

Cooper Part II

Designing Behavior and Form

Jeff Offutt

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

SWE 632
User Interface Design and Development
Cooper, Ch 9-12

Outline / Overview

1. Platform and Posture (*ch 9*)
2. Orchestration and Flow (*ch 10*)
3. Eliminating Excise Tasks (and Navigation) (*ch 11*)
4. Designing Good Behavior (*ch 12*)

Ch 9: Posture

- Posture: The way the program appears to users
 - 1) Sovereign
 - 2) Transient
 - 3) Daemonic
- Different types of user interaction
- Different sets of behavioral attributes

21-Feb-11

© Jeff Offutt, 2004-2011

3

(1) Sovereign Posture

- The only program running
- Uses all of user's attention
- Runs for a long time
 - editors
 - word processors, PPT
 - spreadsheets
- Use the entire screen
- Users are experienced intermediates
 - Little time as a novice compared to total time using the software
 - Optimize for speed & power
- Make colors restful
- Provide a rich visual feedback
- Button icons can be small
- Allow flexible input – multiple ways to enter data
- Usually document-centric – users focus on a specific document

21-Feb-11

© Jeff Offutt, 2004-2011

4

(2) Transient Posture

- Invoked when needed, does a job, disappears
- Single-function, simple application
 - scanning a picture
 - sending mail
 - file manager
 - radio players
- Intermittent use
- Instructions must be clear and built-in
- Do not use more screen than necessary
 - Single window, single view
- Larger widgets

21-Feb-11

© Jeff Offutt, 2004-2011

5

(2) Transient Posture – cont'd

- Bright colors
- Avoid dialog boxes – they add complexity
- Don't cover crucial sovereign information
- Make the window movable
- Keep it simple
- Remember the state from the last use

21-Feb-11

© Jeff Offutt, 2004-2011

6

(2) Transient Posture Memory – Auto-customization

- Remember what the user did the last time
- Avoid unnecessary questions
- Imagine a secretary that asked you every time whether you wanted copies on front and back!
- Dialog boxes ask questions, buttons offer choices

21-Feb-11

© Jeff Offutt, 2004-2011

7

(2) Transient Posture Auto-customization Examples

- MS Word : I always put my files in C:\offutt
But MS Word always thinks I'm going to open a file in C:\Program Files\ ...
(took me years to find the customization!)
- PPT : I often print "Handouts", "2", "Pure black and white"
If I print several PPT files in a row, I have to click all three boxes every time!
- ATM : I usually withdraw \$150.
Why does the ATM always use \$40 and \$60 as defaults?

21-Feb-11

© Jeff Offutt, 2004-2011

8

(3) Daemonic Posture

- Usually do not interact with the user
 - printer drivers
 - network drivers
- Try not to bother users
 - I got a new printer recently, every time I print a little dialog box comes up to tell me it's printing
- Control panels are confusing and often unnecessary
 - But users do need them sometimes for configuration

21-Feb-11

© Jeff Offutt, 2004-2011

9

Designing for the Web

- HTML supports a very limited widget set
 - Javascript can be used to create many more widgets, but that is relatively complicated
 - This is becoming more common
- On the other hand, it takes fewer technical skills to implement a GUI front-end with HTML than with other technologies
- Complex, multi-screen transactional applications are just as hard, and perhaps harder, to develop
- HTML provides greater separation between engineers, GUI designers, and graphics designers

Some Web-specific usability points ...

21-Feb-11

© Jeff Offutt, 2004-2011

10

Postures for the Web

- Same postures, some specializations
- Types of Web sites
 - Information-oriented sites
 - Offer information, no interaction (632 class web site)
 - Sovereign : frequent users, detailed information, navigation
 - Transient : occasional users, bookmarked, less navigation
 - Transactional sites and Web applications
 - Full user interaction – entering data, modifying server state
 - Frequency of use is also important to posture
 - Web applications
 - Heavily interactive and provide complex functionality
 - Should usually try to look more like a sovereign
 - Design to look like a desktop application – remember that the request / response interaction is confusing to users

21-Feb-11

© Jeff Offutt, 2004-2011

11

Terms

- Web Site : Provides information but not interaction
 - cs.gmu.edu, 632 web site, washingtonpost.com, ...
 - Navigation is important
- Web Applications : User interactive programs that are deployed on the web
 - Amazon, patriotweb, ...
 - User interaction is important
- Types of Web Applications:
 - Sovereign Web Apps
 - Transient Web Apps
 - Internet-enabled

21-Feb-11

© Jeff Offutt, 2004-2011

12

Sovereign Web Apps

- Sovereign : Rich interactive experience
 - Make people feel they are in an environment, not looking at pages
 - Build full-screen applications
- The fact that it is a Web application should not be obvious
 - Design them as if they are desktop applications
 - Include multiple controls and data objects
 - Use panes to group functions (not frames!)
- The UI should emphasize interaction, not navigation
- Remember that users are often infrequent users
- The transaction flow must match the user's mental model
 - Don't make users scroll back up or down to proceed to the next step
- Use Ajax and JS to avoid request / response cycle
 - Request / response is odd to most users

21-Feb-11

© Jeff Offutt, 2004-2011

13

Transient Web Apps

- Transient : Quick occasional access to information
 - Unfortunately, browsers run as full screen apps ...
- Interactive components of web pages, often inside informational web sites
 - *cnn.com*'s Weather and QuickVote boxes
- They should very clearly explain what they are
- Very simple and small
- Should fit within the rest of the web site
 - Follow the user's mental model
- Make login or data retrieval as simple as possible
- We need more integrated transients like google & weather

...

21-Feb-11

© Jeff Offutt, 2004-2011

14

Internet-enabled Applications

- Use the Internet, but do not run inside a Web browser
 - The most common technologies are Java applet and tools that are effectively mini, special purpose browsers
 - RealPlayer is a common example
- Advantages
 - Can add more robust controls than HTML supports
 - Easier to access data on disk
 - Performance

Web Design Summary

**The ability to write HTML
is not the same as
the ability to design
an effective Web-based UI**

Designing for Embedded Systems

- An embedded system is software that is integrated into a device that is not primarily for computing
 - Phones, PDAs, remotes, ATMs, TVs, appliances, cars, planes, ...
- The input / output devices are often much different
 - Often much more constrained in abilities
- Designing the UI is very different from designing UIs for desktop applications

21-Feb-11

© Jeff Offutt, 2004-2011

17

General Principles

- Don't think of the product as a computer
- UI design must coordinate with hardware design
- External environment will affect the UI
- Should use many fewer modes
- Navigation is harder and should be reduced
- Software must be more reliable and usable

21-Feb-11

© Jeff Offutt, 2004-2011

18

Phones

- Many cell phones have huge hierarchical menus that are very confusing
 - All functions are equally difficult to find
 - But turning off the ringer is much more common than setting the “NAM”
- Very small screen
- Limited keyboard
- It should be easy to synchronize with phone books on my desktop computer
- UI designer needs to specify the buttons on the hardware
- Function bloat : who uses all that junk?

21-Feb-11

© Jeff Offutt, 2004-2011

19

Handheld Devices

- Consider how they are held – two hands or one?
- Do they connect with a desktop application?
 - How often ?
 - If they do, the UIs should be very different (consider the Palm)
- No pop-up windows or dialog boxes
- No dragging
- Make controls and fonts large and bright
 - Let me use my finger, not a nerdy pen
 - Don't make me shift my input mode (fingers to pen)

21-Feb-11

© Jeff Offutt, 2004-2011

20

ATMs and Kiosks

- These are publicly accessible computers with specialized applications
- Consider the environment:
 - Lighting, noise, privacy, security
 - The ATM in Johnson Center blocks foot traffic
- Kiosks are infrequently used
- Consider the main use
 - Transaction : provide a service (ATM, tickets, self-checkout)
 - Exploration : present information (museums)
- No dragging, scrolling, or on-screen keyboards

21-Feb-11

© Jeff Offutt, 2004-2011

21

Embedded Systems Summary

Embedded software will be a major growth area for the foreseeable future

21-Feb-11

© Jeff Offutt, 2004-2011

22

Posture Summary

Everybody wants to design sovereigns!

Effective UI designers must know their users

21-Feb-11

© Jeff Offutt, 2004-2011

23

Ch 10 : Orchestration and Flow

- **Flow:** The next thing the interface wants to do is exactly what the user expects
 - Follow users' mental model
 - Let the user direct the software
 - Don't talk with the user
 - Keep all related tools available
 - Modeless feedback : The user should not have to respond
- Interfaces should be invisible, not cool
- It's easy to make things complicated, it's hard to make things simple!

21-Feb-11

© Jeff Offutt, 2004-2011

24

Flow is a State of Mind

- When users are “in the flow”, they exhibit very good concentration and are not easily distracted
- Dealing with syntax always interferes with flow
- Surprises interrupt flow
- Good flow allows users to concentrate on task-semantic knowledge, not syntactic

21-Feb-11

© Jeff Offutt, 2004-2011

25

Flow Example

Do you want to save?			
Yes	No	Cancel	Help

Of course!!! Don't ask me.

Yes, GUIs are differentiated from CLs by being easier, but slower ...

But, that is not an excuse to design them to be slow!

21-Feb-11

© Jeff Offutt, 2004-2011

26

Orchestration – Stay in Character

- First rule in acting : Always be the character you are playing, not yourself
- Syntax interrupts users' thinking and annoys them
- Good actors become the person they are playing
- Good basketball referees disappear

21-Feb-11

© Jeff Offutt, 2004-2011

27

Make the UI Disappear

- Less interaction, not more – what do users need?
- Make likely choices (probabilities) default, and unlikely choices (possibilities) available
 - “Do you want to save?” duhhh !
- Give information, not data – 40% saved, not 20,000 bytes
- Software should indicate status visually – active, busy, idle
- Don't use dialogs to report normal behavior
- Provide default behavior and mechanisms to change it
- Separate commands (print) from configuration (setup)
- Don't ask questions, give users choices
- Make dangerous choices hard to reach

21-Feb-11

© Jeff Offutt, 2004-2011

28

Design for the Probable Provide for the Possible

- Choices should be based on probability, not logic
 - Logic : 1 out of a million is still possible ... *if p then s1 else s2 ...*
 - Probability : Make the 999,999 default, make the 1 hard to find
- Default should always be to save changes when I exit
 - Of course I want to save !
 - Include an option to discard and exit ... in a menu somewhere

21-Feb-11

© Jeff Offutt, 2004-2011

29

Flow Summary

**To provide good flow,
designers must know their users**

21-Feb-11

© Jeff Offutt, 2004-2011

30

Ch 11 : Overhead & Navigation

- Overhead relates to solving problems:
 1. Revenue Tasks: Sub-tasks that work to solve the problem directly
 - Designing, requirements
 - Studying, homework, listening to lectures, taking tests
 2. Excise Tasks: Sub-tasks that must be done but that are not really part of the problem
 - Compiling, debugging
 - Driving to school, parking, “makework” kind of homework
- Excise tasks often satisfy the needs of the tools, not the users

21-Feb-11

© Jeff Offutt, 2004-2011

31

Overhead Philosophical Lineage

- This is related to the “*accidental and essential*” problems of Fred Brooks
 - Frederick P. Brooks, “No Silver Bullet: Essence and Accidents of Software Engineering,” *Computer*, Vol. 20, No. 4 (April 1987) pp. 10-19
 - <http://www.virtualschool.edu/mon/SoftwareEngineering/BrooksNoSilverBullet.html>
- More historically, to Aristotle’s philosophy on “*accidents*” and “*essential*” categories of nature
 - <http://plato.stanford.edu/entries/aristotle-metaphysics/>
- A major goal of software engineering is to reduce the number of accidental problems engineers have to solve

21-Feb-11

© Jeff Offutt, 2004-2011

32

Essential and Accidental Problems

- Essential problems :
 - Difficulties that are inherent in the nature of software
- Accidental problems :
 - Difficulties related to the production of software
- 80 / 20 rule (one version)
 - Immature engineering fields spend 80% of the time on accidental problems
 - Mature engineering fields spend 20%
 - Software engineering is probably approaching 50 / 50

From Brooks' paper

21-Feb-11

© Jeff Offutt, 2004-2011

33

4 Reasons Why Software is Hard

1. Complexity : Software is by far the most complex thing human have every built
2. Conformity : Software is expected to conform to users and the system
3. Changeability : Software is expected to constantly change
4. Invisibility : Software is made out of bits, not atoms, so there is no physical reality

From Brooks' paper

21-Feb-11

© Jeff Offutt, 2004-2011

34

Essential and Accidental Problems

- Most software engineering advances try to solve accidental problems
- Skepticism is not pessimism
 - Magic solutions tool vendors claim will not solve this
 - Buzzword processes will not solve this (spiral, structured, OO, agile, TDD, ...)
- We've made slow steady progress in 20 years
 - By basic research, solid engineering, and technology transition
 - Strong typing, modularity, inheritance, modeling, IDEs, test criteria, education, ...

From Brooks' paper

21-Feb-11

© Jeff Offutt, 2004-2011

35

Excise Tasks

- Excise tasks are trivial, unless we have a lot of them
 - Eliminate them if possible
 - Automate them as much as possible
- Excise for users with comp-semantic knowledge is often perceived as revenue for users without

21-Feb-11

© Jeff Offutt, 2004-2011

36

GUI Excise

- Users who are competent with command lines see a lot of excise in GUI – primarily the navigation
 - Partly using the mouse
 - Partly having to go through multiple screens
 - Generally – GUIs require more navigation (ch 11)
- Example : Changing background in all SWE 632 slides
 - PPT : More than 30 minutes; load each file separately, 1 or 2 minutes to change each file
 - VIM : Less than five minutes; one process, repeat searching and commands (assuming text files)
- Convert 20 files to PDF
 - Word : about an hour, print dialog for each file
 - Latex : 3 minutes, write a simple shell script

21-Feb-11

© Jeff Offutt, 2004-2011

37

Command Line Excise

- Users must learn all the syntax – a significant tax !
 - Equivalent to learning programming languages
 - CLs are primarily preferred by programmers
- Command line users will often make extensive use of shortcuts and customization in GUIs

21-Feb-11

© Jeff Offutt, 2004-2011

38

Balancing Beginners and Experts

- Beginners need help in the form of simple ways to accomplish tasks
- This help usually slows down expert and intermediate users
- The help that beginners need is excise for everyone else!

- Must make it easy to eliminate introductory features

Identifying excise requires knowing user goals

21-Feb-11

© Jeff Offutt, 2004-2011

39

Avoiding Excise

- Put the mouse focus in the first input box
- Don't interrupt flow unless absolutely necessary
- Try not to show error messages
- Don't ask users to "correct" what they don't understand
- Don't separate input from output
- Don't require passwords for everything
 - washingtonpost.com ???
 - Security is always excise !
- Don't force the user to go to different windows
- Don't make users remember where files are
 - MUST let users define file organization (MS Word does not)

21-Feb-11

© Jeff Offutt, 2004-2011

40

Avoiding Excise – cont'd

- Don't make users move or resize windows
- Don't make users remember or reenter personal settings
- Don't make users enter unnecessary data
 - Telephone number for DB key – use the name or invent a number!
- Don't make users confirm actions – unless undo is impossible
- Don't let users make errors – haha!

Excise Summary

**It's easy to make things hard
It's hard to make things easy**

Navigation

- Navigation is moving around a UI
- Navigation is never about the users goals
- Navigation is always excise
- Four types of navigation
 1. Among multiple windows and screens
 2. Navigation among panes or frames in a window
 3. Navigation among tools and menus
 4. Navigation within information (scrolling, zooming, ...)

21-Feb-11

© Jeff Offutt, 2004-2011

43

(1) Navigation Among Windows

- The most intrusive because all information changes
- Navigational trauma : User loses track of location
- Avoid making users toggle back and forth
- Don't make users remember something from one screen to another

21-Feb-11

© Jeff Offutt, 2004-2011

44

(2) Navigation Among Panes

- Use adjacent panes on one window to
 - Provide supporting functions
 - Display related data
 - Provide helpful links
- Too many panes can become confusing
- Group related panes together
- Try to avoid scrolling
 - If scrolling is necessary, user should only have to scroll once, not back and forth
- Tabbed panes can group multiple documents and separate views of the same document
 - Tabbing is a navigation that hides the previous screen

21-Feb-11

© Jeff Offutt, 2004-2011

45

(3) Navigation Among Tools & Menus

- Group commonly used tools
- Group tools that are used together
- Make frequently used tools immediately available
 - In toolbars or palettes
- Menus require more navigation than toolbars and palettes

21-Feb-11

© Jeff Offutt, 2004-2011

46

(4) Navigation Within Information

- Try to avoid scrolling when possible
 - Horizontal scrolling is very difficult
 - Scrolling “back and forth” is slow & annoying
- Linking is disruptive and needs to be clearly marked
 - Non-Web applications are starting to include link idioms
- Navigating within information needs to be fast
 - Mouse is a particularly slow tool
- Compare the navigation in MS-Word with VIM

21-Feb-11

© Jeff Offutt, 2004-2011

47

General Hints for Improving Navigation

- Reduce the number of places to go
 - Fewer pages, panes, etc
- Include signposts
 - Use persistent screen objects to inform users (titles, menus, ...)
- Include overviews so the users can have context
 - Should be persistent
 - Web “breadcrumbs” and annotated scrollbar (*PPT*)
- Map controls to functions carefully
 - This is about rate of user errors
 - Both the target and the result of an operation must be clear

21-Feb-11

© Jeff Offutt, 2004-2011

48

General Hints for Improving Navigation

- Organize navigation for most common actions (*inflection*)
 - Users are willing to work harder for low frequency actions
 - I keep my soap on the sink, dishwasher soap in the front, and Comet in the back of the cabinet
 - This depends on the user
 - Auto-customization changes availability based on statistics
 - User-customization allows users to move actions around by hand
- Keep hierarchies shallow (2 or 3 levels)
 - Programmers are good at logic and abstraction – hierarchies
 - Many users are not
 - Help users find information

21-Feb-11

© Jeff Offutt, 2004-2011

49

Navigation Controls

Duhhh ... which knob controls which burner ?



21-Feb-11

© Jeff Offutt, 2004-2011

50

Navigation Summary

Cooper thinks that navigation is the number one problem in UI design

21-Feb-11

© Jeff Offutt, 2004-2011

51

Ch 12 : Designing Good Behavior

- People respond to computer user interfaces as if they are sentient beings
 - We should make our software considerate, likeable, supportive
 - What does the human need ?
- People should think, computers should work
- This is not hard ...
 - But being considerate takes more time than being inconsiderate
 - Inconsiderate people think they are being efficient ...but many of their colleagues think they're self-centered, curt, and cryptic
- Remember that the software is there to help the user, not the other way around

21-Feb-11

© Jeff Offutt, 2004-2011

52

Making Software Considerate

- Take an interest
 - Remember things about the users
 - Google map, firefox, ...
 - Considerate people ask what you like once, then remember
 - PPT never remembers my print preferences or my file saving preferences
- Considerate products are deferential
 - Give users choices, not orders
 - Never judge users – don't tell users they're wrong, the software doesn't understand
- Considerate products are forthcoming
 - Give users related information that might help them
 - The print queue is full ...
 - Suggest possible words on misspellings ...

21-Feb-11

© Jeff Offutt, 2004-2011

53

Making Software Considerate – cont'd

- Considerate products use common sense
 - Put controls in logical places
 - A considerate program would never think I want my class roster for the next semester
- Considerate products anticipate human needs
 - Operating systems “pre-fetch” sectors that are near the last sector we read ... why don't web browsers do the same ?
- Considerate products are conscientious
 - Consider the broader goal
 - When copying a file with the same name – should let me merge, show me the differences, rename the old file, or simply overwrite
 - When printing color ppt to a black-and-white printer, should automatically change to “pure black and white”

21-Feb-11

© Jeff Offutt, 2004-2011

54

Making Software Considerate – cont'd

- Considerate products don't burden you with their personal problems
 - A receptionist at the dentist's office who complains about her schedule to customers is very annoying
 - Software should not :
 - Tell us it successfully saved ...
 - Whine about a full recycle bin ...
 - Tell us it cannot render some weird fonts ...
 - Just do it !
- Considerate products keep us informed
 - Tell users about what matters to the users
 - What can I do next ?
 - How do I complete my order ?
 - How do I quit ?
 - How can I change something later ?

21-Feb-11

© Jeff Offutt, 2004-2011

55

Making Software Considerate – cont'd

- Considerate products are perceptive
 - If I check my roster in the morning, then I log into the system that afternoon, shouldn't it automatically tell me if anybody added or dropped the class since my last check ?
 - If I resize adobe reader to use the full height of my screen and be just wide enough to fit a document ... shouldn't it create the window with that size next time ?
 - When I print with "Adobe PDF" in PPT, I always choose "Handouts", "Pure Black and White" and "Slides per page" = 2 ... PPT should notice and remember
- Considerate products are self-confident
 - "are you sure ?" just gets in the way ... on the other hand, the software should provide an "undelete"

21-Feb-11

© Jeff Offutt, 2004-2011

56

Making Software Considerate – cont'd

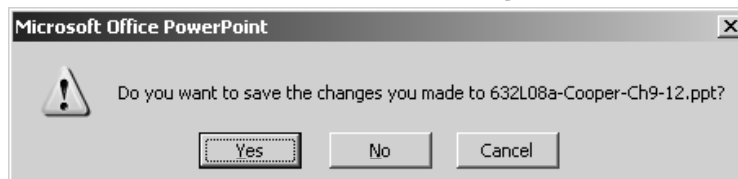
- Considerate products don't ask a lot of questions
 - Give users choices, not questions
 - Don't offer choices nobody ever wants
 - Don't offer choices whose consequences are not clear
 - “are you sure you want to quit? (yes, no, cancel)”
 - What does cancel do !!!!
 - About once a week I shut down my computer and go to bed ... the next morning when I look at my computer I find a dialog box “something is still open, should I shut down?” ... when I say “yes”, then it shuts down, wasting time as well as electricity
- Considerate products fail gracefully
 - When you fail, apologize and try to fix it (kindergarten lesson ...)
 - Don't throw away data when crashing ...
 - If I fill out 10 form fields and get one wrong, keep the data from the other 9

21-Feb-11

© Jeff Offutt, 2004-2011

57

An Inconsiderate Question



- Uhhmmm I didn't change anything ...
- All I did was print !
- Did I accidentally change something else ?
- Why are you bothering me ?
- Are you stupid or am I ?
- What does “Cancel” do ?
- Mommy ! Help !

21-Feb-11

© Jeff Offutt, 2004-2011

58

Making Software Considerate – cont'd

- Considerate products know when to bend the rules
 - To reimburse a visitor for a trip, we have to create a banner record
 - We have to have a G-number
 - To get a G-number, we need a social security number !
 - The automated system is stupid enough to check that the SSN is valid ...
 - Allow users to complete part of the process today and come back later to finish
 - Binary logic : yes/no, true/false, done/not done
 - Trinary logic : yes/no/maybe, true/false/possibly, done/not done/started ...
 - In real life, we never get the rules right
 - We need some way to allow for when the rules don't quite fit
- Considerate products take responsibility
 - Software should understand hardware devices and deal with them
 - When my printer runs out of paper, my computer tells me “could not print” ... so I fill the printer up and print again ... then the printer finishes my first print and happily prints my second copy!

21-Feb-11

© Jeff Offutt, 2004-2011

59

Designing Smart Products

- This section makes some interesting general points, but few specific actionable suggestions (*research agenda?*)
- Smart products take care of users
 - They don't behave intelligently
- Smart products are proactive, not just reactive
 - Most computers spend > 90% of their time doing ... nothing
 - Use the time to look for files the user might need
 - Cooper spends several paragraphs complaining that computers don't use spare cycles to help the users, but does not offer any other concrete suggestions
- Smart products have a memory : A way to track and use actions that users take over multiple sessions
 - Remember previous preferences and choices
 - “Dynamic defaults” resets defaults based on the user's past history
 - It's easier for programmers to let the UI ask the user what to do each time – it's easier for users if the UI remembers

21-Feb-11

© Jeff Offutt, 2004-2011

60

Designing Smart Products

- Task coherence : Our goals and how we achieve them is usually similar from day to day
 - Similar usage patterns in software
 - Often the same document, or documents in the same directory
 - Why should I tell Word to full justify paragraphs ... every time ?
 - Why should my phone keep asking “did you say ...” when I say “yes” ... every time ?
 - Applications should remember how big they were and where they were on-screen
 - If a user goes through the same sequence of commands several times – the application should automatically create a macro
- What should a program remember ? **EVERYTHING !**
 - Disk space is now not a problem for saving information – save everything !
 - My TA emails me the grade spreadsheet at least once a week and I put it in the same place every time ... yet every time, my mail client asks me where to put it ... can you imagine an assistant acting that stupidly ?

21-Feb-11

© Jeff Offutt, 2004-2011

61

Designing Smart Products

- Software should automatically keep track of old versions
 - Change management algorithms were developed **THIRTY YEARS AGO !**
 - I’ve been using rcs / cvs since 1982
 - Why doesn’t Word keep track of old versions automatically ?
 - Why can’t I compare the contents of two Word documents, line by line ?
- Software should keep track of the undo stack between sessions
 - Office products throw away the undo stack when we save
 - Very 1995 “memory is scarce” thinking ...
- Data fields should remember the values entered in previous sessions
 - Modern browsers have started doing this

21-Feb-11

© Jeff Offutt, 2004-2011

62

Five Powerful Ideas

1. Posture of UIs
2. Flow through the UIs
3. Overhead of UI interaction
4. Navigation among windows, panes, tools and information
5. Considerate user interfaces

**Remember to think about
the users and you'll be fine**