Design Principles & Heuristic Evaluation

Nielson's 10 principles for user interface design

Heuristic evaluation

This section contains slides adapted from Saul Greenberg's slide deck

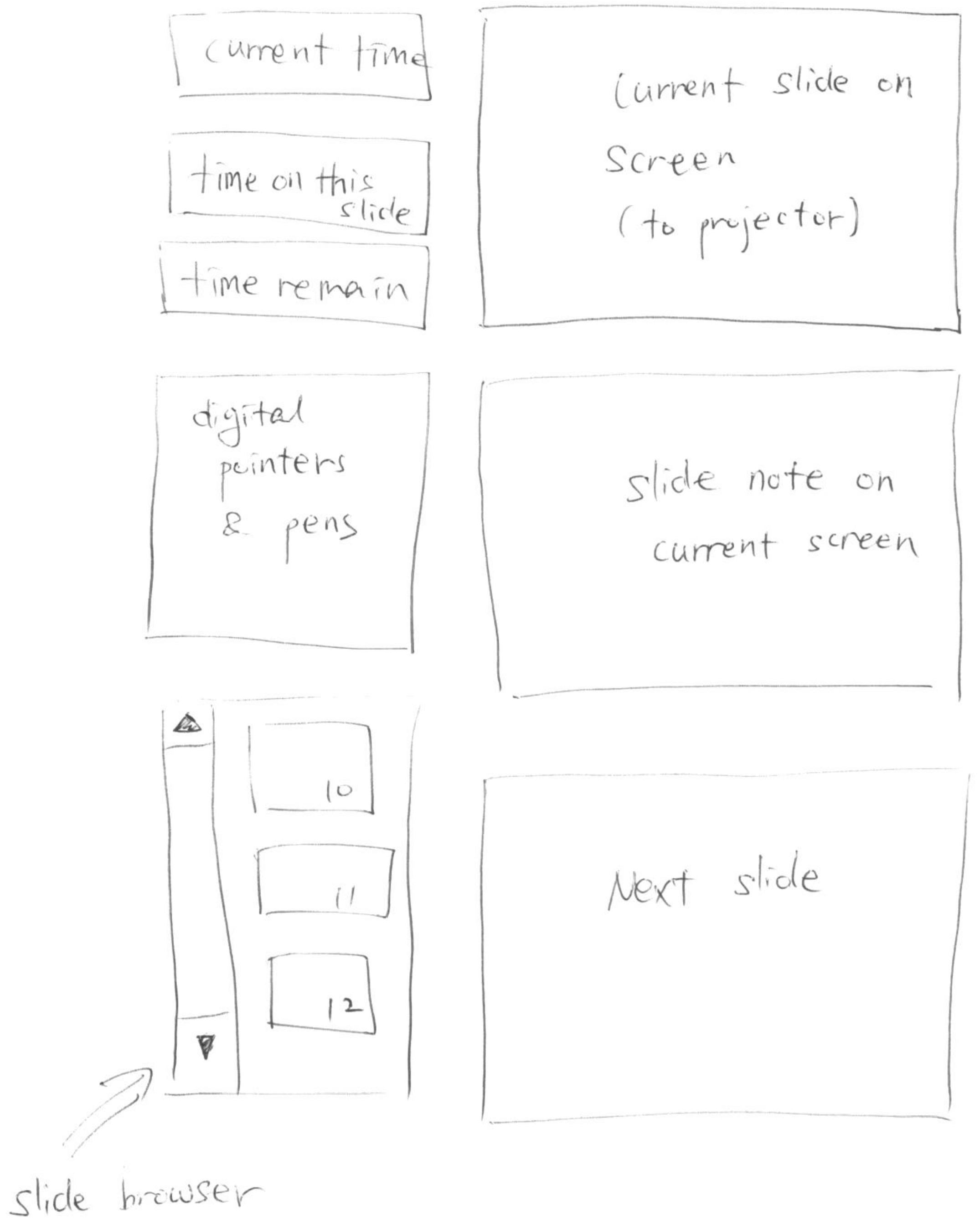
Hall of Fame or Shame

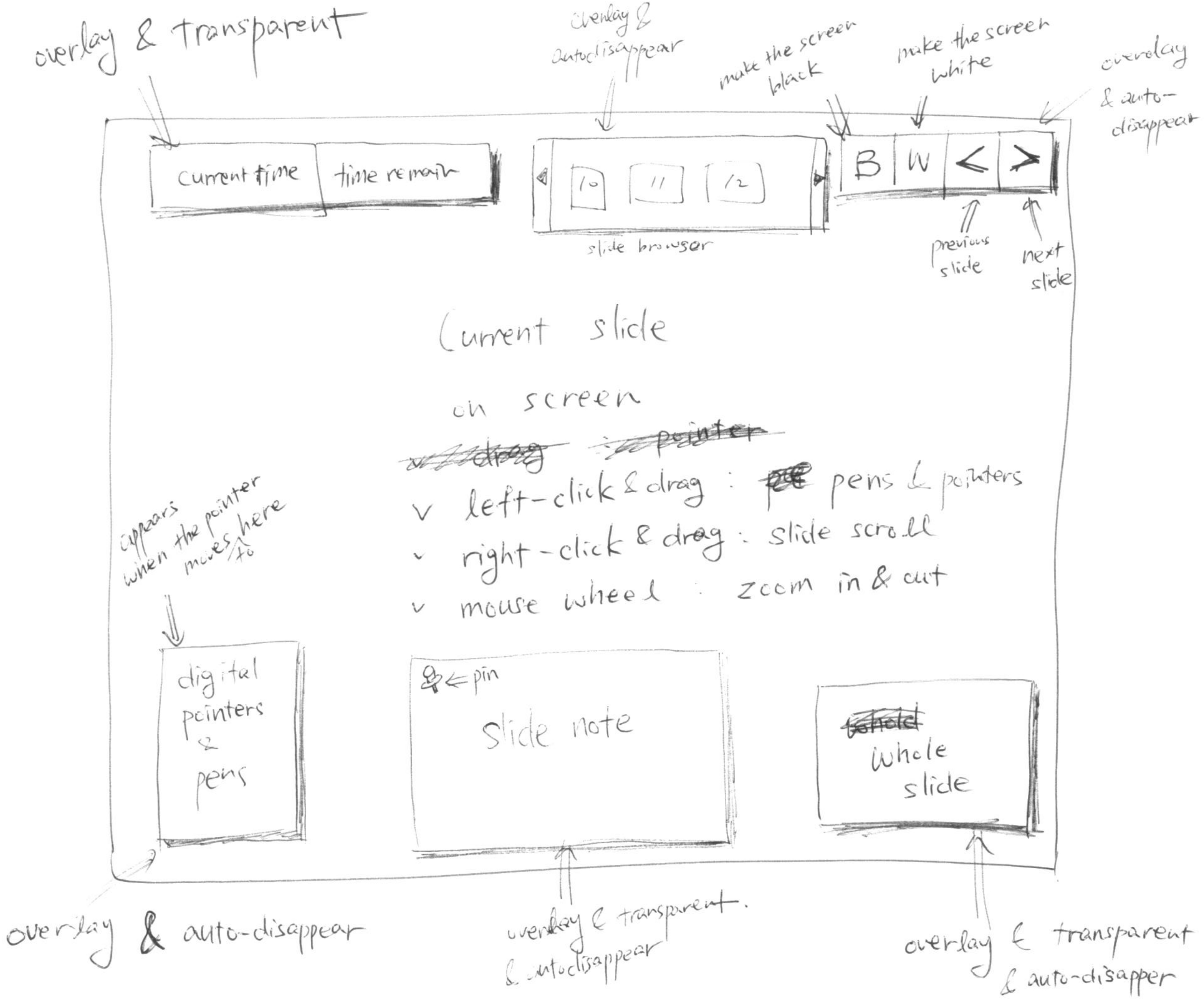
File – Print

Printer button on toolbar

Ctrl+P

Homework Pres




overlay & transparent
overlay & autodisappear
make the screen black
make the screen white
overlay & auto-disappear
current time
time remain
10
11
12
slide browser
B
W
<
>
previous slide
next slide
Current slide
on screen
~~drag~~ : ~~pointer~~
v left-click & drag : ~~pe~~ pens & pointers
v right-click & drag : slide scroll
v mouse wheel : zoom in & out
appears when the pointer moves here/to
digital pointers & pens
pin
slide note
~~whold~~
whole slide
overlay & auto-disappear
overlay & transparent. & autodisappear
overlay & transparent & auto-disapper

General UI Design Principles

"heuristics" – rule of thumbs

Plenty of choices

Help designers choose from design alternatives

Help find usability problems

→ "heuristic evaluation"

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

1. Visibility of System Status

keep users informed of system state

- Use status bar to show status/progress
- Make selection explicit
- Use different cursor for different mode

provide instantaneous informative feedback

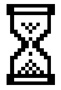
- Perceptual cycle time: 100ms
- 0.1-1s: delay noticed but no feedback necessary
- 1-5s: show hourglass cursor
- >5s: ?
- 10s:limit for keeping user’s attention focused on the dialog
- >10s:user will want to perform other tasks while waiting

1. Visibility of System Status

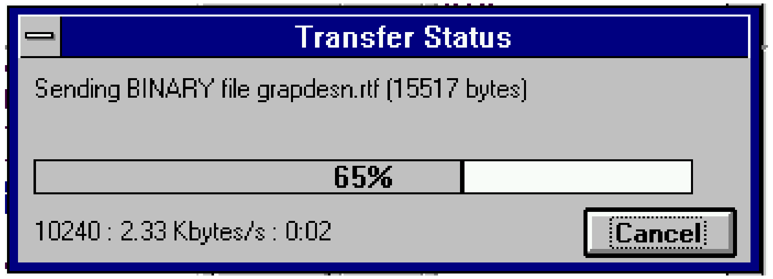
Dealing with long delays

Cursors

- for short transactions



- Percent done dialogs
  - time left
  - estimated time

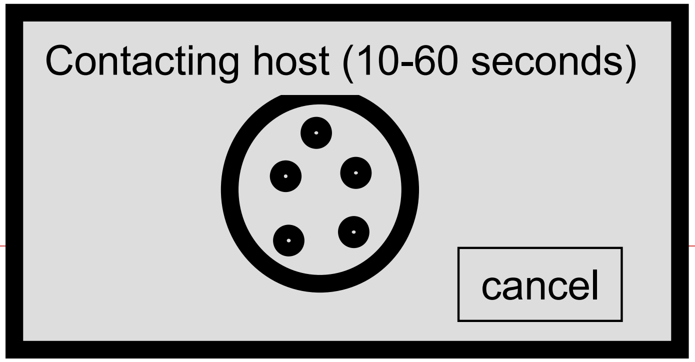


Sending BINARY file grapdesn.rtf (15517 bytes)

65%

10240 : 2.33 Kbytes/s : 0:02

- Random
  - for unknown times

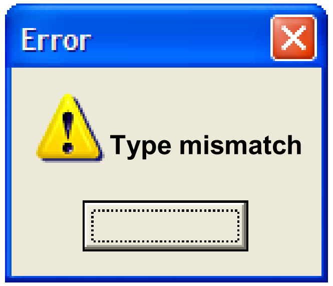


Contacting host (10-60 seconds)

2. Match the Real World

The system should speak the users' language

Avoid technical jargon



Type mismatch

Follow real-world conventions

- alias, synonyms

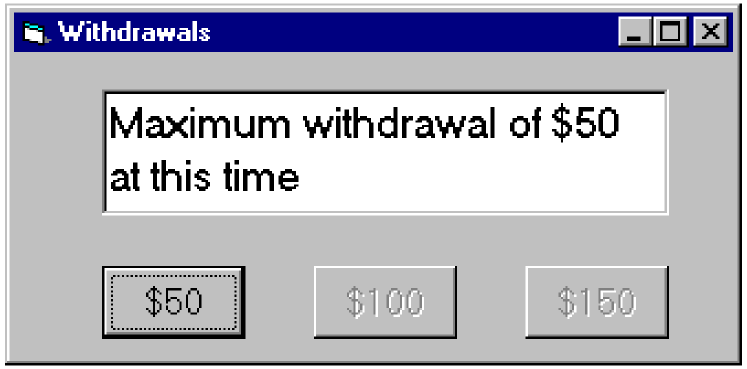
Use metaphors appropriately

Avoid non-natural limits on length or contents

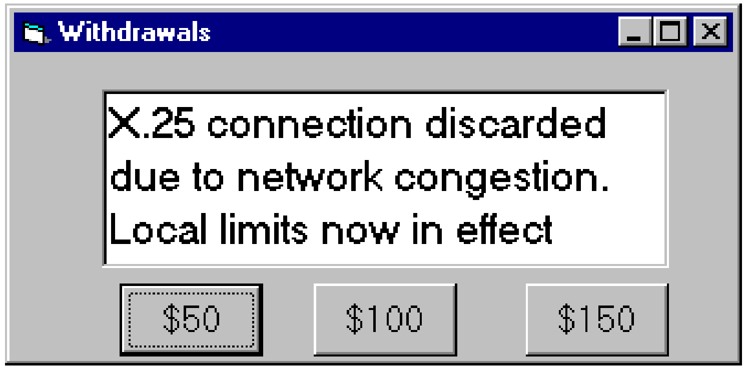
2. Match the Real World

Terminology based on users' language for task

- e.g. withdrawing money from a bank machine

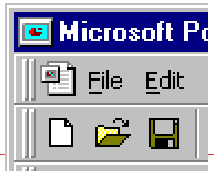


Maximum withdrawal of $50 at this time



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

Use meaningful mnemonics, icons & abbreviations

- eg File / Save
  - Ctrl + S (abbreviation)
  - Alt FS (mnemonic for menu action)
  -  (tooltip icon)

3. User control and freedom

Provide a clearly marked exit

- Undo/Redo
- Allow users to cancel
  - Long jobs or any dialogs

Put users in charge of interface

4. Consistency and standards

Least surprise to users

- Similar things should mean the similar/same thing
- Different things should mean different things

Follow platform conventions

- Argument order: prefix/postfix

Internal/External/Metaphorical consistency

5. Error prevention

Much better than good error messages

Eliminate error-prone conditions

- Use constrains
- Disable options/commands properly
- Don’t go too far

Present users with a confirmation option before commitment

5. Error prevention

Consider type of error (slip)

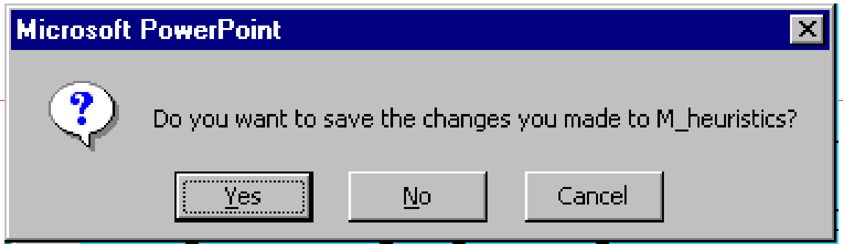
- Capture errors
- **Description errors**
- Data-driven errors
- Associative activation errors
- Loss-of-activation errors
- **Mode errors**

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)
  - confirm saving of a file when you don't want to delete it

- 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

6. Recognition rather than recall

Minimize the user’s working memory load

- Make things visible
- Combo>textbox
- Open, Save, Cut, Copy, Paste

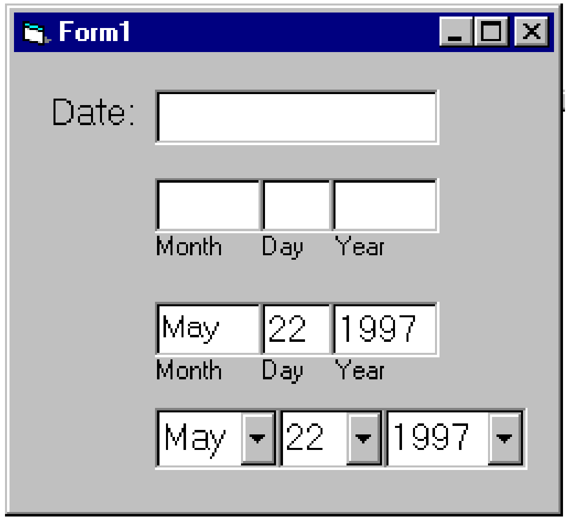
Perception vs. cognition

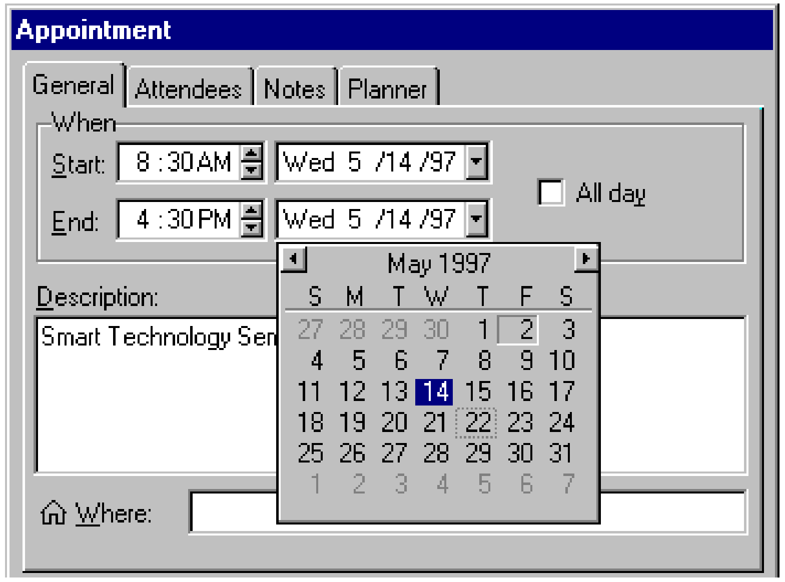
Working vs. long-term memory

Knowledge in the world vs.
 Knowledge in the head

6. Recognition rather than recall

Gives input format, example and default





7. Flexibility and efficiency of use

Accelerators or shortcuts

- For expert users
- For frequently used action/object

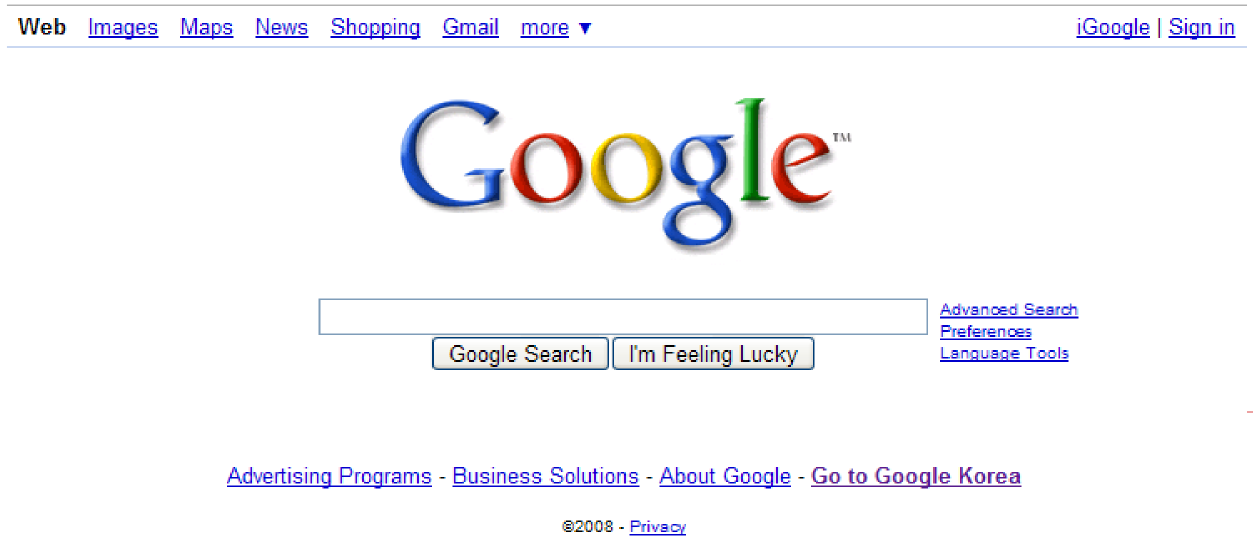
Allow users to tailor frequent actions

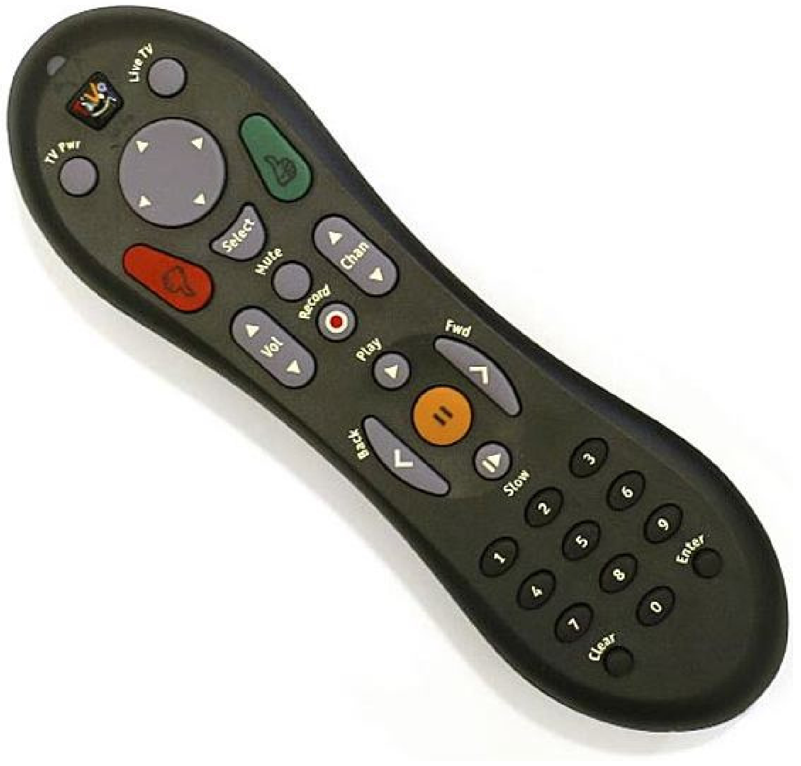
- macros

8. Aesthetic and minimalist design

Mute/hide irrelevant or rarely needed info

Give maximum visibility to the relevant





9. Help users recognize, diagnose and recover from error

Use plain language (no codes) in error messages

Precise indication of the error

- "Cannot save file" vs.
  "Cannot save file lecture.ppt"
- But hide tech details until requested

Give constructive suggestion/help

- Explain the cause of error
- Don't blame users

Generic system responses for errors

General idea: Forcing functions

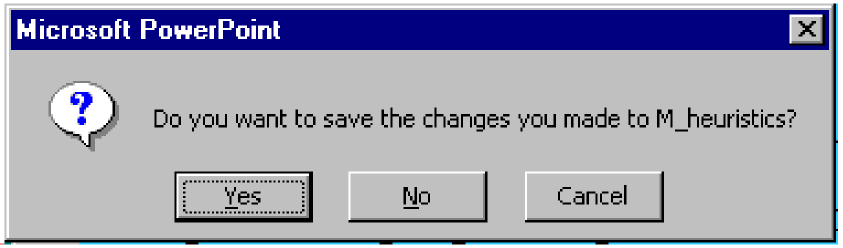
- prevent / mitigate continuation of wrongful action

Gag

- deals with errors by preventing the user from continuing
  - eg cannot get past login screen until correct password entered

Warn

- warn people that an unusual situation is occurring
- when overused, becomes an irritant
  - e.g.,
    - audible bell
    - alert box


Microsoft PowerPoint

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

Generic system responses for errors

Do nothing

- illegal action just doesn’t do anything
- user must infer what happened
  - enter letter into a numeric-only field (key clicks ignored)
  - put a file icon on top of another file icon (returns it to original position)

Self-correct

- system guesses legal action and does it instead
- but leads to a problem of trust
  - spelling corrector

Generic system responses for errors

Lets talk about it

- system initiates dialog with user to come up with solution to the problem
  - compile error brings up offending line in source code

Teach me

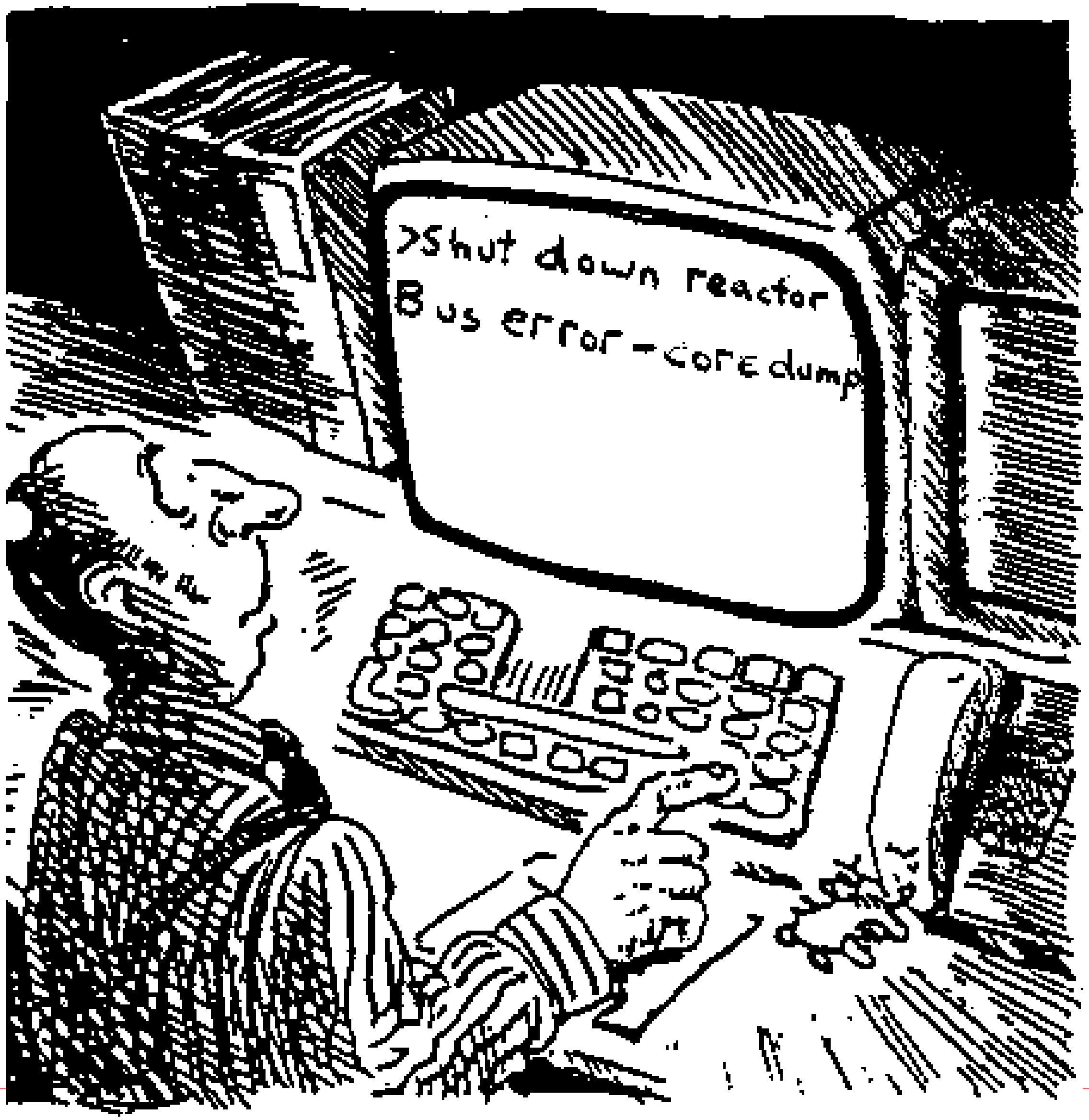
- system asks user what the action was supposed to have meant
- action then becomes a legal one

Meaningful error messages

DO IT
ERROR 15762

What is “error 15762”?

Meaningful error messages



A problematic message to a nuclear power plant operator

Meaningful error messages

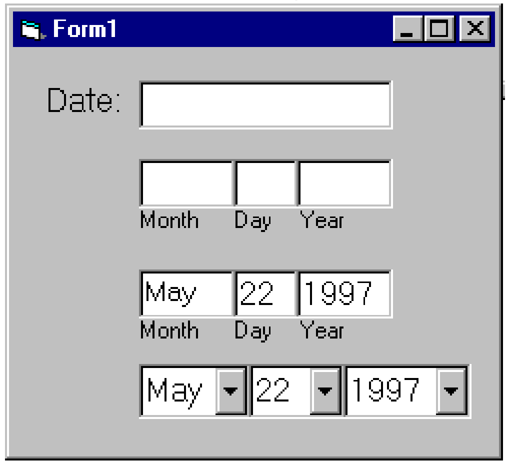
don’t make people feel stupid

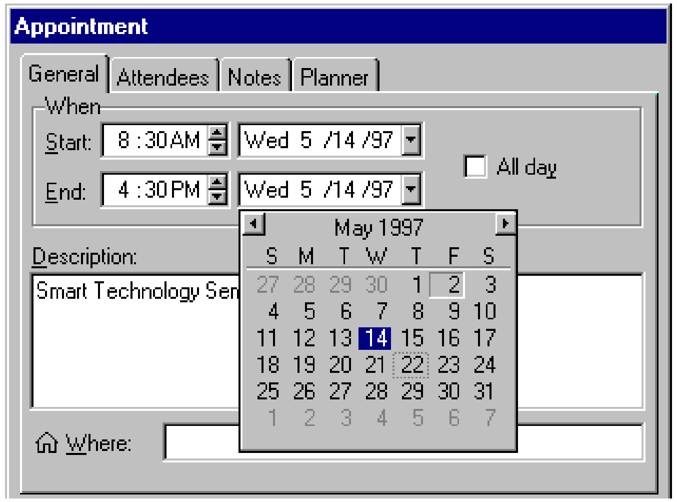
- Try again, bonehead!
- Error 25
- Cannot open this document
- Cannot open “chapter 5” because the application “Microsoft Word” is not on your system
- Cannot open “chapter 5” because the application “Microsoft Word” is not on your system. Open it with “Teachtext” instead?

Check Inputs

Prevent errors

- try to make errors impossible
- modern widgets: can only enter legal data





Provide reasonableness checks on input data

- on entering order for office supplies
  - 5000 pencils is an unusually large order. Do you really want to order that many?

10. Help and documentation

Easy to search

Focused on the user’s task

List concrete steps to follow

Not be too large

Note no need of docs always better

Documentation and how it is used

Many users do not read manuals

- prefer to spend their time pursuing their task

Usually used when users are in some kind of panic

- paper manuals unavailable in many businesses!
  - e.g. single copy locked away in system administrator’s office
- Indicates need for online documentation, good search/lookup tools
- online help can be specific to current context

Sometimes used for quick reference

- syntax of actions, possibilities...
- list of shortcuts ...

Types of help

Tutorial and/or getting started manuals

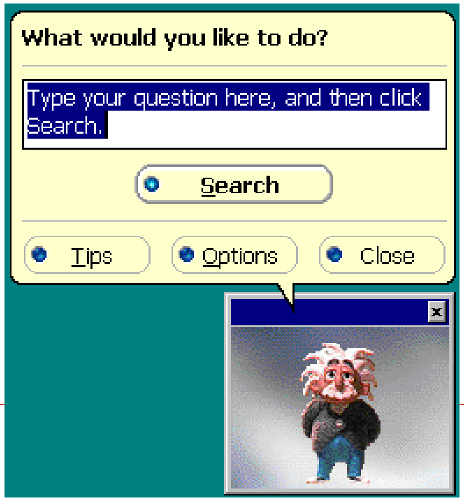
- short guides that people are likely to read when first obtaining their systems
  - encourages exploration and getting to know the system
  - tries to get conceptual material across and essential syntax

- on-line “tours”, exercises, and demos
  - demonstrates very basic principles through working examples

Types of help

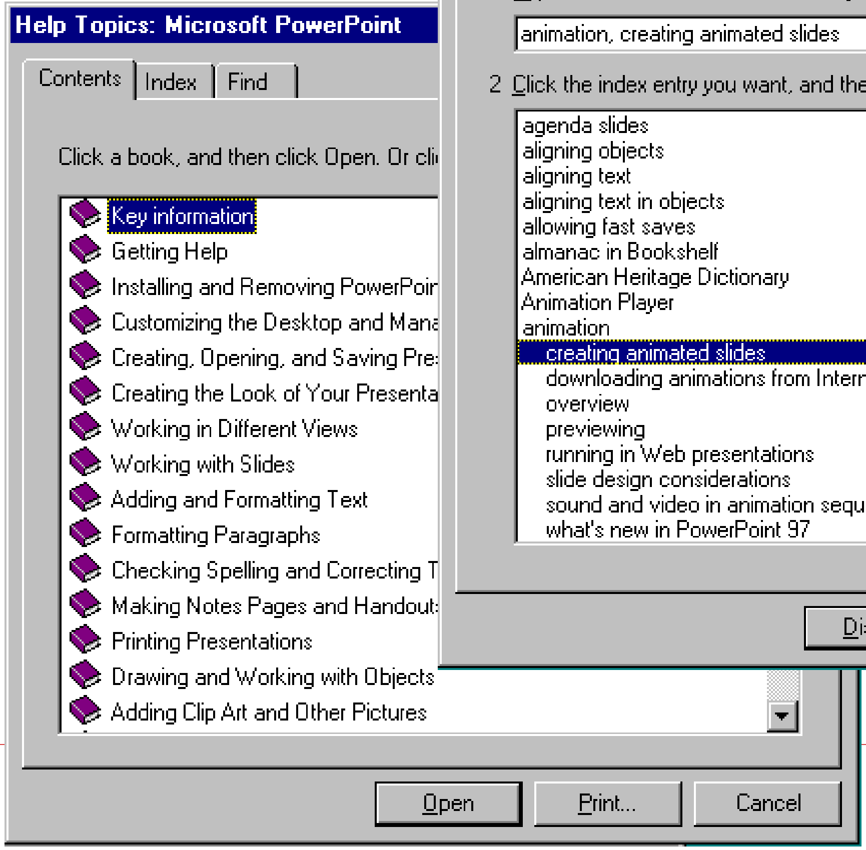
Reference manuals

- used mostly for detailed lookup by experts
  - rarely introduces concepts
  - thematically arranged
- on-line hypertext
  - search / find
  - table of contents
  - index
  - cross-index



What would you like to do?

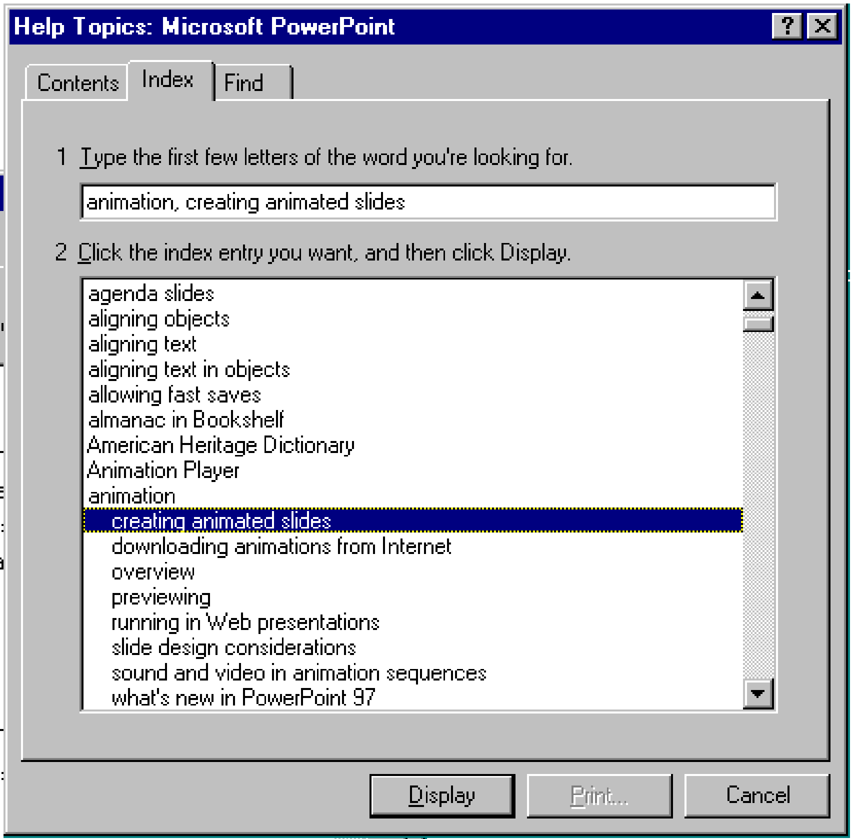
Type your question here, and then click Search.



Help Topics: Microsoft PowerPoint

Click a book, and then click Open. Or cli

- Key information
- Getting Help
- Installing and Removing PowerPoir
- Customizing the Desktop and Mana
- Creating, Opening, and Saving Pre
- Creating the Look of Your Presenta
- Working in Different Views
- Working with Slides
- Adding and Formatting Text
- Formatting Paragraphs
- Checking Spelling and Correcting T
- Making Notes Pages and Handout
- Printing Presentations
- Drawing and Working with Objects
- Adding Clip Art and Other Pictures



Help Topics: Microsoft PowerPoint

1 Type the first few letters of the word you're looking for.

animation, creating animated slides

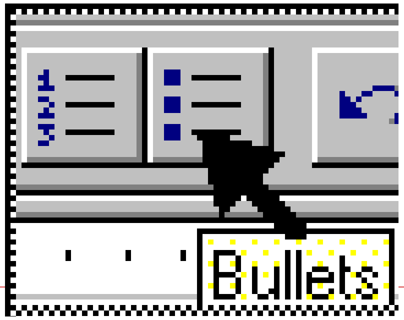
2 Click the index entry you want, and then click Display.

- agenda slides
- aligning objects
- aligning text
- aligning text in objects
- allowing fast saves
- almanac in Bookshelf
- American Heritage Dictionary
- Animation Player
- animation
  - creating animated slides
  - downloading animations from Internet
  - overview
  - previewing
  - running in Web presentations
  - slide design considerations
  - sound and video in animation sequences
  - what's new in PowerPoint 97

Types of help

Reminders

- short reference cards
  - expert user who just wants to check facts
  - novice who wants to get overview of system’s capabilities
- keyboard templates
  - shortcuts/syntactic meanings of keys; recognition vs. recall; capabilities
- tooltips and other context-sensitive help
  - text over graphical items indicates their meaning or purpose



Microsoft Word

Types of help

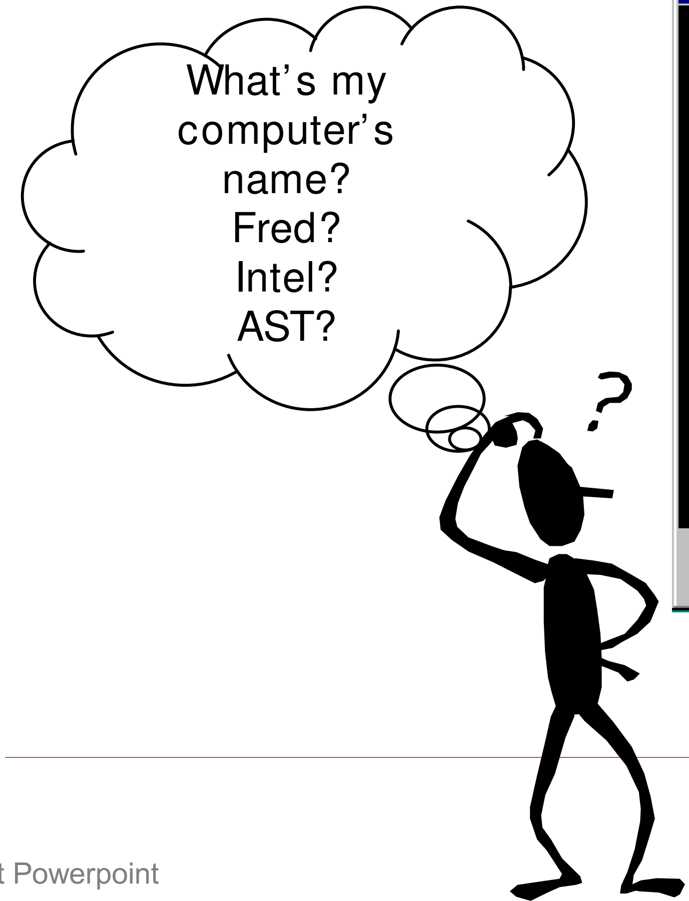
Context-sensitive help

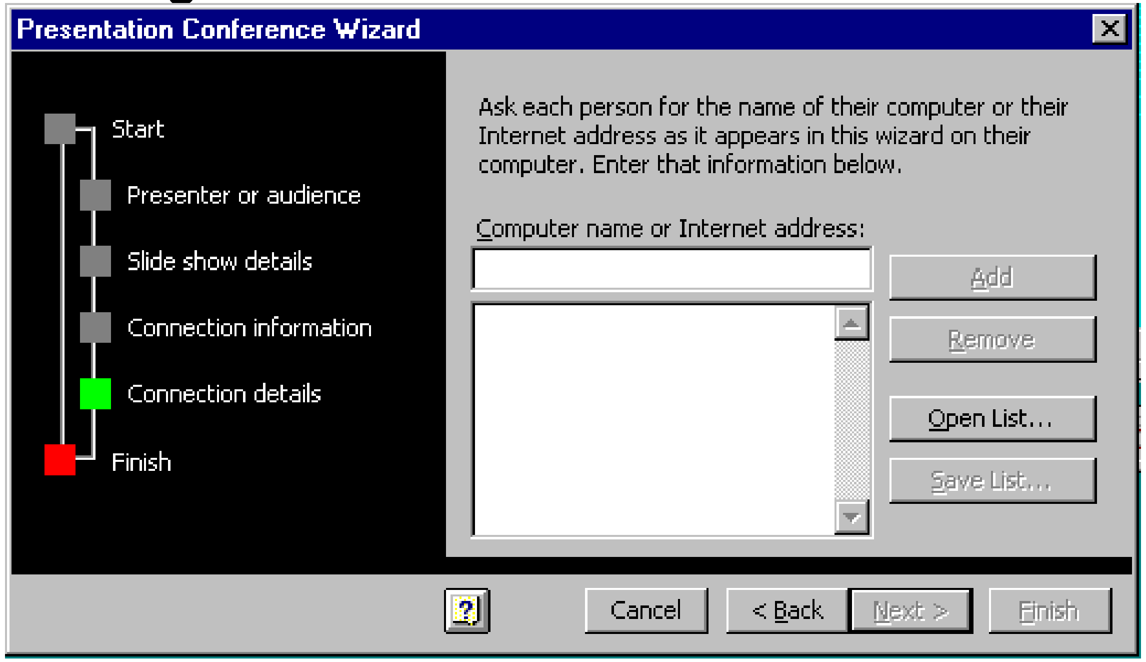
- System provides help on the interface component the user is currently working with
  - Macintosh “balloon help”
  - Microsoft “What’s this” help

Types of help

Wizards

- walks user through typical tasks
- *but* dangerous if user gets stuck





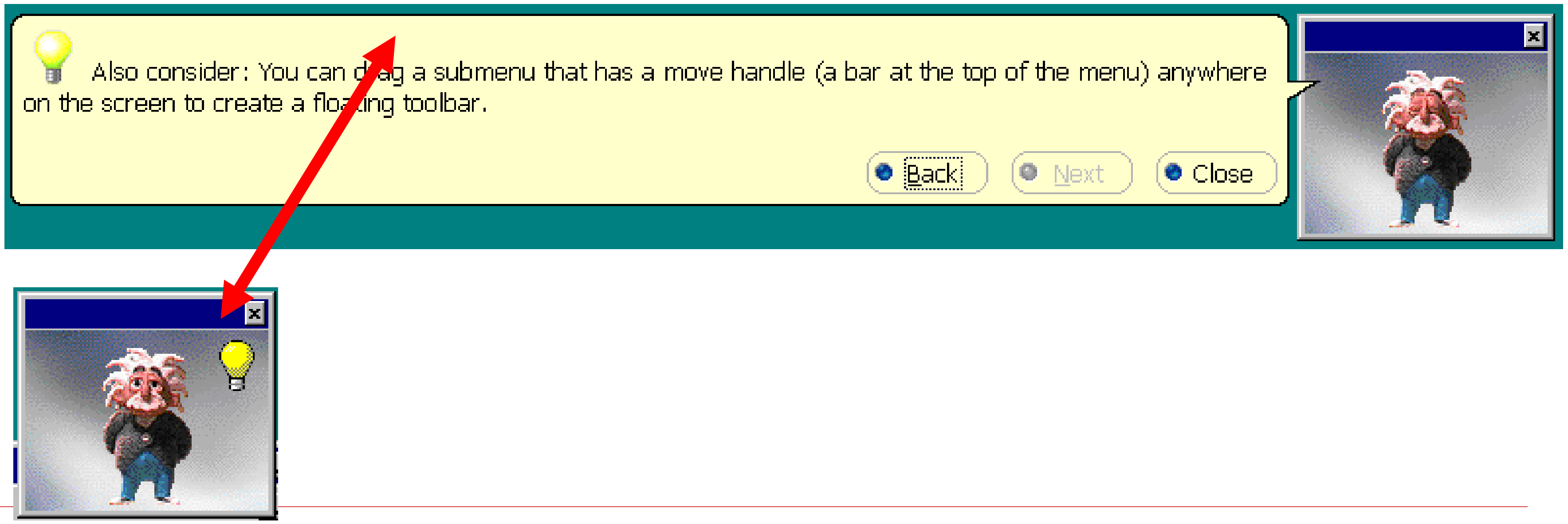
Ask each person for the name of their computer or their Internet address as it appears in this wizard on their computer. Enter that information below.

Computer name or Internet address:

Types of help

Tips

- migration path to learning system features
- also context-specific tips on being more efficient
- must be “smart”, otherwise boring and tedious


Also consider: You can drag a submenu that has a move handle (a bar at the top of the menu) anywhere on the screen to create a floating toolbar.

Microsoft Word

Heuristic Evaluation, Jakob Nielson

Usability engineering method for finding the usability problems in a user interface

A small set of evaluators examine UI (“expert review”)

- Independently check for compliance with heuristics (usability principles)
- Different evaluators will find different problems
- Evaluators only communicate afterwards
  → findings are then aggregated

Works for paper, prototypes, and working systems

Becoming popular

- No user involvement
- Catches many design flaws

Heuristic evaluation

Advantages

- the “minimalist” approach
  - a few guidelines identify many common usability problems
  - easily remembered, easily applied with modest effort

- discount usability engineering
  - end users not required
  - cheap and fast way to inspect a system
  - can be done by usability experts

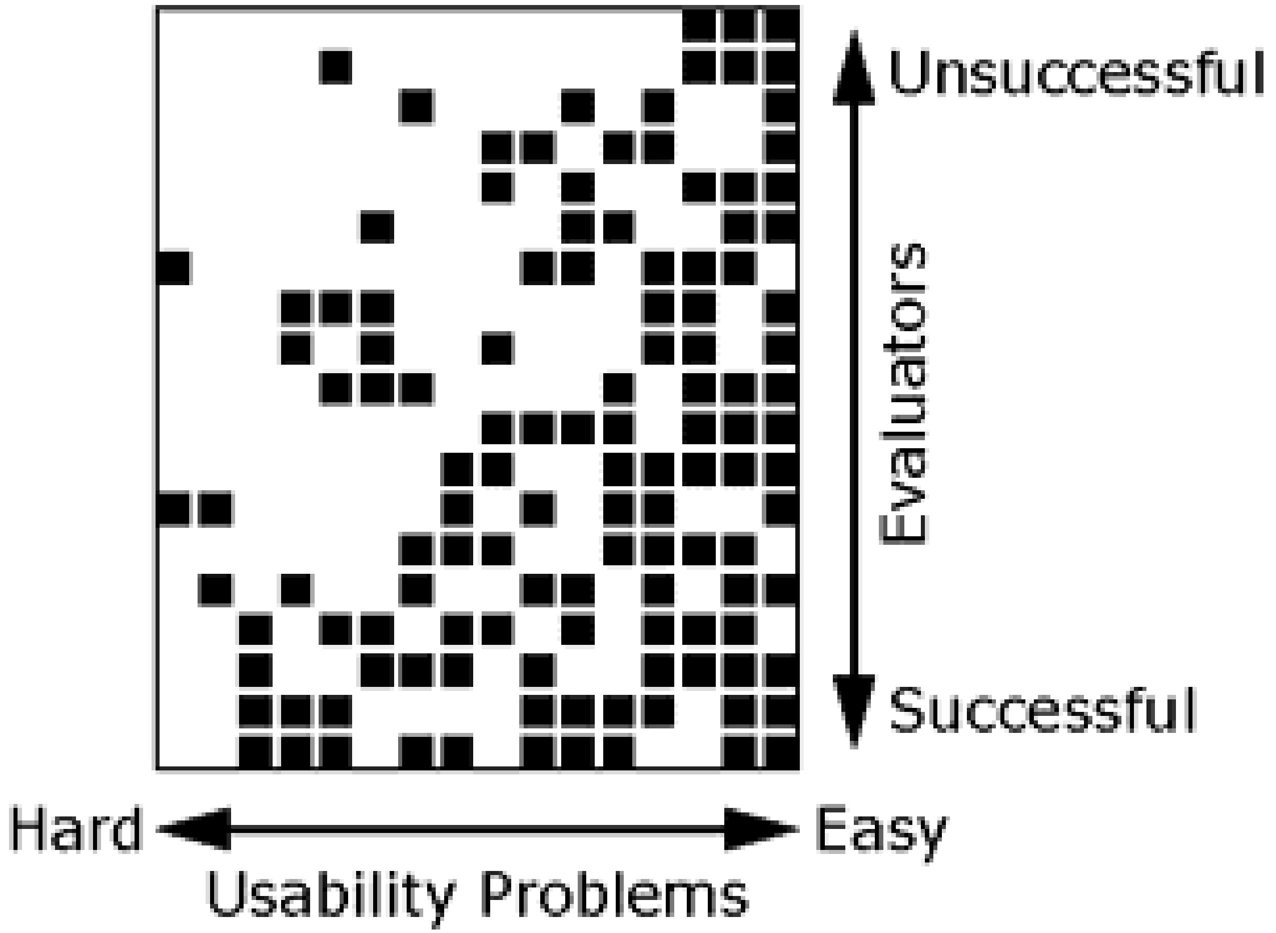
Disadvantages

- Principles can’t be treated as a simple checklist
- subtleties involved in their use

Problems found by a single inspector

Evaluators miss both easy and hard problems

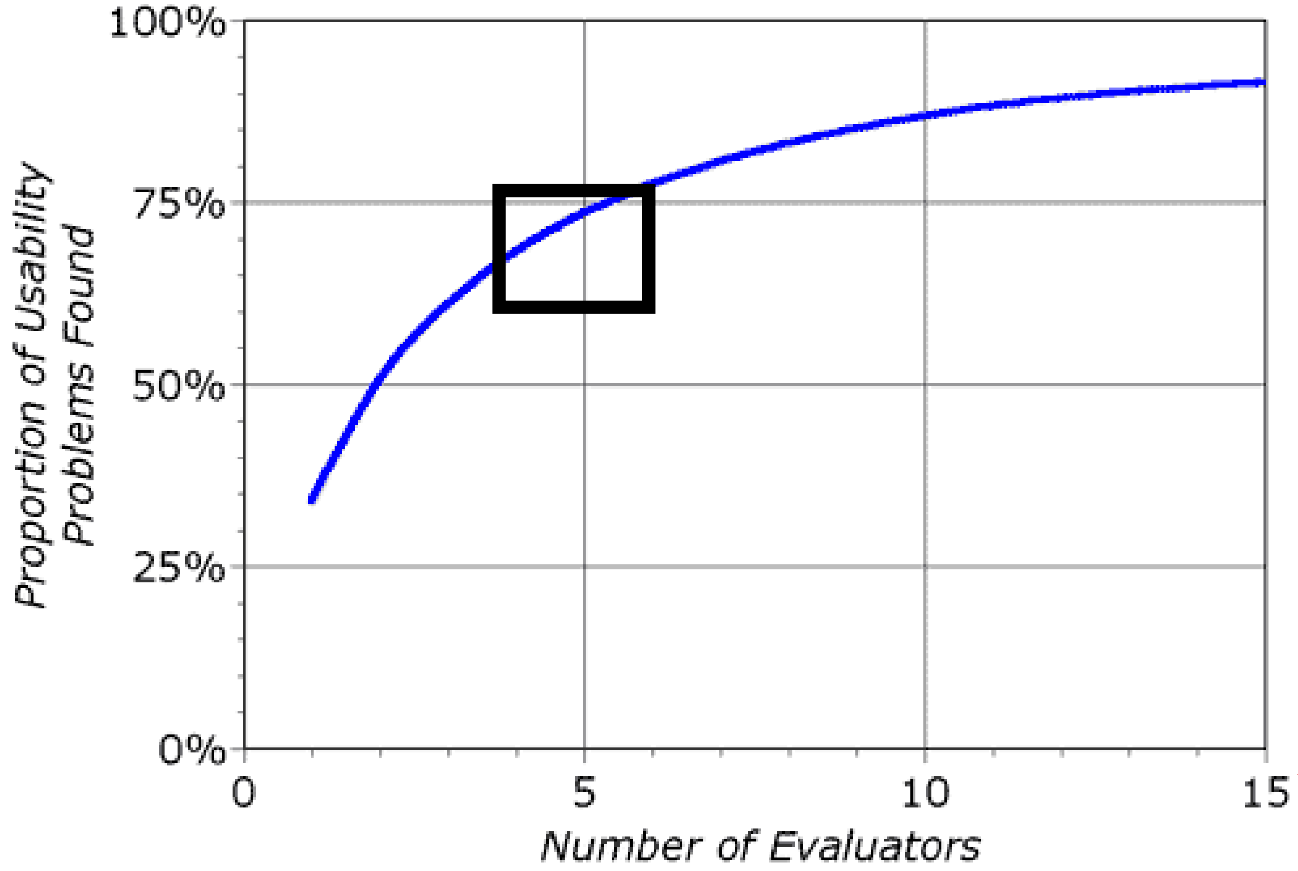
- 'best' evaluators can miss easy problems
- 'worse' evaluators can discover hard problems



Problems found by multiple evaluators

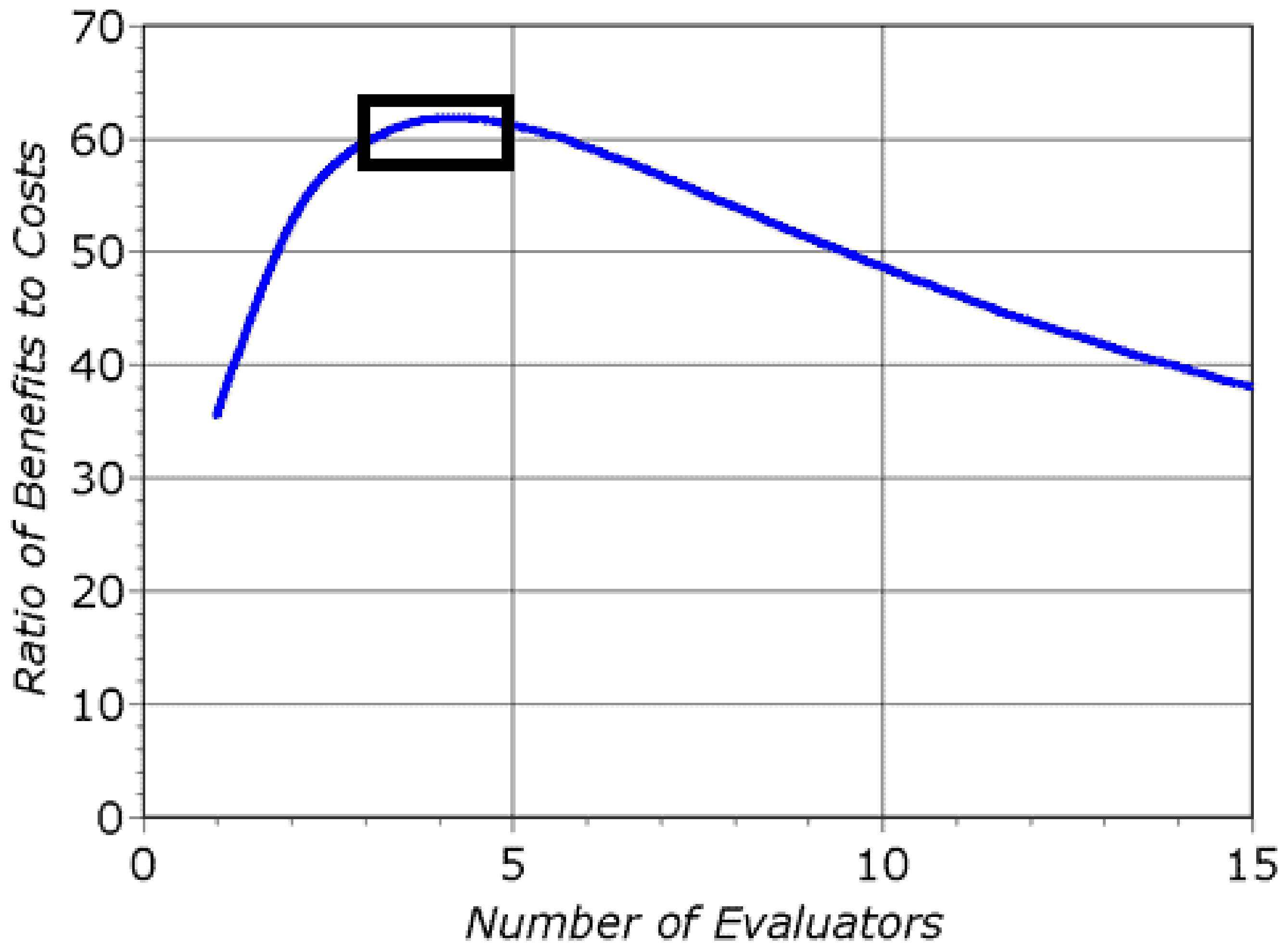
3-5 evaluators find 66-75% of usability problems

- different people find different usability problems
- only modest overlap between the sets of problems found



Problems found by multiple evaluators

Where is the best cost/benefit?



adopted from Saul Greenberg's slide deck

Heuristic Evaluation Process

Evaluators go through UI several times

- First to get feeling for flow and scope of system
- Second to focus on specific elements
- Inspect various dialogue elements
- Compares with list of heuristics
- Consider other principles/results that come to mind

If system is walk-up-and-use or evaluators are domain experts, then no assistance needed

Otherwise, might supply evaluators with scenarios

Each evaluator produces list of problems

- Explain why with reference to heuristic or other info
- Be specific and list each problem separately

Phases of Heuristic Evaluation

1) Pre-evaluation training
   - Give evaluators need domain knowledge and information on the senario

2) Evaluation
   - Individuals evaluate and then aggregate results

3) Severity rating
   - Determine how severe each problem is (priority)

4) Debriefing
   - Discuss the outcome with design team

Examples

Cannot copy information from one window to another

- Violates “Minimize the users’ memory load”
- Fix: allow copying

Typography uses mix of upper/lower case formats and fonts

- Violates “Consistency and standards”
- Slow users down
- Probably wouldn’t be found by user testing
- Fix: pick a single format for entire interface

Severity Rating

Used to allocate resources to fix problems

Estimates of need for more usability efforts

Combination of

- Frequency
- Impact
- Persistence (one time or repeating)

Should be calculated after all evaluations are in

Should be done independently by all judges

Severity Rating

1 – cosmetic problem

2 – minor usability problem

3 – major usability problem; important to fix

4 – usability catastrophe; imperative to fix

Debriefing

Conduct with evaluators, observers, and development team members

Discuss general characteristics of UI

Suggest potential improvements to address major usability problems

Development team rates how hard things are to fix

Make it a brainstorming session

- Little criticism until end of session

Homework #2

Pick a electronic device smaller than GPS navigators

Identify any usability problems through task analysis

Redesign UI on paper

Perform a heuristic evaluation

Redesign UI again on paper

Work-through with a user

Redesign UI using VB, VC++, C#, or Flash

Deliverables (compiled in a file folder):

- description of usability problems and tasks identified (11/7)
- initial paper prototype and heuristic evaluation result (11/14)
- second paper prototype and work-through result (11/21)
- final medium fidelity prototype (11/28)

Class presentation (12/2 & 12/5)