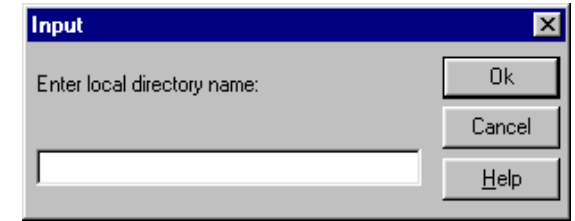


Theories, Principles, and Guidelines

Design of Everyday Things



Today's Experiment

What screen colors do you respond to best / worst?

Procedure


- Type in your Age
- Press the letter on the screen
- Do not pause during test.
- Work as quickly as you can

HCI in the News

It's not just a phone, it's an adventure The New York Times ON THE WEB
Published: April 3, 2005, 5:00 AM PDT
By Michel Manoff and Katie Hafner
The New York Times
Larry Azlin, a software engineer in El Cerrito, Calif., considers himself one of the lucky ones. His aging clamshell cell phone, a Motorola V60, seems to work just fine. But once he gives it some thought, it occurs to him that he does have a few complaints.

"The buttons on the sides are a bit annoying," he said. They seem to do different things when the phone is open and when it is closed.

His biggest complaint is that the phone insists on making noise at every opportunity. "You can't even turn it off without it making a sound," he said, noting that when he tried to discreetly silence the phone at a concert, it squawked.



Azlin is hardly alone in being confused and confounded by his cell phone at times. Gone are the days when the most one expected from a mobile phone was to place or to receive a call.

Practically every new iteration of cell phone promises more: digital music, streaming video, 3D video games, location-based navigation and full Internet browsing, not to mention a camera. With more features often come more buttons, complications and costs, and thicker operating manuals.

Some people call it feature creep.

Feature Creep

Multifunction

Each function good but not excellent...

Chapter 2

Guidelines, Principles, and Theories

Guidelines

- Shared language
- Best practices
- Critics
 - Too specific, incomplete, hard to apply, and sometimes wrong
- Proponents
 - Encapsulate experience

Navigating the interface

- Sample of the National Cancer Institutes guidelines:
- Standardize task sequences
 - Ensure that embedded links are descriptive
 - Use unique and descriptive headings
 - Use check boxes for binary choices
 - Develop pages that will print properly
 - Use thumbnail images to preview larger images

Accessibility guidelines

- Provide a text equivalent for every nontext element
- For any time-based multimedia presentation synchronize equivalent alternatives
- Information conveyed with color should also be conveyed without it
- Title each frame to facilitate from identification and navigation

Organizing the display

- Smith and Mosier (1986) offer five high-level goals
 - Consistency of data display
 - Efficient information assimilation by the user
 - Minimal memory load on the user
 - Compatibility of data display with data entry
 - Flexibility for user control of data display

Getting the user's attention

- Intensity (2)
- Marking
- SIZE (4)
- Fonts (3)
- Inverse Video
- Blinking (2 to 4 Hz)
- Colors (4)
- Audio

Facilitating data entry

- Smith and Mosier (1986) offer five high-level objectives as part of their guidelines for data entry
 - Consistency of data-entry transactions
 - Minimal input actions by user
 - Minimal memory load on users
 - Compatibility of data entry with data display
 - Flexibility for user control of data entry

Principles

- More fundamental, widely applicable, and enduring than guidelines
- Need more clarification
- Fundamental principles
 - Determine user's skill levels
 - Identify the tasks
- Five primary interaction styles
- Eight golden rules of interface design
- Prevent errors
- Automation and human control

Determine user's skill levels

- "Know thy user" Hansen (1971)
- Age, gender, physical and cognitive abilities, education, cultural or ethnic background, training, motivation, goals and personality
- Design goals based on skill level
 - Novice or first-time users
 - Knowledgeable intermittent users
 - Expert frequent users
- Multi-layer designs

Identify the tasks

- Task Analysis usually involve long hours observing and interviewing users
- Decomposition of high level tasks
- Relative task frequencies

Job title	TASK				
	Query by Patient	Update Data	Query across Patients	Add Relations	Evaluate System
Nurse	0.14	0.11			
Physician	0.06	0.04			
Supervisor	0.01	0.01	0.04		
Appointment personnel	0.26				
Medical record maintainer	0.07	0.04	0.04	0.01	
Clinical researcher			0.08		
Database programmer			0.02	0.02	0.05

Choose an interaction style

- Direct Manipulation
- Menu selection
- Form fillin
- Command language
- Natural language

Advantages	Disadvantages
Direct manipulation Visually presents task concepts Allows easy learning Allows easy retention Allows errors to be avoided Encourages exploration Affords high subjective satisfaction	May be hard to program May require graphics display and pointing devices
Menu selection Reduces learning Reduces keystrokes Structures decision making Permits use of dialog management tools Allows easy support of error handling	Presents danger of menu errors May slow frequent users Consumes screen space Requires rapid display rate
Form fillin Simplifies data entry Requires modest training Gives convenient assistance Permits use of form-management tools	Consumes screen space
Command language Is flexible Appeals to "power" users Supports user initiative Allows convenient creation of user-defined macros	Has poor error handling Requires substantial training and memorization
Natural language Relieves burden of learning syntax	Requires clarification dialog May not show context May require more keystrokes Is unpredictable

8 golden rules of interface design

- Strive for consistency
- Cater to universal usability
- Offer informative feedback
- Design dialogs to yield closure
- Prevent errors
- Permit easy reversal of actions
- Support internal locus of control
- Reduce short term memory

Prevent errors

- Make error messages specific, positive in tone, and constructive
- Mistakes and slips (Norman, 1983)
- Correct actions
 - Gray out inappropriate actions
 - Selection rather than freestyle typing
 - Automatic completion
- Complete sequences
 - Single abstract commands
 - Macros and subroutines

Automation and human control

Humans Generally Better

Sense low level stimuli
 Detect stimuli in noisy background
 Recognize constant patterns in varying situations
 Sense unusual and unexpected events
 Remember principles and strategies
 Retrieve pertinent details without a priori connection
 Draw on experience and adapt decisions to situation
 Select alternatives if original approach fails
 Reason inductively; generalize from observations
 Act in unanticipated emergencies and novel situations
 Apply principles to solve varied problems
 Make subjective evaluations
 Develop new solutions
 Concentrate on important tasks when overload occurs
 Adapt physical response to changes in situation

Machines Generally Better

Sense stimuli outside human's range
 Count or measure physical quantities
 Store quantities of coded information accurately
 Monitor prespecified events, especially infrequent ones
 Make rapid and consistent responses to input signals
 Recall quantities of detailed information accurately
 Process quantitative data in proscribed ways
 Reason deductively; infer from a general principle
 Perform repetitive preprogrammed actions reliably
 Exert great, highly controlled physical force
 Perform several activities simultaneously
 Maintain operations under heavy information load
 Maintain performance over extended periods of time

Automation and human control

Successful integration:

Users can avoid:

Routine, tedious, and error prone tasks

Users can concentrate on:

Making critical decisions, coping with unexpected situations, and planning future actions

Automation and human control

Supervisory control needed to deal with real world open systems

E.g. air-traffic controllers with low frequency, but high consequences of failure

FAA: design should place the user in control and automate only to improve system performance, without reducing human involvement

Automation and human control

Goals for autonomous agents

knows user's likes and dislikes

makes proper inferences

responds to novel situations

performs competently with little guidance

Tool like interfaces versus autonomous agents

Aviators representing human users, not computers, more successful

Automation and human control

User modeling for adaptive interfaces

keeps track of user performance

adapts behavior to suit user's needs

allows for automatically adapting system

response time, length of messages, density of feedback, content of menus, order of menu items, type of feedback, content of help screens

can be problematic

system may make surprising changes

user must pause to see what has happened

user may not be able to

predict next change

interpret what has happened

restore system to previous state

Automation and human control

Alternative to agents:

user control, responsibility, accomplishment

expand use of control panels

style sheets for word processors

specification boxes of query facilities

information-visualization tools

Theories

Beyond the specifics of guidelines

Principles are used to develop theories

Descriptions/explanatory or predictive

Motor task, perceptual, or cognitive

Explanatory and predictive theories

Explanatory theories:

Observing behavior

Describing activity

Conceiving of designs

Comparing high-level concepts of two designs

Training

Predictive theories:

Enable designers to compare proposed designs for execution time or error rates

Perceptual, Cognitive, & Motor

Perceptual or Cognitive subtasks theories

Predicting reading times for free text, lists, or formatted displays

Motor-task performance times theories:

Predicting keystroking or pointing times

Taxonomy (explanatory theory)

- Order on a complex set of phenomena
- Facilitate useful comparisons
- Organize a topic for newcomers
- Guide designers
- Indicate opportunities for novel products.

Models

- Foley and van Dam four-level approach
 - Conceptual level:*
 - User's mental model of the interactive system
 - Semantic level:*
 - Describes the meanings conveyed by the user's command input and by the computer's output display
 - Syntactic level:*
 - Defines how the units (words) that convey semantics are assembled into a complete sentence that instructs the computer to perform a certain task
 - Lexical level:*
 - Deals with device dependencies and with the precise mechanisms by which a user specifies the syntax
- Approach is convenient for designers
 - Top-down nature is easy to explain
 - Matches the software architecture
 - Allows for useful modularity during design

Stages of action models

- Norman's seven stages of action
 - Forming the goal
 - Forming the intention
 - Specifying the action
 - Executing the action
 - Perceiving the system state
 - Interpreting the system state
 - Evaluating the outcome
- Norman's contributions
 - Context of cycles of action and evaluation.
 - Gulf of execution:* Mismatch between the user's intentions and the allowable actions
 - Gulf of evaluation:* Mismatch between the system's representation and the users' expectations

Stages of action models (cont.)

- Four principles of good design
 - State and the action alternatives should be visible
 - Should be a good conceptual model with a consistent system image
 - Interface should include good mappings that reveal the relationships between stages
 - User should receive continuous feedback
- Four critical points where user failures can occur
 - Users can form an inadequate goal
 - Might not find the correct interface object because of an incomprehensible label or icon
 - May not know how to specify or execute a desired action
 - May receive inappropriate or misleading feedback

GOMS

- Goals, operators, methods, and selection rules (GOMS) model
- Keystroke-level model: Predict performance times for error-free expert performance of tasks
- Transition diagrams
- Natural GOMS Language (NGOMSL)
- Several alternative methods to delete fields, e.g.
 - Method 1 to accomplish the goal of deleting the field:
 - Decide: If necessary, then accomplish the goal of selecting the field
 - Accomplish the goal of using a specific field delete method
 - Report goal accomplished

GOMS

- Method 2 to accomplish the goal of deleting the field:
 - Decide: If necessary, then use the Browse tool to go to the card with the field
 - Choose the field tool in the Tools menu
 - Note that the fields on the card background are displayed
 - Click on the field to be selected
 - Report goal accomplished
- Selection rule set for goal of using a specific field-delete method:
 - If you want to past the field somewhere else, then choose "Cut Field" from the Edit menu.
 - If you want to delete the field permanently, then choose "Clear Field" from the Edit menu.
 - Report goal accomplished.

Consistency through grammars

Consistent user interface goal

- Definition is elusive - multiple levels sometimes in conflict
- Sometimes advantageous to be inconsistent.

Consistent	Inconsistent A	Inconsistent B
delete/insert character	delete/insert character	delete/insert character
delete/insert word	remove/bring word	remove/insert word
delete/insert line	destroy/create line	delete/insert line
delete/insert paragraph	kill/birth paragraph	delete/insert paragraph

Consistency through grammars

Inconsistent action verbs

- Take longer to learn
- Cause more errors
- Slow down users
- Harder for users to remember

Consistency through grammars

- Task-action grammars (TAGs) try to characterize a complete set of tasks.

Example: TAG definition of cursor control

Dictionary of tasks:

move-cursor-one-character-forward	[Direction=forward,Unit=char]
move-cursor-one-character-backward	[Direction=backward,Unit=char]
move-cursor-one-word-forward	[Direction=forward,Unit=word]
move-cursor-one-word-backward	[Direction=backward,Unit=word]

Consistency through grammars

High-level rule schemas describing command syntax:

task [Direction, Unit] -> symbol [Direction] + letter [Unit]
symbol [Direction=forward] -> "CTRL"
symbol [Direction=backward] -> "ESC"
letter [Unit=word] -> "W"
letter [Unit=char] -> "C"

Generates a consistent grammar:

move cursor one character forward CTRL-C
move cursor one character backward ESC-C
move cursor one word forward CTRL-W
move cursor one word backward ESC-W

Widget-level theories

Follow simplifications made in higher-level, user-interface building tools.

Potential benefits:

Possible automatic generation of performance prediction

A measure of layout appropriateness available as development guide

Estimates generated automatically and amortized over many designers and projects

perceptual complexity

cognitive complexity

motor load

Higher-level patterns of usage appear

Object/Action Interface model

Syntactic-semantic model of human behavior

used to describe

programming

database-manipulation facilities

direct manipulation

Distinction made between meaningfully-acquired semantic concepts and rote-memorized syntactic details

Semantic concepts of user's tasks well-organized and stable in memory

Syntactic details of command languages arbitrary and required frequent rehearsal

Object/Action Interface model

With introduction of GUIs, emphasis shifted to simple direct manipulations applied to visual representations of objects and actions.

Syntactic aspects not eliminated, but minimized.

Object/Action Interface model

Object-action design:

understand the task.

real-world objects

actions applied to those object

create metaphoric representations of interface objects and actions

designer makes interface actions visible to users

Task hierarchies

Decomposition of real-world complex systems natural

human body

buildings

cities

symphonies

baseball game

Task hierarchies

Computer system designers must generate a hierarchy of objects and actions to model users' tasks:

Representations in pixels on a screen

Representations in physical devices

Representations in voice or other audio cue

Interface hierarchies

Interface includes hierarchies of objects and actions at high and low levels. E.g. A computer system:

Interface Objects
rectory
 name
 length
 date of creation
 owner
 access control
files of information
 lines
 diffields
 characters
 fonts
 pointers
 binary numbers

Interface hierarchies

Interface Actions

load a text data file

insert into the data file

save the data file

 save the file

 save a backup of the file

 apply access-control rights

 overwrite previous version

 assign a name

Interface hierarchies

Interface objects and actions based on familiar examples.

Users learn interface objects and actions by:

- seeing a demonstration
- hearing an explanation of features
- conducting trial-and-error sessions

The disappearance of syntax

Users must maintain a profusion of device-dependent details in their human memory.

- Which action erases a character
- Which action inserts a new line after the third line of a text file
- Which abbreviations are permissible
- Which of the numbered function keys produces the previous screen.

The disappearance of syntax

- Learning, use, and retention of this knowledge is hampered by two problems
 - Details vary across systems in an unpredictable manner
 - Greatly reduces the effectiveness of paired-associate learning
- Syntactic knowledge conveyed by example and repeated usage
- Syntactic knowledge is system dependent

The disappearance of syntax

- Minimizing these burdens is the goal of most interface designers
- Modern direct-manipulation systems
- Familiar objects and actions representing their task objects and actions.
- Modern user interface building tools
- Standard widgets



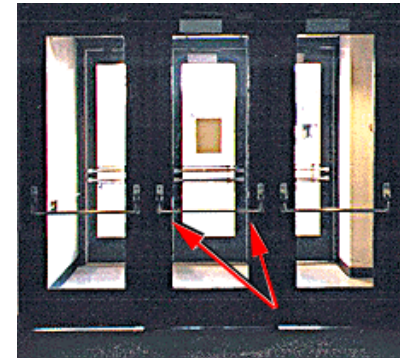
VCR Clocks

Doors

Phones

Doors

Which Side to Push On?



Sidewalks



Figure out where people want to go first

What Side is the Gas Tank On?



Conceptual Models

Confusions Arise When

Conceptual

Conflicts with

Reality

Conceptual Models



Conceptual Models

Power Window Buttons



Mapping

Action = Reaction

Relationship between
Control and Result

Feedback

Design

The Stove Top

Bad Design



Good
Design



Homework

Myers-Briggs Type Indicator
Download and unzip MBTI.zip
Answer the questions
Save the file as *lastname.fi.mbt*
Email it to
submit.homework@gmail.com

Subject line: MBTI *name*
Email it Sunday 6pm
Only send it once!

End of This Lesson