

# **Cooper Part III**

## **Designing Interaction Details**

**Jeff Offutt**

<http://www.cs.gmu.edu/~offutt/>

**SWE 632**

**User Interface Design and Development**

**Cooper, Ch 15-17**

### **Outline / Overview**

1. Searching and Finding (*ch 15*)
2. Undo operation (*ch 16*)
3. File systems (*ch 17*)

## Ch 15 : Searching

- As search engines get more powerful, users find more ways to use them
  - In the 1990s I brought paper to class for the syllabus and schedule
    - Changes would require more paper ...
  - A few years ago we would all bookmark the course website
  - Now we go to google and type in “swe 632”
- If we search within a particular website, we expect to get exactly what we want, first time !
- Searching represents a new way to retrieve information

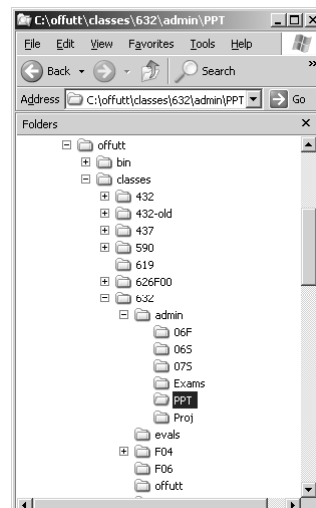
4/4/2011

© Jeff Offutt, 2004-2011

3

## Storing and Searching Files

- The windows file manager has a tree view of files
  - Navigating a tree is hard work ... up and down, up and down ...
- Searching is not integrated into the file manager
  - Start – Search – For files ...
  - Huge window ...
  - Doesn't know my current context (current directory)
  - Confusing fields
  - Folder doesn't seem to work



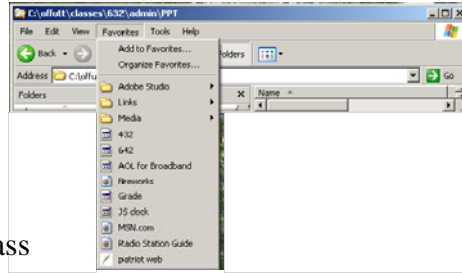
4/4/2011

© Jeff Offutt, 2004-2011

4

## Storing and Searching Files

- Unix has the concept of a “*classpath*”, which allows users to define certain often-visited directories as special and the *cd* command will look for those directories regardless of the current directory
- Unix *find* is very very slow
- Ahh ... the File Manager has a “Favorites” ...
- Oops ... this is connected to IE ... no favorites for File Manager
- In Unix, once we find the right directory, we can type  
`grep Upsorn grades*`  
to find Upsorn’s grades in any class
- How to do that in Windows ?



4/4/2011

© Jeff Offutt, 2004-2011

5

## Storage and Retrieval

- Generally, we need places to put our things
- And ways to find them when we need them
- Indexed retrieval works by creating an index of each thing we store
- To get to a file on our disk, we have to remember where it is !
  - Directory (folder) and file name

4/4/2011

© Jeff Offutt, 2004-2011

6

## Finding Digital Documents

- Three ways to find digital documents
  1. Positional retrieval – remember where it is
  2. Identity retrieval – remember its name
  3. Associative (attribute-based) retrieval – remember something about the file
- Positional and identity retrievals impose a memory burden on the user – hard with tens of thousands of files !
- Unix grep can be used as a primitive associative retrieval
- A “find and grep” script is slightly more advanced ...

4/4/2011

© Jeff Offutt, 2004-2011

7

## Attribute-Based Retrieval

- GUI systems still don't have usable attribute-based retrieval systems
- We'll get there
- The rest of chapter 15 goes into details about what such a system might look like

**This is not all that hard ...  
Why we don't have it, I do not know**

4/4/2011

© Jeff Offutt, 2004-2011

8

## Ch 16 : Undo

- Bet you thought undo was simple ...
  - Before you read the book
- Undo has multiple purposes
  - Rescuing mistakes
  - Exploration – finding out what functions do
  - Hypothesizing – looking for the correct function
- When users make mistakes, they tend to blame it on the software, computer, or UI
- UI should assume that everything the user does is intentional
  - “customer is always right”

4/4/2011

© Jeff Offutt, 2004-2011

9

## Types of Undo

- Procedural undo : Many undo operates on actions, not on data
  - Users are reverting to a previous state, but only programmers think that way
- Exploratory undo : User does not know what action will do, so tries it

4/4/2011

© Jeff Offutt, 2004-2011

10

## Types of Undo – Single vs. Multiple

- Most uses of undo is to go back one step
- Multiple undo is more expensive for programmer, but important for user
- But ... what if a user goes through 10 actions and the 3rd one was wrong?
  - Undo works on a LIFO queue
- Group multiple undo: Users can see a list of previous operations and select any of them
  - WordFormat (~1991), MS Word (~2003)
- Redo : After undoing, put it back
- Repeat : Apply the same command to next data

4/4/2011

© Jeff Offutt, 2004-2011

11

## Undo Summary

**Original hypertext theorists thought  
that a back button was for correcting mistakes**

**nope**

4/4/2011

© Jeff Offutt, 2004-2011

12

## Ch 17 : File Systems and Computer-Semantic Knowledge

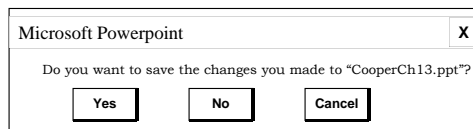
- Most non-engineers do not want to know the difference between files and memory
- TODAY
- But second graders now understand the difference!
- The concept of saving files is based on a strange artifact of computer hardware – the disk

4/4/2011

© Jeff Offutt, 2004-2011

13

### “Save Changes?” Dialog box



- The two choices are NOT equally likely
  - “Yes” is chosen almost all the time
  - Why does printing a document change it?
  - What is the difference between “no” and “cancel”?
    - “no” responds to the question, “cancel” does not
- We use this is a major undo operation
- This requires understanding the implementation model

4/4/2011

© Jeff Offutt, 2004-2011

14

## Mental Model

- Users think there is only one document, and they own it
- Implementation model:
  - Two documents, one on disk and one in memory
  - The computer program owns it
- When I bring my laptop to class, I don't have another, "real" one at home ...
- Users should not need to understand the file system!
- This is a user-centric view ...  
Outside-in design rather than inside-out

4/4/2011

© Jeff Offutt, 2004-2011

15

## Cooper's Unified File Model

- User's view: One copy of the file
- Users should be able to
  - Save documents automatically
  - Copy
  - Milestone copy (revision control)
  - Rename
  - Move
  - Change the format (Visio – vsd, jpg, eps, ...)
  - Reverse changes
  - Abandon all changes
- We don't need a "File" menu anymore ...

4/4/2011

© Jeff Offutt, 2004-2011

16

## Summary

Software construction of user interfaces is the easy part

The hard parts are

- understanding the users' perspective
- designing interaction that satisfies their needs rather than the needs of the designers