memories being visual image store and audio image store
  - Hold output of sensory system while it is being symbolically coded
Cognitive System

- Receives symbolically coded information from sensory image stores in its working memory
- Uses that with previously stored information in long-term memory to make decisions on how to respond

Motor System

- Carries out appropriate response
Principles of Operation

- Set of principles that describe how behavior occurs (based on experimental findings about humans)
  - Recognize-act cycle, variable perceptual processor rate, encoding specificity, discrimination, variable cognitive processor rate, Fitt’s law, Power law of practice, uncertainty, rationality, problem space

Applying the MHP

- Example: Designing menu displays
  - 16 menu items in total
  - Breadth (1x16) vs. Depth (4x4)?
MHP: Calculations

**Breadth (1x16):**
- $\tau_p$ perceive item, transfer to WM
- $\tau_c$ retrieve meaning of item, transfer to WM
- $\tau_c$ Match code from displayed to needed item
- $\tau_c$ Decide on match
- $\tau_m$ Execute eye mvmt to (a) menu item number (go to step 6) or (b) to next item (go to step 1)
- $\tau_p$ Perceive menu item number, transfer to WM
- $\tau_c$ Decide on key
- $\tau_m$ Execute key response

Time = $\left(\frac{16+1}{2}\right) (\tau_p + 3\tau_c + \tau_m) + \tau_p + \tau_c + \tau_m$

Time = 3470 msec

Serial terminating search over 16 items

**Depth (4x4):**

Same as for breadth, but with 4 choices, and done up to four times (twice, on average):

Time = $2 \times \left[ \frac{(4+1)}{2} (\tau_p + 3\tau_c + \tau_m) \right] + \tau_p + \tau_c + \tau_m$

Time = 2380 msec

Therefore, in this case, 4x4 menu is predicted to be faster than 1x16.

---

Related Modeling Techniques

- Many techniques fall within this “human as information processor” model

- Common thread - hierarchical decomposition
  - Divide behaviors into smaller chunks
  - Questions:
    - What is unit chunk?
    - When to start/stop?
2. GOMS

- **Goals, Operators, Methods, Selection Rules**
  - Developed by Card, Moran and Newell ’83
- Probably the most widely known and used technique in this family

Assumptions

- “Expert” is performing UI operations
- Interacting with system is problem solving
- Decompose into subproblems
- Determine goals to attack problem
- Know sequence of operations used to achieve the goals
- Timing values for each operation
Goal

- End state trying to achieve
- Then decompose into subgoals

Moved sentence

Select sentence
Cut sentence
Paste sentence

Move to new spot
Place it

Operators

- Basic actions available for performing a task (lowest level actions)

- Examples: move mouse pointer, drag, press key, read dialog box, ...
Methods

- Sequence of operators (procedures) for accomplishing a goal (may be multiple)

- Example: Select sentence
  - Move mouse pointer to first word
  - Depress button
  - Drag to last word
  - Release

Selection Rules

- Invoked when there is a choice of a method

- GOMS attempts to predict which methods will be used

- Example: Could cut sentence either by menu pulldown or by ctrl-x
GOMS Procedure

- Walk through sequence of steps
- Determine branching tree of operators, methods, and selections, and add up the times

GOMS: Example

- Menu structure (breadth vs. depth, again)
- Breadth (1x16):
  - **Goal**: perform command sequence
  - **Goal**: perform unit task of the command
  - **Goal**: determine which unit task to do
    - **Operator**: Look at screen, determine next command
    - **Goal**: Execute unit task
    - **Select**: Which method to enter number of command
e.g. IF item # between 1 & 9 THEN use 1-KEY METHOD
    - **Operator**: Use 1-Key Method
    - **Operator**: Verify Entry... etc.

Result: Average Number of Steps = 33
GOMS: Example, cont’d

- Depth (4x4):
  - Similar steps, in slightly different order and looping conditions
    - Result: Average Number of Steps = 24

- Comparison: Depth is ~25% faster in this case
  - Card et al. did not specify step length (in time)
  - Assume 100msec/step, then depth is 0.9 sec faster
  - Similar to Model Human Processor results

Application

- NYNEX telephone operation system
  - GOMS analysis used to determine critical path, time to complete typical task
  - Determined that new system would actually be slower
  - Abandoned, saving millions of dollars
Limitations

- GOMS is not for
  - Tasks where steps are not well understood
  - Inexperienced users
- Why?

GOMS Variants

- GOMS is often combined with a keystroke level analysis
  - KLM - Keystroke level model
  - Analyze only observable behaviors such as keypresses, mouse movements
  - Low-level GOMS where method is given
  - Assumes error-free performance
KLM Characteristics

- Operators:
  - K: keystroke, mouse button push
  - P: point with pointing device
  - D: move mouse to draw line
  - H: move hands to keyboard or mouse
  - M: mental preparation for an operation
  - R: system response time

- Tasks split into two phases
  - Acquisition of task - user builds mental rep.
  - Execution of task - using system facilities

Procedure

- How KLM works
  - Assigns times to different operators
  - Plus: Rules for adding M’s (mental preparations) in certain spots

- Chart on next slide
Table 8.1 Performance times for keystroke-level operations; from Card et al. (1983).

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description and remarks</th>
<th>Time (s)</th>
</tr>
</thead>
</table>
| K              | PRESS KEY OR BUTTON. Pressing the shift or control key counts as a separate K operation. Time varies with the typing skill of the user; the following shows the range of typical values:   
  - Best typist (330 wpm) 0.05  
  - Good typist (90 wpm) 0.12  
  - Average skilled typist (55 wpm) 0.33  
  - Average non-secondary typist (40 wpm) 0.50  
  - Typing random letters 0.50  
  - Typing complex codes 0.75  
  - Worst typist (unfamiliar with keyboard) 1.20  |
| P              | POINT WITH MOUSE TO TARGET ON A DISPLAY. The time to point varies with distance and target size according to Piirt's Law, ranging from 0.8 to 1.5 sec, with 1.2 being an average. This operator does not include the (2 sec) button press that often follows. Mouse pointing time is also a good estimate for other efficient analog pointing devices, such as joysticks (see Chapter 7). |
| H              | HOME HANDS ON KEYBOARD OR OTHER DEVICE.                                                 | 0.69     |
| D(s)           | DRAW s STRAIGHT-LINE SEGMENTS OF TOTAL LENGTH L CM.                                      | 0.80 + 0.58  
  This is a very restricted operator; it assumes that drawing is done with the mouse on a system that constrains all lines to fall on a square 100 cm grid. Users vary in their drawing skill; the time given is an average value. |
| M              | MENTALLY PREPARE. Different commands require different response times. The response time is counted only if it causes the user to wait.                                             | 1.35     |
| R(s)           | RESPONSE BY SYSTEM.                                                                      |          |

Begin with a method of encoding that includes all physical operations and response operations. Use Rule 0 to place candidate M's, and then cycle through Rules 1 to 4 for each M to see where it should be deleted.

**Rule 0.** Insert M's in front of all K's that are not part of text or numeric argument strings proper (e.g., text or numbers). Place M's in front of all P's that select commands (not arguments).

**Rule 1.** If an operator following an M is fully anticipated in an operator previous to M, then delete the M (e.g., PMK → PK).

**Rule 2.** If a string of M's belongs to a cognitive unit (e.g., the name of a command), then delete all M's but the first.

**Rule 3.** If a K is a redundant terminator (e.g., the terminator of a command immediately following the terminator of its argument), then delete the M in front of it.

**Rule 4.** If a K terminates a constant string (e.g., a command name), then delete the M in front of it; but if the K terminates a variable string (e.g., an argument string) then keep the M in front of it.
Example

Move Sentence

1. Select sentence
   Reach for mouse  H  0.40
   Point to first word  P  1.10
   Click button down  K  0.60
   Drag to last word  P  1.20
   Release  K  0.60
   3.90 secs

2. Cut sentence
   Press, hold ^
   Point to menu
   Press and release ‘x’ or Press and hold mouse
   Move to “cut”
   Release

3. ...

Example of KLM

- Breadth menu (1x16)
  - M: Search 16 items
  - 1 or 2 K: Enter 1 or 2-digit number
  - K: Press return key

Time = M + K(first digit) + 0.44K(second digit) + K(Enter)
   (Look up values, and when to apply “M” operator)
Time = 1.35 + 0.20 + 0.44(0.20) + 0.20 = 1.84 seconds

Note: Many assumptions about typing proficiency, M’s, etc.
Also ignores most of the time spent determining which task to perform, and how to perform it.
Example of KLM, cont’d

• Depth menu (4x4)
  – M: Search 4 items
  – K: Enter 1-digit number (no M, since expert user)
  – K: Press return key
  Time = M + K(Digit) + K(Enter)
  Time = 1.35 + 0.20 + 0.20 = 1.75 seconds

• Compare the various models in terms of times and predictions:
  – Vary in times, but not in performance predictions

Other GOMS Variants

• NGOMSL (Kieras)
  – Very similar to GOMS
  – Goals expressed as noun-action pair, eg., delete word
  – Same predictions as other methods
  – More sophisticated, incorporates learning, consistency
  – Handles expert-novice difference, etc.
3. Production Systems

- IF-THEN decision trees (Kieras & Polson '85)
  - Cognitive Complexity Theory
  - Uses goal decomposition from GOMS and provides more predictive power
  - Goal-like hierarchy expressed using production rules
    - if condition, then action
    - Makes a generalized transition network
- Very long series of decisions
  - Note: In practice, very similar to NGOMSL
    - Bovair et al (1990) claim they are identical
      - NGOMSL model easier to develop
      - Production systems easier to program

4. Grammars

- To describe the interaction, a formalized set of productions rules (a language) can be assembled.
- “Grammar” defines what is a valid or correct sequence in the language.
- Used to determine the consistency of a system design
Task Action Grammars (TAG)

- Payne & Green (1986, 1989)
- Concentrates on overall structure of language rather than separate rules
- Designed to predict relative complexity of designs
- Not for quantitative measures of performance or reaction times.
- Consistency & learnability determined by similarity of rules

Summary of Cognitive Models

1. Model Human Processor (MHP)
2. GOMS
   - Basic model
   - Keystroke-level models (KLM)
   - NGOMSL
3. Production systems
   - Cognitive Complexity Theory
4. Grammars
Modeling Problems

• 1. Terminology - example
  – High frequency use experts - cmd language
  – Infrequent novices - menus
  – What’s “frequent”, “novice”? 

• 2. Dependent on “grain of analysis” employed
  – Can break down getting a cup of coffee into 7, 20, or 50 tasks
  – That affects number of rules and their types
  – (Same issue as task analysis)

Modeling Problems (contd.)

• 3. Does not involve user per se
  – Don’t inform designer of what user wants

• 4. Time-consuming and lengthy

• 5. One user, one computer model
  – No social context
Lower Level Models

- More physically-based
  - Human as biomechanical machine

- Fitt’s Law
  - Models movement times for selection tasks

- Basic idea: Movement time for a well-rehearsed selection task
  - Increases as the distance to the target increases
  - Decreases as the size of the target increases

Model Description 1

- Move from START to STOP

  ![Diagram](https://via.placeholder.com/150)

  - Index of difficulty
  
    \[ ID = \log_2 \left( \frac{2A}{W} \right) \]
• Index of difficulty

\[ ID = \log_2 \left( \frac{2A}{W} \right) \]

- Both quantities are distances
- So unit-less result

Model Description 2

• MT - Movement time

\[ MT = a + b \times ID \]

MT is a linear function of ID
Exact Equation

- Run empirical tests to determine $a$ and $b$ in $MT = a + b \cdot ID$
- Will get different ones for different input devices and ways the device is used

Common Equation

- $MT = a + b \log_2 (A/W + 1)$
- Provides useful numbers
Questions

- What do you do in 2D?

- Where can this be applied in user interface design?

Video

- IBM keyboard pointing stick
Upcoming

- Descriptive Cognitive Models
  - Social context

- Evaluation
  - Experimental design
  - Data collection
  - Subjective measures
  - Data analysis