Design Heuristics

Evaluating Without Users

- Goal
  - Evaluate the evolving design when no users are present
  - Catch problems that an evaluation with only a few users may not reveal
- Cognitive Walkthroughs
  - **Task-oriented** technique in the context of task-centered design
  - Path through interface pre-determined
  - One analyzer
- Action Analysis
  - Allows a designer to predict the **time** for an **expert** to perform a task
  - Forces the designer to take a detailed look at the interface
- Heuristic Analysis
  - **Interface** (not task) oriented
  - Overall examination. Path through interface NOT pre-determined
  - Several analyzers

Cognitive Walkthrough

What is it?

- A formalized way of imagining people's thoughts and actions when they use an interface for the first time

Requirements:

- Description or prototype of interface (or a prototype)
- Task description
- Complete, written list of the actions to complete the task
- User background

What you look for:

- Will users know how to perform the action? (Will they try to do it?)
- Will users see the control?
- Will users know if the control does what they want?
- Will users understand the feedback?

Cognitive Walkthrough

How to do it? – an Example

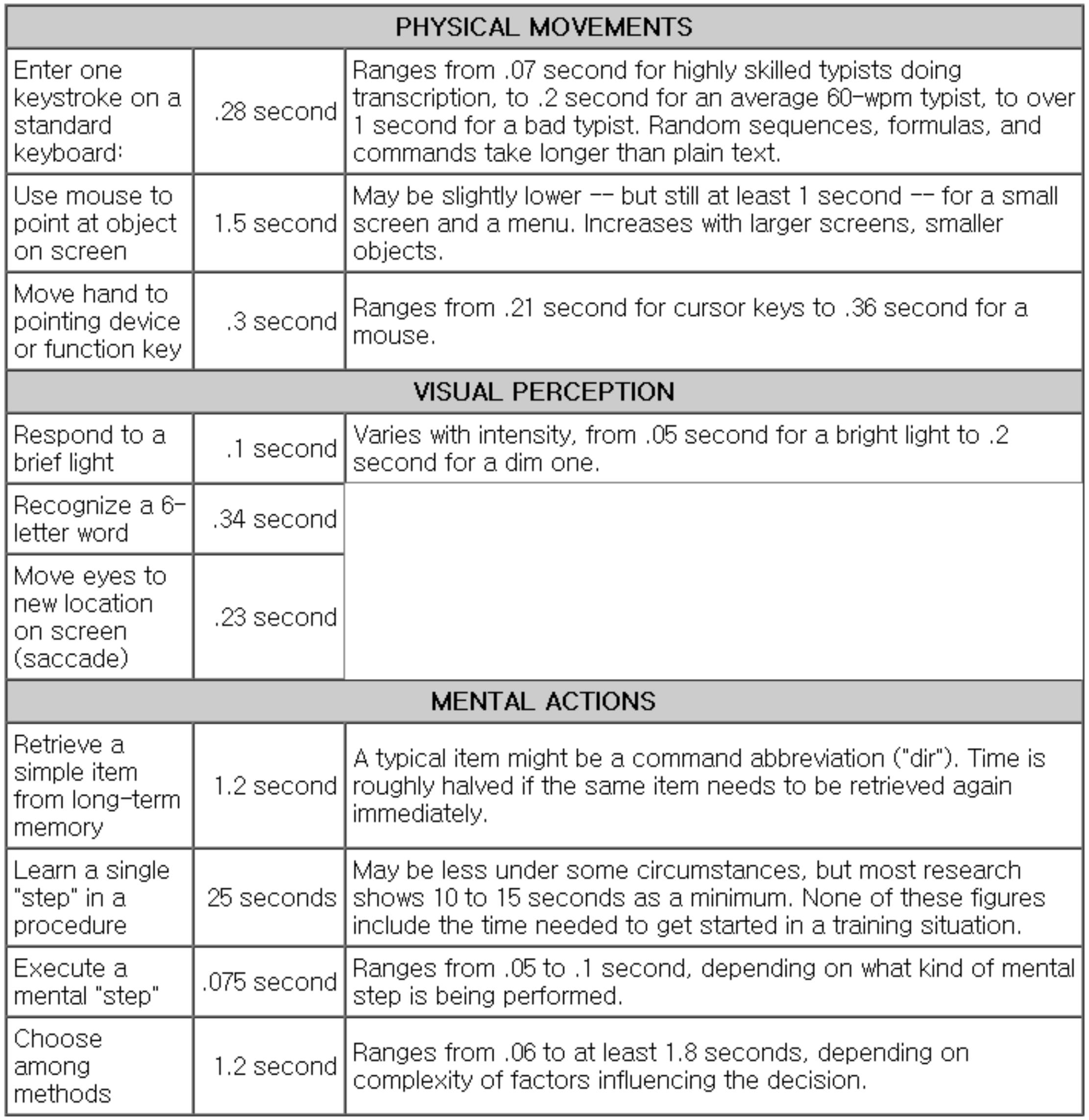
- You have a **prototype** or a detailed design description of the interface, and you know who the users will be.
- You select one of the **tasks** that the design is intended to support.
- Then you try to tell a **believable story about each action** a user has to take to do the task.
- To make the story believable you have to motivate each of the user's actions, relying on the user's general knowledge and on the prompts and feedback provided by the interface.

- If you can't tell a believable story about an action, then you've located a problem with the interface.

http://www.hcibib.org/tcuid/chap-4.html#4-1

Action Analysis

- decide what physical and mental steps a user will perform to complete tasks
- analyze those steps, looking for problems
  - too many steps to perform a simple task
  - too long to perform the task
  - too much to learn about the interface



PHYSICAL MOVEMENTS		
Enter one keystroke on a standard keyboard:	.28 second	Ranges from .07 second for highly skilled typists doing transcription, to .2 second for an average 60-wpm typist, to over 1 second for a bad typist. Random sequences, formulas, and commands take longer than plain text.
Use mouse to point at object on screen	1.5 second	May be slightly lower -- but still at least 1 second -- for a small screen and a menu. Increases with larger screens, smaller objects.
Move hand to pointing device or function key	.3 second	Ranges from .21 second for cursor keys to .36 second for a mouse.
VISUAL PERCEPTION		
Respond to a brief light	.1 second	Varies with intensity, from .05 second for a bright light to .2 second for a dim one.
Recognize a 6-letter word	.34 second	
Move eyes to new location on screen (saccade)	.23 second	
MENTAL ACTIONS		
Retrieve a simple item from long-term memory	1.2 second	A typical item might be a command abbreviation ("dir"). Time is roughly halved if the same item needs to be retrieved again immediately.
Learn a single "step" in a procedure	25 seconds	May be less under some circumstances, but most research shows 10 to 15 seconds as a minimum. None of these figures include the time needed to get started in a training situation.
Execute a mental "step"	.075 second	Ranges from .05 to .1 second, depending on what kind of mental step is being performed.
Choose among methods	1.2 second	Ranges from .06 to at least 1.8 seconds, depending on complexity of factors influencing the decision.

Average times for computer interface actions

Judith Reitman Olson and Gary M. Olson, "The growth of cognitive modeling in human-computer interaction since GOMS," Human-Computer Interaction, 5 (1990), pp. 221-265.

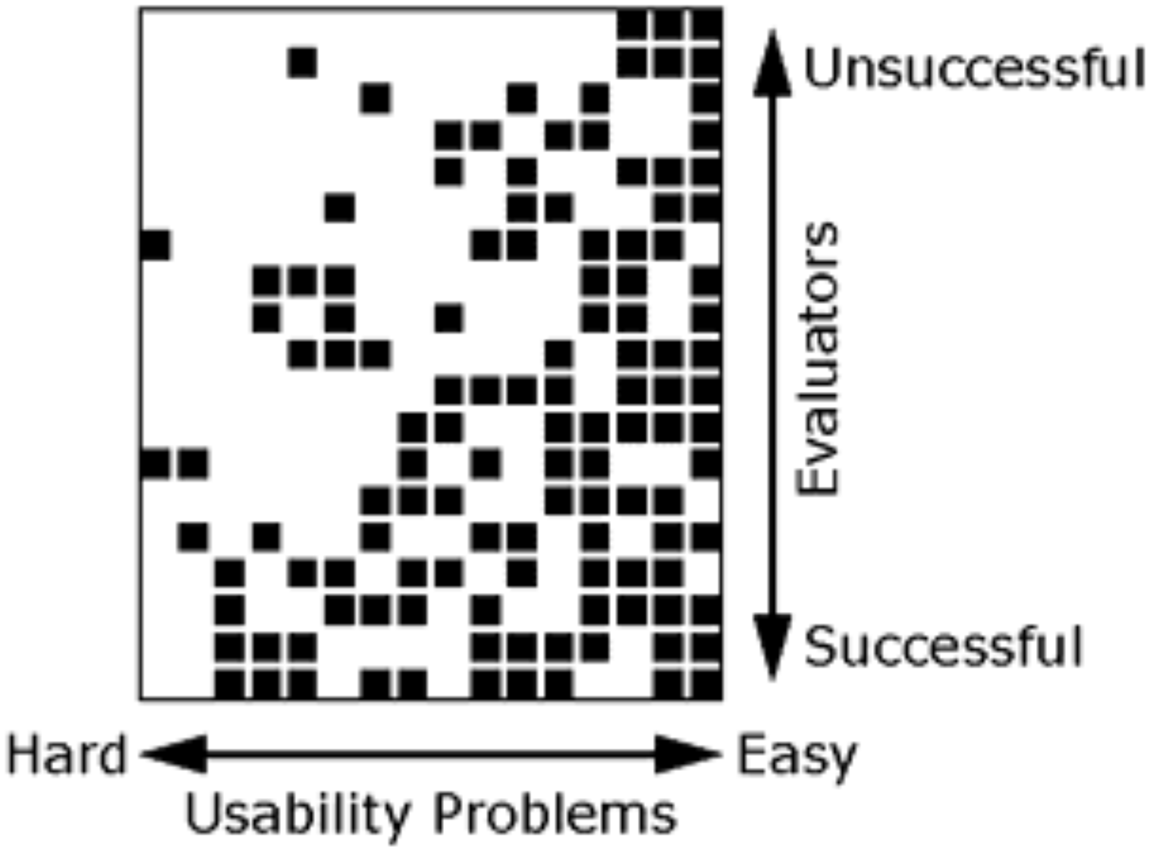
Heuristic Analysis

- "Rules of thumb" that describe features of usable systems
  - Can be used as design principles
  - Can be used to evaluate a design

- Pros and cons
  - Easy and inexpensive
    - *Performed by expert*
    - *No users required*
    - *Catches many design flaws*
  - More difficult than it seems
    - *Not a simple checklist*
    - *Cannot assess how well the interface will address user goals*

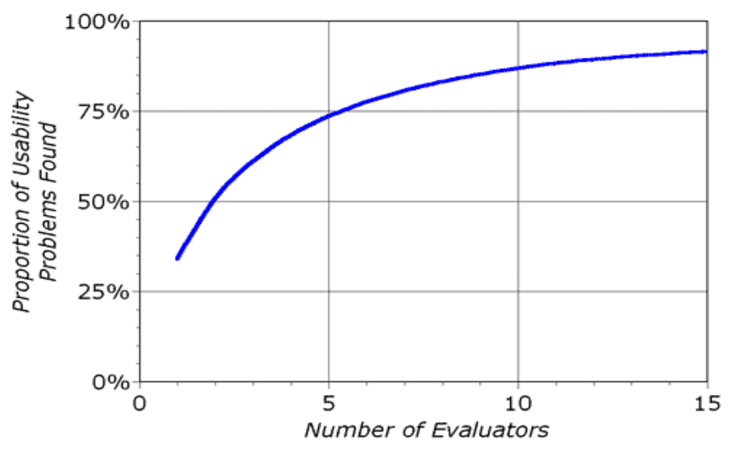
Problems found by a single inspector

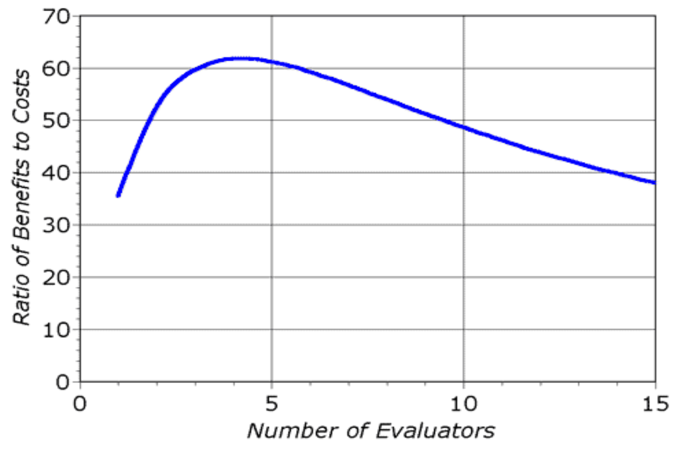
- Evaluators miss both easy and hard problems
  - ‘best’ evaluators can miss easy problems
  - ‘worse’ evaluators can discover hard problems



Usability Engineering

- Introduced by Nielsen (1994)
- Can be performed on working UI or sketches
- Requires a small set (3-5) of evaluators to examine the UI
  - Check compliance with usability principles
    - *Each evaluator works independently*
    - *Go through the interface several times with different perspectives*
  - All reviews are aggregated in one final usability report





Nielsen's evaluation phases (1-2)

- Pre-evaluation training
  - Provide the evaluator with domain knowledge if needed
- Evaluation
  - First step: get a feel for flow and scope
  - Second step: focus on specific elements
    - *Multiple passes approach is better*
    - *Create a list of all problems*
    - *Rate severity of problem*

Nielsen's evaluation phases (3-4)

- Severity rating
  - Performed by individuals
  - Then aggregated by group
  - Establishes a ranking between problems
  - Reflects frequency, impact and persistence
    - *Cosmetic, minor, major and catastrophic*

- Debriefing
  - Discuss outcome with design team
  - Suggest potential solutions
  - Assess how hard things are to fix

Nielsen’s heuristics :

10 general principles for user interface design

- Simple and natural dialog
- Speak the users’ language
- Minimize user memory load
- Consistency
- Feedback
- Clearly marked exits
- Shortcuts
- Prevent errors
- Good error messages
- Provide help and documentation

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

General UI Design Principles: Alternatives

- Donald Norman’s principles of design
  - Adequate Visibility
    - *Affordances*
    - *Visible Constraints*
    - *Natural Mappings*
  - Good Conceptual Model (Mental Model)
  - Feedback – Causality
  - Comfort
  - Consistency /Cultural Standard
  - Transfer Effects

General UI Design Principles: Alternatives

- Ben Shneiderman's 8 golden rules
  - Strive for consistency
  - Cater to **universal usability**
  - Offer informative feedback
  - Design dialogs to yield closure (beginning, middle, and end)
  - Prevent errors
  - Permit easy reversal of actions
  - Support internal locus of control (users are in charge?)
  - Reduce short term memory

General UI Design Principles: Alternatives

- Bruce Tognazzini's Principles of Interaction Design
  - Anticipation
  - Autonomy
  - Color Blindness
  - Consistency
  - Defaults
  - Efficiency of the User
  - Explorable Interfaces
  - Fitts' Law
  - Human Interface Objects
  - Latency Reduction
  - Learnability
  - Metaphors
  - Protect Users' Work
  - Readability
  - Track State
  - Visible Navigation

Jakob, Don, Tog

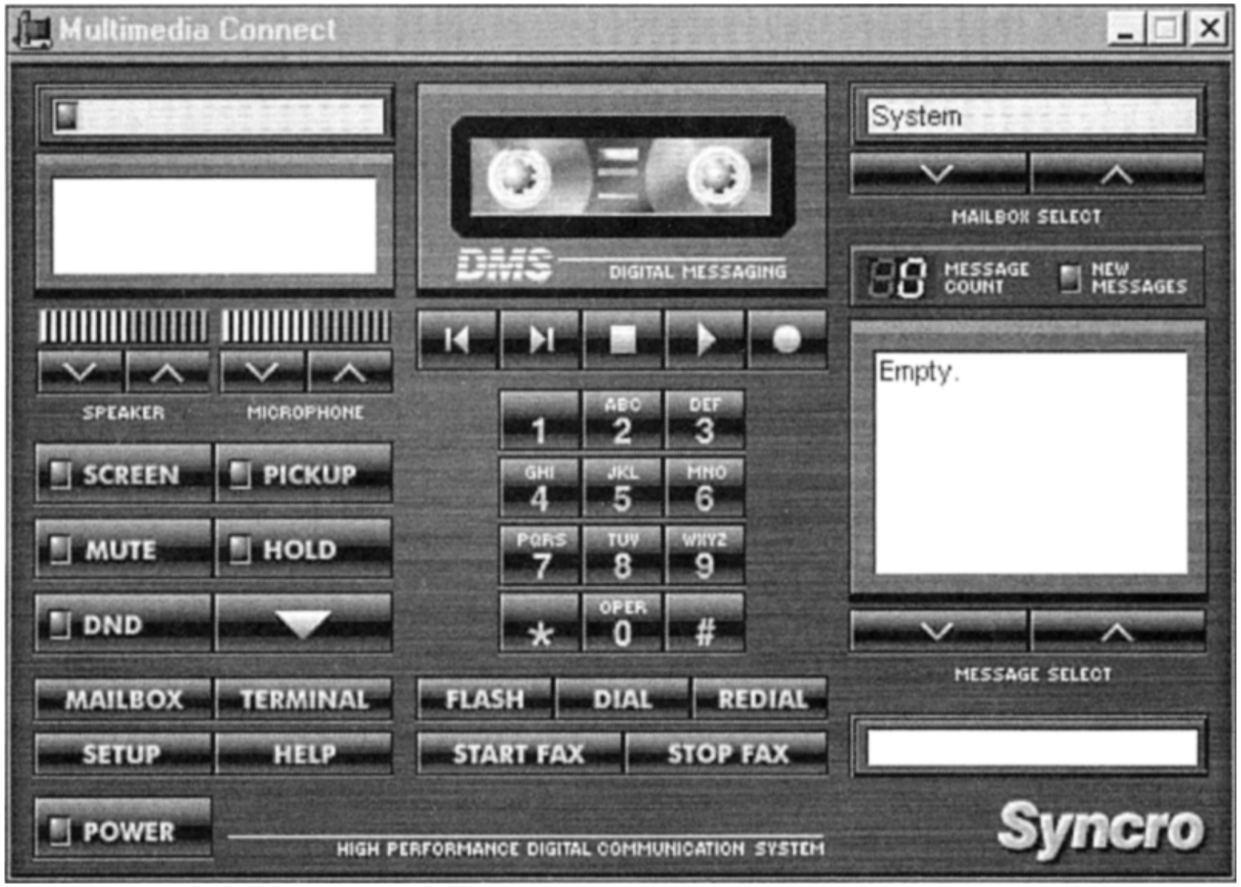
Jakob Nielsen, Ph.D.
"The Guru of Web Page Usability" (New York Times)
Donald A. Norman, Ph.D.
"The Guru of Workable Technology" (Newsweek)
Bruce "Tog" Tognazzini
"Leading Authority on Software Design" (HotWired)

http://www.nngroup.com/

http://www.asktog.com/basics/firstPrinciples.html

Simple and natural dialog

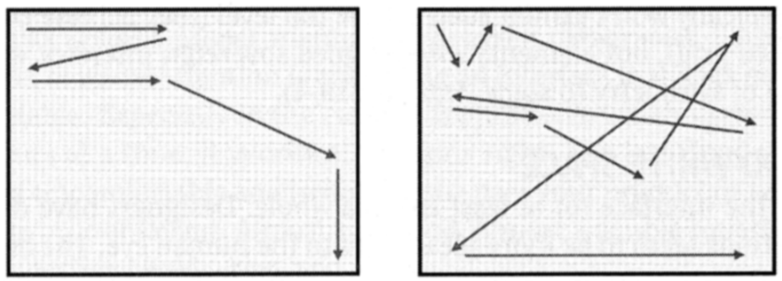
- Minimalist design ("less is more")
  - UI should be simplified as much as possible (reduce learning effort & possibility of errors)
  - UI should match the users task in a natural way



From Cooper's "The inmates are running the asylum"

Simple and natural dialog

- Present information in a natural order (logical flow)

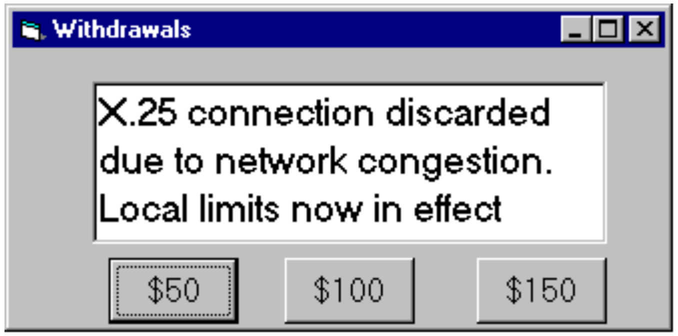


From Cooper's "About face 2.0"

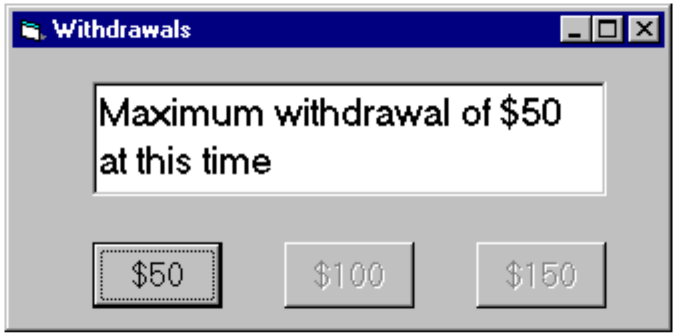
- Simple is good
  - Remove or hide irrelevant or rarely needed information
    - *They compete with important information on screen*
      - Palm Pilot, Dynamic menus
  - Use windows frugally
    - *Avoid complex window management*

Speak the users' language

- Match the real world
- Use a language compatible with users' conceptual model
  - Example: withdrawing money at an ATM

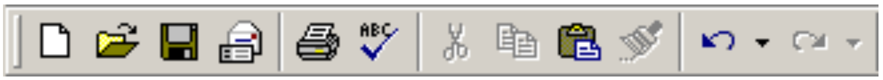
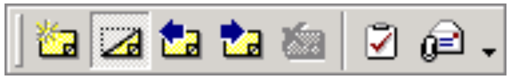

Withdrawals

X.25 connection discarded due to network congestion. Local limits now in effect


Withdrawals

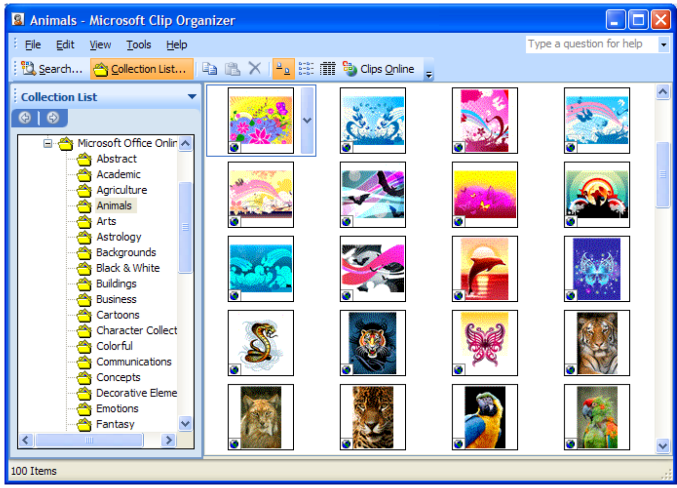
Maximum withdrawal of $50 at this time

- Use meaningful mnemonics, icons and abbreviations

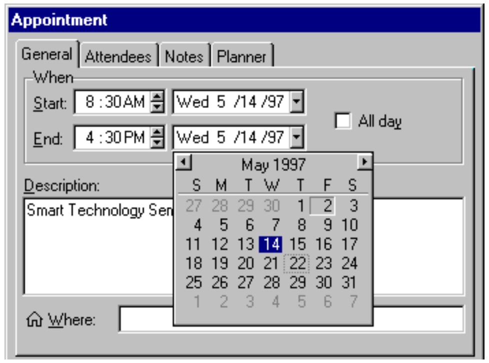



Minimize user memory load

- Recognition rather than recall
  - Recognize things previously experienced
  - Recall the things from memory
- Promote recognition over recall
  - Recognition is easier than recall
  - Recognition memory is much easier to develop
    - *Through simple exposure*
  - Make things visible
  - Familiar option is selected over unfamiliar options
  - Combo>textbox



- Describe expected input clearly
  - Don't allow for incorrect input
    - *Enter date (DD-Mmm-YY, e.g., 2-AUG-93):*
    - *Left margin:_____ points [0 – 128]*



Start: 8 :30AM Wed 5 /14 /97

End: 4 :30PM Wed 5 /14 /97

☐ All day

Description: Smart Technology Sen

May 1997

S	M	T	W	T	F	S
27	28	29	30	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
1	2	3	4	5	6	7

Where:

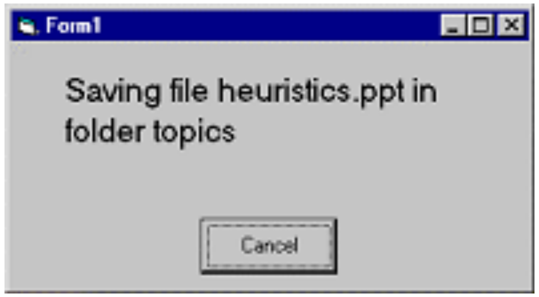
- Create orthogonal command systems
  - Use a **small** number of pervasive rules throughout the UI
  - Using generic commands that can be applied to all interface objects
  - Open, Save, Cut, Copy, Paste

Consistency

- Consistency and standards
- Be consistent in
  - Command design
    - *Same action, same effect in equivalent situations*
  - Graphic design
    - *Input format*
    - *Output format*
  - Flow design
    - *Similar tasks are handled in similar ways*
- Least surprise to users
  - Similar things should mean the similar/same thing
  - Different things should mean different things
- Consistency promotes skills acquisition and/or transfer

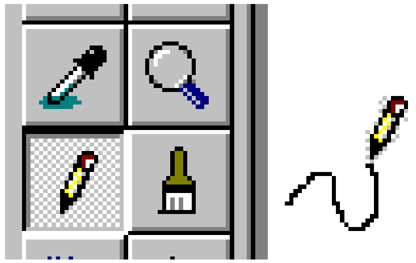
Feedback (Semantic)

- Visibility of system status
- Users should always be aware of what is going on
  - So that they can make informed decision
    - *Be specific*



Saving file heuristics.ppt in folder topics

  - But do not overburden users!
  - Provide redundant information



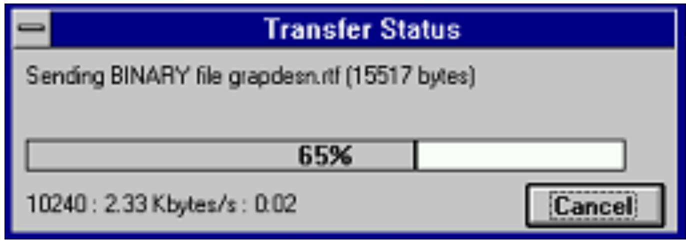
Feedback: Toolbar, cursor, ink

Feedback (Time)

- Different feedback time scales
  - Shall I wait for that task to finish or go for coffee?

> 10s	User will switch to another task while waiting
10s	Difficult to stay focused
1s	Delay but user's flow of thought is uninterrupted
.1s	Causality

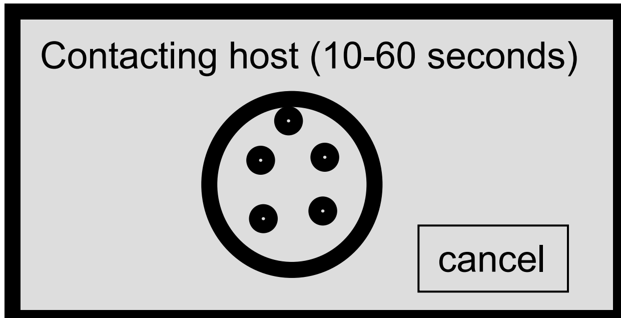
- Different techniques
  - Short transaction: hour glass cursor
  - Longer transaction: estimate of time left
    - *An overestimate is always better!*


Transfer Status

Sending BINARY file grapdesn.rtf (15517 bytes)

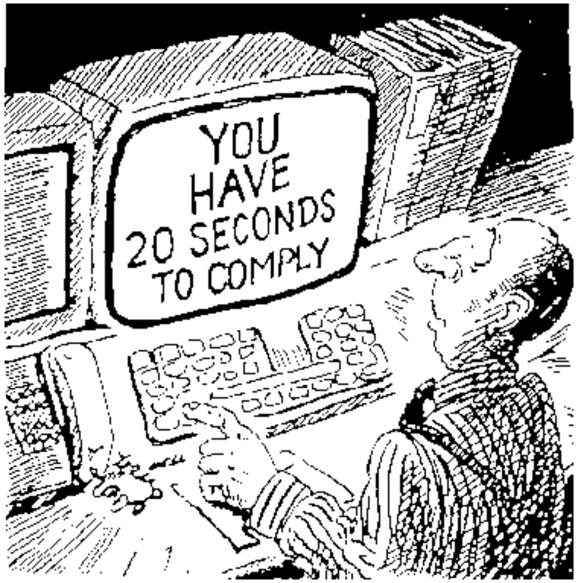
65%

10240 : 2.33 Kbytes/s : 0:02


Contacting host (10-60 seconds)

Clearly marked exits

- User control and freedom
- Users don’t like to be trapped!



- Strategies
  - Cancel button (or Esc key) for dialog
    - *Make the cancel button responsive!*
  - Universal undo/Redo

Shortcuts

- Flexibility and efficiency of use
- Expert users should be able to perform operations rapidly
  - Try to limit the training necessary to access advanced features

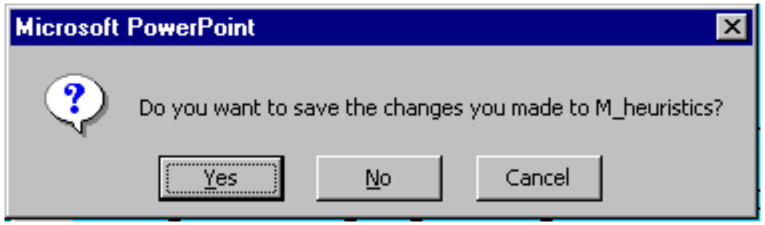
- Strategies
  - Keyboard and mouse accelerators (shortcuts)
    - *menu shortcuts and function keys*
    - *command completion, command abbreviations and type-ahead*
  - Toolbars and tool palettes
    - *Trade screen real estate for rapid access*
  - Navigation jumps
    - *History systems*
      - (command-line system) 35% of all commands are identical to one of the five previous commands, and 75% of the commands had been issued at least once before
  - Allow users to tailor frequent actions
    - *macros*

Preventing errors

- Error prevention
- Error types: slips and mistakes
  - Mistakes
    - *Conscious decision with unforeseen consequences*
    - *When users don’t know the system well, they cannot formulate the right goal*
    - *Magnifying glass: for find or for magnifying texts?*
    - *Radical redesign or improved training*
  - Slips
    - *Users know what and how to do, but fail to do it correctly*
    - *Capture errors*
    - *Mode errors*
    - *Loss of activation*
    - …

Types of slips

- Capture error
  - frequently done activity takes charge instead of one intended
  - occurs when common & rarer actions have same initial sequence
    - *change clothes for dinner and find oneself in bed* (William James, 1890)
    - *1,2,3,4,5,6,7,8,9,10,J,Q,K*
  - minimize by
    - *make actions undoable instead of confirmation*
    - *allows reconsideration of action by user*
      - e.g. open trash to undelete a file


Microsoft PowerPoint

Do you want to save the changes you made to M_heuristics?

Types of slips

- Description error
  - intended action similar to others that are possible
    - *usually occurs when right & wrong objects* ***physically near*** *each other*
      - pour juice into bowl instead of glass
      - throw sweaty shirt in toilet instead of laundry basket
      - move file to wrong folder with similar name
  - minimize by
    - *rich feedback*
    - *check for reasonable input, etc.*
    - *undo*

Types of slips

- Loss of activation
  - forget what the goal is while undergoing the sequence of actions
    - *start going to room and forget why you are going there*
    - *navigating menus/dialogs & can′t remember what you are looking for*
    - *but continue action to remember (or go back to beginning)!*
  - minimize by
    - *if system knows goal, make it explicit*
    - *if not, allow person to see path taken*

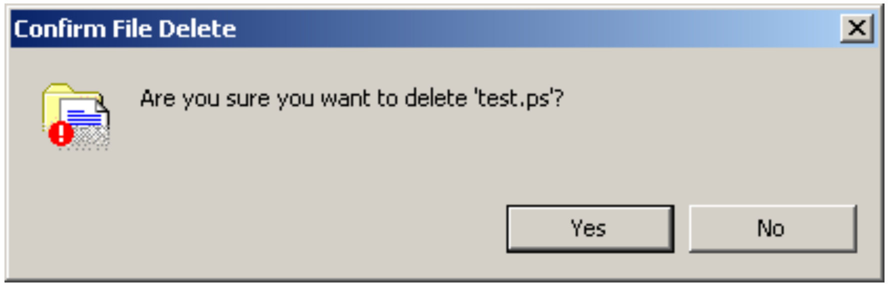
Types of slips

- Mode errors
  - people do actions in one mode thinking they are in another
    - *refer to file that's in a different directory*
    - *look for commands / menu options that are not relevant*
  - minimize by
    - *have as few modes as possible (preferably none)*
    - *make modes highly visible*

Designing for slips

An ounce of prevention is worth more than a pound of cure!

- Examples
  - Design modeless interfaces
  - Instead of confirmations, provide undo mechanisms



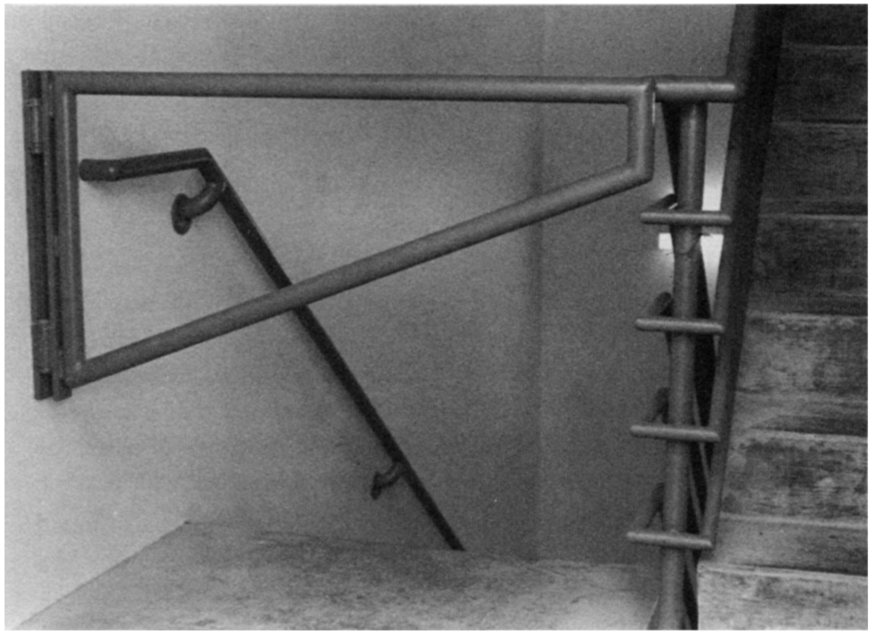
Confirm File Delete

Are you sure you want to delete 'test.ps'?

  - Check for reasonable input
    - *Be prepared to handle several formats*
    - *Make entering a incorrect format impossible*
  - Make the current goal clear
    - *Prevent lost of activations*

Forcing functions

- Interlock mechanisms
  - Require step A before step B can be performed
  - Ex: Switching from P to D in a car requires pressing brake pedal
- Lockin mechanisms
  - Process continues unless user removes constraint before stopping it
  - Ex: No eject button for CD drive on Mac
- Lockout mechanisms
  - Process won’t occur unless user removes constraint before starting it
  - Ex: Basement stairway



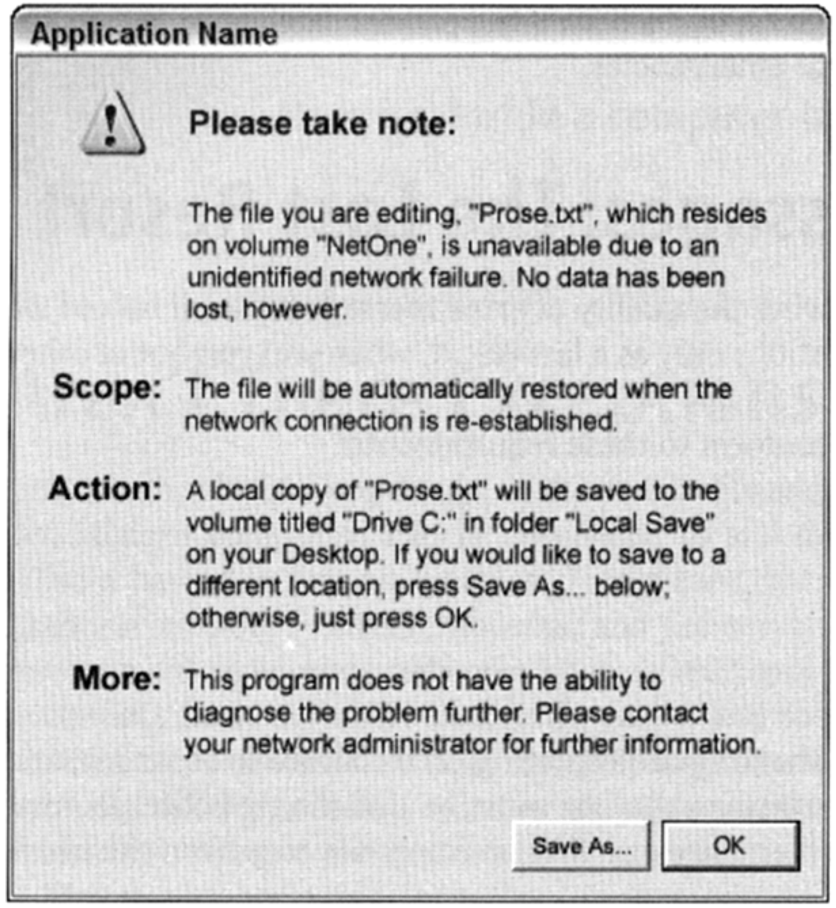
Dealing with errors

- People will make errors!
  - You can ignore them
    - *Generally very confusing*
  - You can correct them automatically (DWIM – Do What I Mean)
    - *Spelling corrector*
    - *But is the system right 100% of the time?*
  - You can discuss it
    - *But novice/expert tradeoff*
  - You can try to teach the user what to do
    - *Office assistant*

- Respect users feelings!
  - The user is never wrong

Good error messages

- Be phrased in clear language, avoid obscure codes
- Be precise rather than vague or general
  - *Cannot open "Chapter 5" because the application is not on the disk*
- Constructively help the user solve the problem
  - *Real app name than "the application"*
- Be polite, and not intimidate the user (don't put the blame on users)
  - *ILLEGAL USER ACTION, JOB ABORTED*


Application Name

Please take note:

The file you are editing, "Prose.txt", which resides on volume "NetOne", is unavailable due to an unidentified network failure. No data has been lost, however.

Scope: The file will be automatically restored when the network connection is re-established.

Action: A local copy of "Prose.txt" will be saved to the volume titled "Drive C:" in folder "Local Save" on your Desktop. If you would like to save to a different location, press Save As... below; otherwise, just press OK.

More: This program does not have the ability to diagnose the problem further. Please contact your network administrator for further information.

From Cooper's "About Face 2.0"

Good error messages

- Provide meaningful error messages
  - Explain the problem in terms of the user conceptual model
  - Don’t make the user feel stupid
  - Offer a way to correct the problem

  - Compare
    - *Error 25: access denied*
    - *Cannot open “chapter 5” because “Microsoft Word” is not installed. Do you want to use Notepad instead?*

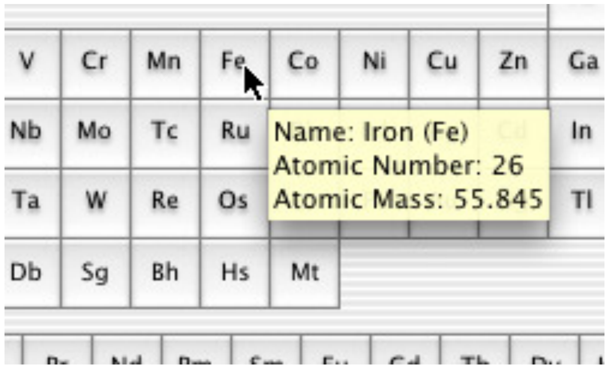
Provide help and documentation

- Providing help is not an excuse for poor design!
  - Note no need of docs always better
  - Saving a couple of line of code or writing several pages of documentation?
  - Users don’t like to read manuals
    - *They prefer to learn while making progress toward their goals*

- Most users will stay at the intermediate level
  - Need reminders and a clear learning path
  - Need a quick way to access critical information
    - *Online documentation and good search tool*

Types of help (I)

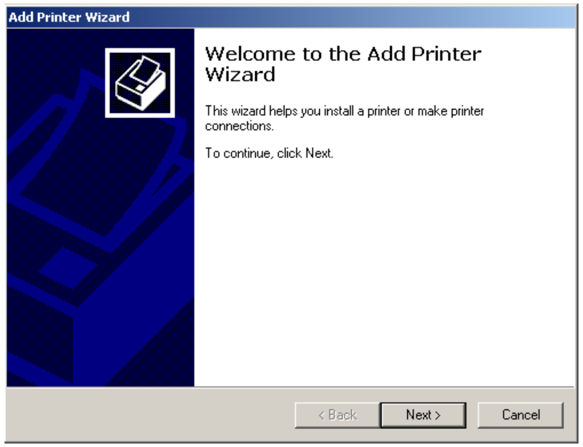
- Tutorial and/or getting started manuals
  - Presents the system conceptual model
    - *Basis for successful explorations*
  - Provides on-line tours and demos
    - *Demonstrates basic features*
- Reference manuals
  - Designed with experts in mind
- Reminders
  - Short reference cards, keyboard templates, tooltips…
- "Show me" videos



Name: Iron (Fe)
Atomic Number: 26
Atomic Mass: 55.845

Types of help (II)

- Wizards
  - Walks user through typical tasks
    - *Users feel they are losing control*
    - *What if I do not have the information requested?*

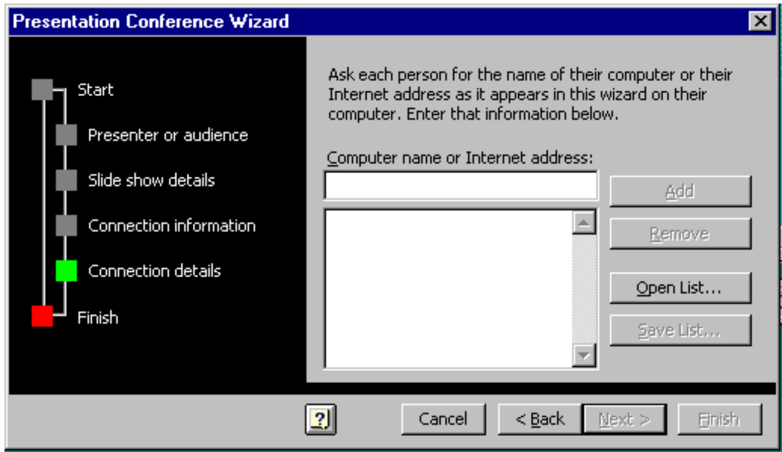


Welcome to the Add Printer Wizard

This wizard helps you install a printer or make printer connections.

To continue, click Next.

- Tips
  - Migration path to learning new features
  - Can become boring and tedious
  - *Can be* dangerous if user gets stuck

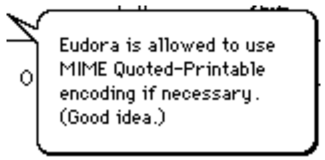


Ask each person for the name of their computer or their Internet address as it appears in this wizard on their computer. Enter that information below.

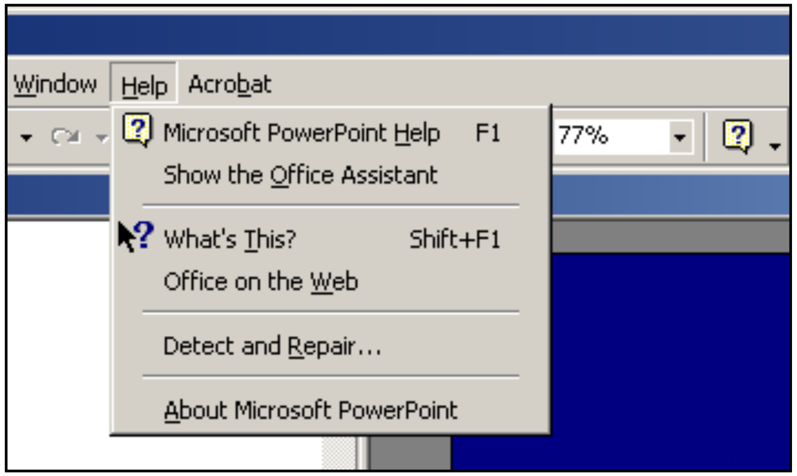
Computer name or Internet address:

Types of help (II)

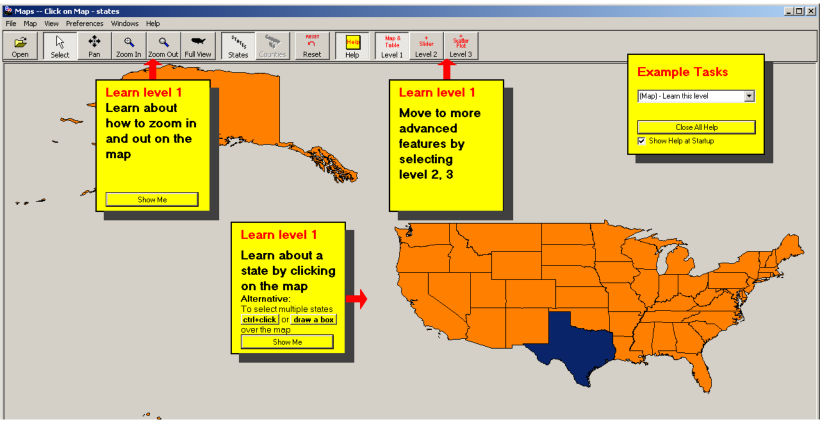
- Context sensitive help
  - System provides help on the interface component the user is currently working with
    - *Macintosh "balloon help"*
    - *Microsoft "What's this" help*



Eudora is allowed to use MIME Quoted-Printable encoding if necessary. (Good idea.)



Window
Help
Acrobat
Microsoft PowerPoint Help F1
Show the Office Assistant
What's This? Shift+F1
Office on the Web
Detect and Repair...
About Microsoft PowerPoint
77%

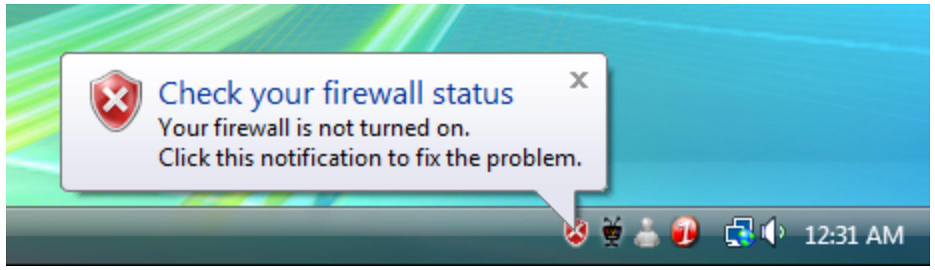


Learn level 1
Learn about how to zoom in and out on the map

Learn level 1
Move to more advanced features by selecting level 2, 3

Example Tasks
(Map) - Learn this level
Show Help at Startup

Learn level 1
Learn about a state by clicking on the map
Alternative:
To select multiple states ctrl+click or draw a box over the map



Check your firewall status
Your firewall is not turned on.
Click this notification to fix the problem.

Nielsen's heuristics :
10 general principles for user interface design

- Simple and natural dialog
- Speak the users' language
- Minimize user memory load
- Consistency
- Feedback
- Clearly marked exits
- Shortcuts
- Prevent errors
- Good error messages
- Provide help and documentation

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

Questions?

- Volunteers for Hall of Fame/Shame?