Handling Errors, Help & Documentation

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

• Handling errors
  – Error types
  – Slip types
  – Error prevention guidelines
  – Error recovery guidelines

• Help & documentation
  – Guidelines
  – Types of doc/help
  – Presentation issues
  – Doc organization
Errors

- Three considerations:
  - Avoiding and preventing
  - Identifying and understanding
  - Handling and recovering

Why errors are important

Errors are unavoidable
  - To err is human
  - Making mistakes is part of learning

Designer’s responsibility
  - Understand why errors occur
  - Minimize likelihood
  - Allow for recognition of error and graceful recovery (forward or backward)
User-Computer Dialog

- Three phases
  - Read-scan phase -- Perceptual errors
  - Think phase -- Cognitive errors
  - Respond phase -- Motor errors

Perceptual Errors

- Result from insufficient or poor perceptual cues
  - Examples
    - Display of objects that are visually similar
    - Invisible or poorly expressed states
    - Failure to capture user’s attention
    - Lack of perceivable feedback
Cognitive Errors

- Caused by taxing the memory and problem solving capabilities
  - Examples
    - Tax recall memory
    - Lack of or poor mnemonic aids
    - Inconsistency
    - Lack of context or status info
      - e.g., where came from in a menu
    - Mental calculations and translations

Motor Errors

- Taxing the eye-hand coordination and motor skills
  - Examples
    - Awkward motor movements
    - Highly similar motor sequences
      - e.g., double click, click
    - Pressure for speed
    - Require a high degree of hand-eye coordination
    - Requiring special types of motor skills (type)
Example Studies

- 170 experienced UNIX users over 9 days
  - Individual commands had error rates of 3-50%
    Kraut et al, CHI ’83
- 300 security system users over 20 months
  - 12,117 error messages
  - Most common 11 errors -> 65%
  - 2517 involved repeated errors (with no non-errors in between) within 10 minutes
    • → Bad error recovery/help

Mosteller & Ballas, Human Factors’89

Slips

- Automatic (subconscious) error that occurs without deliberation

- Examples?
Types of Slips

- 1. Capture error - Continue frequently done activity instead of intended one (similar starts)
  - Type “animation” instead of animate
  - Confirm deletion of file instead of cancel

- 2. Description error - Intended action has much in common with others possible (usually when distracted, close proximity)
  - ctrl key & caps lock key / Sun & Mac

- 3. Data driven error - Triggered by arrival of sensory info which intrudes into normal action
  - Call to give someone a number, dial that number instead

- 4. Associative activation - Internal thoughts and associations trigger action
  - Phone rings, yell “come in”
Types of Slips

- 5. Loss of activation - Forgetting goal in middle of sequence of actions
  - Start going into room, then forget why you’re going there

- 6. Mode errors - Do action in one mode thinking you’re in another
  - Delete file, but you’re in wrong directory

Error Prevention Guidelines (1)

- Eliminate modes or provide visible cues for modes
- Use good coding techniques (color, style)
- Maximize recognition, minimize recall
- Design non-similar motor sequences or commands
- Minimize need for typing
Error Prevention Guidelines (2)

- Test and monitor for errors and engineer them out
- Allow reconsideration of action by user (e.g., removing file from trash)

Error Recovery Guidelines (1)

- Provide appropriate type of response
  - Gag - Prevent user from continuing
    - Erroneous login
  - Warn - Warn user an unusual situation is occurring
    - Bell or alert box
  - Nothing - Just don’t do anything (Careful, user must determine problem)
    - Mac: move file to bad place
Error Recovery Guidelines (2)

- Responses (continued)
  - Self-correct - Guess correct action & do it
    - Spell-check correction
  - Dialog - System opens dialog with user
    - Go into debugger on run-time crash
- Query - Ask user what should’ve been done, then allow error action as legal one
  - Command language naming error

Error Recovery Guidelines (3)

- Provide undo function
- Provide cancel function from operations in progress
- Require confirmation for drastic, destructive commands
- Provide reasonableness checks on input data
  - Did you really mean to order 5000?
Error Recovery Guidelines (4)

- Return cursor to error field, allow fix
- Provide some intelligence
  - Guess what they wanted to do
- Provide quick access to context-sensitive help

Error Message - What to Say

- Error: Error code -37
- Description: Disk full
- Prescription: Disk full; recover disk space
- Prescription + aid: Disk full; recover space by deleting files or defragmenting
- Prescription + offer: Disk full; proceed with disk defragmentation? Otherwise delete files
Error Message Wording - Vocabulary

- Problem with previous example - some users will not know what defragmentation means!!
- Vocabulary
  - User-oriented
  - Defined in advance for commonality throughout all messages (in style guide)
- Alternatives to “defragmentation”?

Error Message Wording - Tone

- Sorry, command not recognized
- Command not recognized :-(
- Command not recognized
- Command not recognized!!
Or Even Worse

- Illegal command
- Illegal command!
- ILLEGAL COMMAND !@#&
- **ILLEGAL COMMAND!**

Which may suggest to some users ...

Help & Documentation

- It’s in the manual...
Customer Support

User Support

• Help
  – Problem-oriented and specific

• Documentation
  – System-oriented and general
Help & Documentation

- Never a replacement for bad design, but essential

- Simple systems may not use/require any
  - User walks up and uses it
  - Name some

- Most other systems with rich features require help

Documentation

- Many users don’t read manuals
  - Boring, no goal
  - Just dive in and start working

- Often used in panic mode, when user needs immediate help
  - Manuals probably locked away somewhere
  - Points to need for on-line help with search

- Sometimes want quick ref - emacs card
User Support Requirements (1)

- **Availability**
  - Should be available any time the user is operating the system

- **Accuracy & Completeness**
  - Should be accurate (tricky with changing versions) and should cover all aspects of application

User Support Requirements (2)

- **Consistency**
  - Across different sections, between on-line and paper documentation, in terminology, content and style

- **Robustness**
  - Should be predictable and free of errors
User Support Requirements (3)

- **Flexibility**
  - Appropriate for novices through experts, maybe by having expandable sections of details

- **Unobtrusiveness**
  - Shouldn’t distract from or interfere with normal work flow

Types of Doc/Help

- **1. Tutorial**
  - For start-up
  - Gets user going
  - Convey conceptual model
  - Communicate essential items
  - Sometimes see on-line tour or demo
Types of Doc/Help

Quick start guide as a tutorial

Getting Started

Before you can successfully install Eudora, you need to know the following information about your email account. Either your Internet Service Provider (ISP) or administrator can provide you with this information.

- **Return email address (Return address)**
  
  This is the email address that has been assigned to you by your ISP or your IT. It is the address other people use to send you email. An example return address is jrn@myfirm.com

  ```
  [Input field for Return address]
  ```

- **Username (Username)**
  
  This is the name you use to log into your email account. Your ISP or organization will generally, but not necessarily, assign the name that appears before the @ sign in your return email address (in the example above, "jrn" is the user name).

  ```
  [Input field for Username]
  ```

Tutorial Manual - Outline

1. Introduction
   - Assumed background of reader
     - Ref on where to get it
   - General capabilities
   - Key concepts - model, metaphor

2. Starter kit of tasks and how to accomplish
   - For each task - examples, screen shots
   - Introduce additional elements of conceptual model only as needed
   - How to deal with common errors / exceptions
   - Have plenty of examples, complete sample sessions
Tutorial Manual - Outline

3. More tasks
   – Introduce more commands as needed by tasks
   – More sophisticated uses of earlier commands
   – Changing defaults
   – Etc
   – Etc

N. Index
   § Organized by terms, concepts, tasks, commands

On-line Tutorial

- Work through simple examples, provide a feel for application
Types of Doc/Help

- 2. Quick reference/review
  - Reminder or short reference
  - Often for syntax
  - Can be recall aid for expert
  - Can allow novice to see what’s available

Example
Types of Doc/Help

- 3. Reference Manual (Full explanation)
  - Detailed command descriptions
  - Usually for experts
  - Unix on-line manual pages, for example
Types of Doc/Help

- 4. Searchable
  Sometimes it works....

Searchable

- And sometimes, it doesn’t....
Types of Doc/Help

5. Context-sensitive (task-specific) help
   - System provides help on current situation
   - Macintosh balloon help (old), ToolTips, for example
   - Other examples?
Medium

- Paper versus monitor
- Studies show that people are 15-30% slower reading and comprehending text from a display as compared to paper

Monitor

- Causes for slow-down
  - Poor fonts (monospace, bad kerning “VA”, bad spacing, …)
  - Low contrast of letters & background
  - Emitted vs. reflected light (curved tube)
  - Small display -> page turning
  - Distance, placement of monitor
  - Layout and formatting problems
  - Reduced hand and body motion
Presentation Issues

- Integrate with system, don’t “add on”

- 1. How is help requested?
  - Command, button, function, separate application
  - Advantages, disadvantages?

- 2. How is help displayed?
  - Separate window, whole screen, part of screen, on top of application, pop-up box, command line, highlighted button, light bulb...
  - Largely depends on what type of help it is

Presentation Issues

- 3. Effective presentation of help
  - Design it like any other part of UI: language, terminology, jargon, etc.
  - Use active voice
    - “To close a window, place the mouse cursor in the box at the upper right corner (with the X) and click the mouse button.”

- 4. Implementation issues
  - Fast response time is important
  - How is help stored? File, database, ...?
Adaptive Help

• Tailor help level and style to the particular user
• Usually requires a system to maintain a user model

Help Levels

• 1. Designer model
  – System designer has model of typical user and builds interface with this in mind
• 2. Adaptable help
  – User can edit their own model, for example, .profile on UNIX
• 3. Adaptive help
  – System maintains a user model and can change it on the fly
User Model

- How is user model constructed and maintained?
  - 1. Quantification - Numeric levels of use
  - 2. Stereotype
    - Novice, intermediate, expert
    - Utilize command use and errors to categorize
  - 3. Overlay model
    - Build expert user profile with optimal behavior
    - Compare to what user is currently doing

Adaptive Help Issues

- Initiative & control
  - Does user feel that control was taken away by system?
  - “You’re not performing efficiently in this task”

- Use
  - Is all this work actually useful?

- Scope
  - To what aspect of system or of help does it apply?
Documentation – Worth It?

- Studies have taken documentation and improved it
  - People did perform better with the improved documentation

- -> Effort here is worthwhile

Recommendations

- OK
  - All details of each command
  - BNF or formal notation
  - Terse, technical prose

- Better
  - Subsets of concepts
  - Lots of examples
  - Readable explanations with a minimum of technical terms
Doc Organization

- State educational objectives
- Present concepts in logical sequence, increasing order of difficulty
- Avoid forward references
- Make sections have roughly equal amounts of material
- Have plenty of examples, complete sample sessions

Doc Organization

- Each concept section:
  - Explain reason for concept
  - Describe concept in task-domain semantic terms
  - Show computer-related semantic concepts
  - Offer syntax
- Table of contents and index are important
- Keep reading level simple
Reading Level

- Study on doc at 5th, 10th, 15th grade reading levels among low, mid, high reading level people
- Reading level of person affected performance, but not reading level of text
- People liked 5th grade text best

Roemer & Chapanis, CHI '82

Improving Doc

- Run through think-aloud sessions
- Use on-line example tutorials
- Try to predict common states and problems
- Anticipate errors
- Develop manuals early and pilot test
- Iteratively refine
Human Characteristics

- Don’t anthropomorphize
  - “The computer will calculate an answer after you respond”
    - Gives user inaccurate impression
  - “You can get the solution by pressing F1”
    - Better to put user in control

Terminology

- Avoid
  - know, think, understand, have memory
  - ask, tell, speak to, communicate with

- Better
  - process, print, compute, sort, store, search, retrieve
  - use, direct, operate, program, control
Upcoming

- Prototyping and UI Software
- Exam
- Poster session