

# Command Language Guidelines

Jeff Offutt

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

SWE 632

User Interface Design and Development

Shneiderman, 4th, Ch. 8

## Selection vs. CL Interfaces

- Selection interfaces (menus, forms, GUIs) require
  - little memorization
  - the software initiates the process
- Command Languages (CL) require
  - users to initiate the process
  - more memorization
  - they are faster

Typing is always faster than clicking

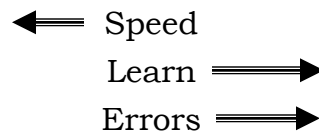
# Command Languages

## Advantages

- Faster
- More control
- Shell scripts (macros)
- Customization
- More flexibility

## Dis-advantages

- Memorization
- Less intuitive
- More errors



2-Feb-11

© Jeff Offutt, 2001-2011

3

# When to Use CL

- Speed is top priority
- Few commands (easy to memorize)
- Lots of commands (selection is painful)
- Lots of options

Voice interface

The future !

2-Feb-11

© Jeff Offutt, 2001-2011

4

# Designing Command Languages

1. Simple Command  
One command, one task
2. Command plus arguments  
more ...
3. Command + arguments + options  
more ...
4. Hierarchical command structure
  - Almost impossible to design
  - Seldom useful

2-Feb-11

© Jeff Offutt, 2001-2011

5

## CL Design: Command Plus Arguments

copy A B  
print X

arguments provide data  
abstraction

Defining separators:

- Blanks, commas, keywords=??, ...
- Be consistent

2-Feb-11

© Jeff Offutt, 2001-2011

6

## CL Design: Command + Arguments + Options

- Options modify the behavior
- Default should be:
  - most often used
  - least dangerous

```
cp -r d1 d2 // Copies recursively (directories)
lpr -Poffice f // Prints f on printer "office"
rm -i *.class // Deletes files, hesitates on each file
```

2-Feb-11

© Jeff Offutt, 2001-2011

7

## CL Design: Command + Arguments + Options (2)

Author: "Users gain satisfaction in overcoming the difficulties and becoming one of the inner circle who are knowledgeable ... " (about Unix)

I disagree: People appreciate the speed, power, and flexibility. Mouse-based interfaces are much slower and often do not provide functionalities that CL interfaces do.

2-Feb-11

© Jeff Offutt, 2001-2011

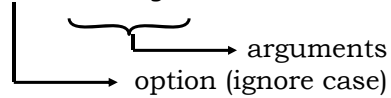
8

# Command Language Structure

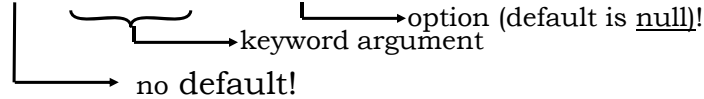
- Most CLs have many inconsistencies
- Most CLs are not designed, rather, each command is added and designed by individuals

## Unix

grep -i swe632 logfile



find . -name index.html -print



2-Feb-11

© Jeff Offutt, 2001-2011

9

# Consistency Points

- The ordering of the arguments
- The separator
- The syntax of the options

2-Feb-11

© Jeff Offutt, 2001-2011

10

## Choosing Command Names

1. Use words in user's vocabulary (common)
2. Use words with specific meaning

These goals are sometimes at odds

(1) aids learning

(2) aids retention

Shneiderman says to prioritize retention

move vs shift → not specific

delete vs remove → not common

2-Feb-11

© Jeff Offutt, 2001-2011

11

## Strategies for Abbreviation

1. Truncation: same length or different length?
2. Vowel drop + truncation
3. First + last letter
4. First letter of words in phrase
5. Standard abbrevs
6. Phonetically, by sound

2-Feb-11

© Jeff Offutt, 2001-2011

12

## Strategies for Abbreviation

- Choose a primary rule plus a secondary rule for conflicts
- Emphasize secondary rule abbreviations so users won't be confused
- Truncation is usually best
  - Use truncation to save users' typing, not screen display space
- Make commands easy to pronounce and “*completable*”

2-Feb-11

© Jeff Offutt, 2001-2011

13

## Voice-activated Systems

- Voice commands are finally becoming practical, causing a rebirth in CLs
- Voice commands must be:
  - short
  - simple
  - easy to pronounce
- Think about commands to your dog
- Think about commands in Star Trek, Next Generation

2-Feb-11

© Jeff Offutt, 2001-2011

14

## Defining and Describing a CL

- Define CLs with a grammar, then use yacc to help develop your interpreter
- Describe CLs the way you specify program methods :
  - Syntax
  - Errors
  - Hesitations (are you sure?)
  - Preconditions for command
  - Effects
  - Time to execute
  - Examples

2-Feb-11

© Jeff Offutt, 2001-2011

15

## Keyboard Shortcuts

- Keyboard shortcuts tend to be badly designed command languages
- Firefox :
  - Ctrl-F – Find
  - Ctrl-O – Open page
  - Ctrl-S – Save
  - Ctrl-P – Print
  - Ctrl-W – Close
  - Ctrl-Q – Exit
  - Ctrl-C – Copy
  - Ctrl-V – Paste
  - Ctrl-A – Select all

<u>Added after version 1</u> Back — backspace Reload — Ctrl-F5 Home — Alt-Home
---

2-Feb-11

© Jeff Offutt, 2001-2011

16

## Summary

- Small command languages show up in all sorts of places
- Most are designed badly
  - Most are not really “designed”
- Voice commands are special types of command languages
- Keyboard shortcuts are very limited types of command languages
- Command languages that satisfy the needs of users are very successful
  - Very few examples ...

**Designing a command language for users is not the same as designing an API for programmers !**