ForUse
DESIGN

SEC(R) 2007

Designing Software for User Performance

Universidade da Madeira

Larry Constantine, IDSA
Director | Laboratory for Usage-Centered Software Engineering
University of Madeira | Funchal, Portugal

Software Engineering Conference (Russia)
Moscow | 1-2 November 2007

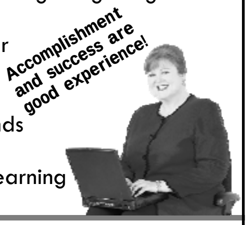
Constantine & Lockwood, Ltd.
Lucy A. D. Lockwood, President | Larry L. Constantine, Chief Scientist
email: llockwood@forUse.com | email: lconstantine@forUse.com | www.forUse.com

58 Kathleen Circle, Rowley, MA 01969, U.S.A. | tel: 1 (978) 948 5012; fax: 1 (978) 948 5036

User Experience, User Performance

User experience - is about everything!
- whole person: emotional, physical, cognitive,…
- entire system/environment: physical, social, psychological,…
- creating "compelling," "engaging" experience
- promoting user satisfaction in every sense

User performance - is about tools for getting things done.
- promoting user success
- enabling users to do more, faster
- enabling users to do new things
- supporting user problem solving
- supporting varied means and ends
- promoting reliable performance
- supporting mastery with rapid learning



1

Usability as User Performance

More than "easy to use," not "feature packed," and **never** "user friendly." Multiple factors:*

★ acquisition	immediate productivity, easy to learn
retention	easy to remember how
★ efficiency	rapid task completion
★ reliability	dependable action and input
satisfaction	goal satisfaction in particular
proficiency†	continual skill improvement
convenience†	perceived efficiency, simple operation

★ Most important performance objectives.
- User performance depends on what is presented where and how, how it is organized, and how it is interacted with.

* adapted from Nielsen, 1993; Shneiderman, 1992 † Constantine

2

Designing for User Performance

- Supporting User Performance
- Organizing for User Performance
- Using Instructive interaction
- Reducing Mistakes and Messages
- Eliminating Hidden Inefficiencies

3

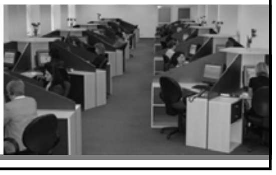
Processes for User Performance

- Design from the outside in, not inside-out.
- Understand and detail the task needs of users based on work in which they are engaged:
  - usage-centered design
  - contextual design
  - scenario-based design
  - activity-driven design
  - use case modeling,…
- Prioritize tasks by anticipated frequency and importance to user success/performance.
- Design for streamlined user performance based on prioritized tasks.
- Pay attention to details of interaction design.

4

Supporting Activity

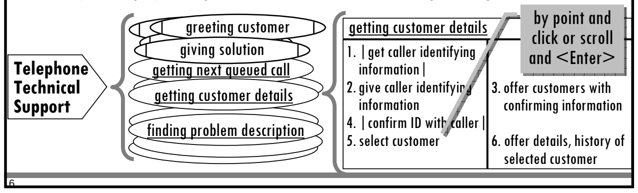
- Understand the larger activity in which your users are engaged.
- Activity - a loosely organized collection of tasks and actions involving artifacts and other players serving a common or shared purpose.
- Focal activities - activities involving user actors with the focal system to be designed.
- Adjacent activities - other activities in the same field (time and place) or involving some of the same participants or resources
- More strongly connected activities are potentially more important for interaction design.



5

Activities, Tasks, and Operations

- Activities, Actions/Tasks, and Operations are three levels of analysis for understanding and supporting work in context.
- **Activities** are unstructured or loosely structured collections of **Tasks** or use cases (interactions with the focal system) and **Actions** (interactions with people, other systems, and artifacts).
- Tasks consist of one or more elementary **Operations** (or steps) adapted to conditions to complete goals.


Telephone Technical Support
greeting customer
giving solution
getting next queued call
getting customer details
finding problem description
getting customer details
1. | get caller identifying information |
2. give caller identifying information
4. | confirm ID with caller |
5. select customer
3. offer customers with confirming information
6. offer details, history of selected customer
by point and click or scroll and <Enter>

6

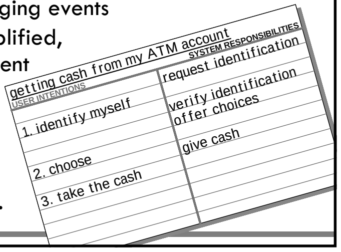
Prioritizing Tasks

- Even in absence of data, we can often make reasonable rank-order judgments on expected frequency, user importance. For example:
  - checking current time is more frequent than setting time is more frequent than changing date.
  - "happy case" is more important for typical user performance than exceptions.
- Card sorting techniques
  - in-hand sort
  - in-line on table
  - bucket (category) sort
- Collaborative sorting: work independently or without talking.

7

Capturing Task Essentials

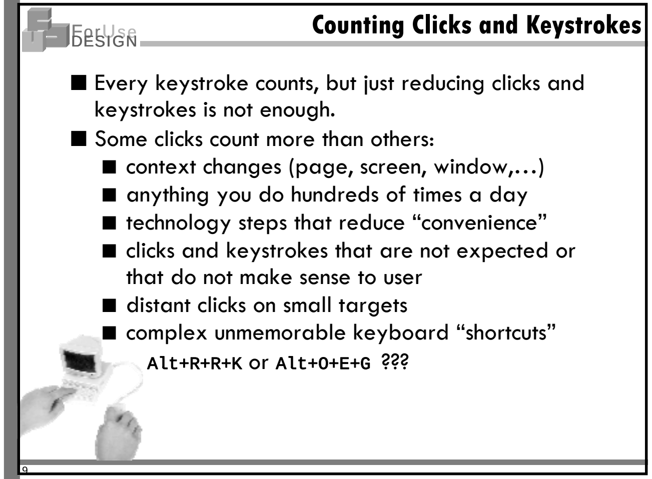
- Task modeling with essential use cases (also called "task case") establishes a user performance baseline.
- A single, discrete intention that is complete, well-defined, and meaningful to a user, in some role as part of an activity, not a complete job or scenario. E.g., defining new event details or canceling event within activity of managing events
- Essential: abstract, simplified, generalized, independent of technology and implementation .
- Raises the bar on interaction design.
- Promotes performance.



getting cash from my ATM account

USER INTENTIONS	SYSTEM RESPONSIBILITIES
	request identification
1. identify myself	verify identification
	offer choices
2. choose	give cash
3. take the cash	

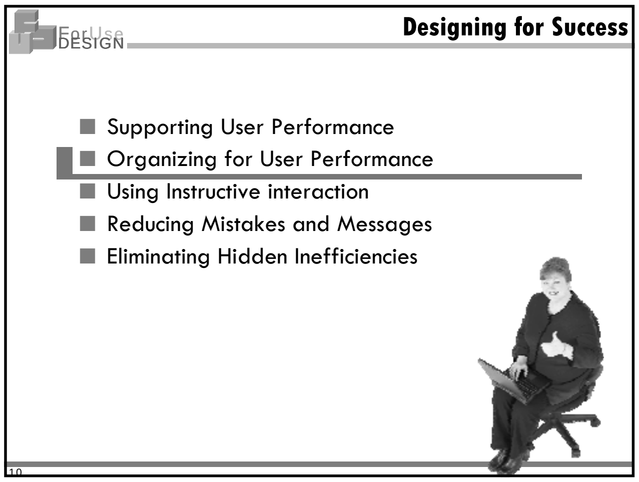
8



Counting Clicks and Keystrokes

- Every keystroke counts, but just reducing clicks and keystrokes is not enough.
- Some clicks count more than others:
  - context changes (page, screen, window,…)
  - anything you do hundreds of times a day
  - technology steps that reduce "convenience"
  - clicks and keystrokes that are not expected or that do not make sense to user
  - distant clicks on small targets
  - complex unmemorable keyboard "shortcuts"
    `Alt+R+R+K` or `Alt+O+E+G` ???

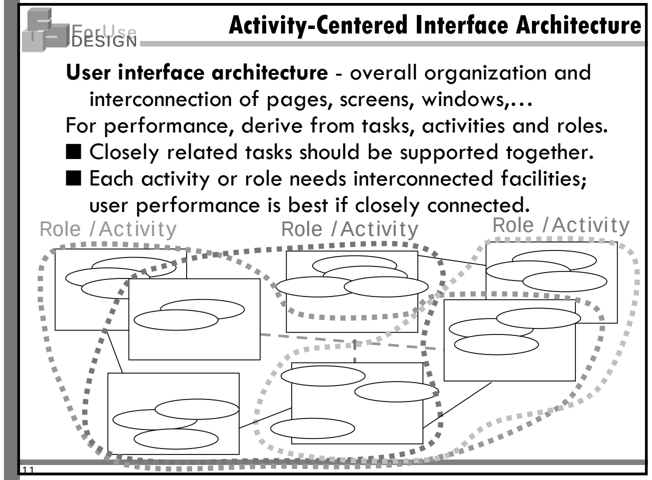
9



Designing for Success

- Supporting User Performance
- Organizing for User Performance
- Using Instructive interaction
- Reducing Mistakes and Messages
- Eliminating Hidden Inefficiencies

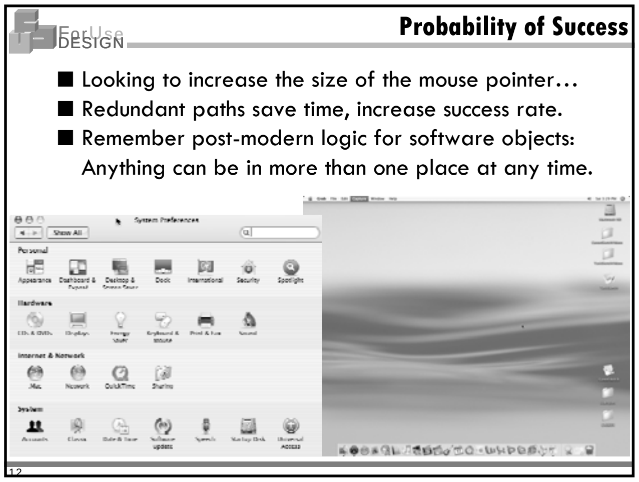
10



Activity-Centered Interface Architecture

User interface architecture - overall organization and interconnection of pages, screens, windows,…
For performance, derive from tasks, activities and roles.
- Closely related tasks should be supported together.
- Each activity or role needs interconnected facilities; user performance is best if closely connected.

11



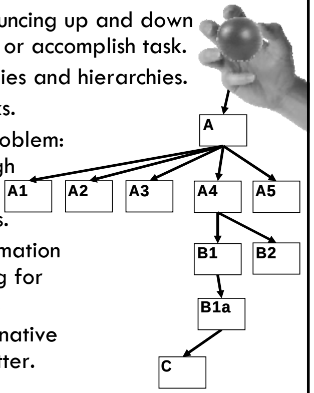
Probability of Success

- Looking to increase the size of the mouse pointer…
- Redundant paths save time, increase success rate.
- Remember post-modern logic for software objects: Anything can be in more than one place at any time.

12

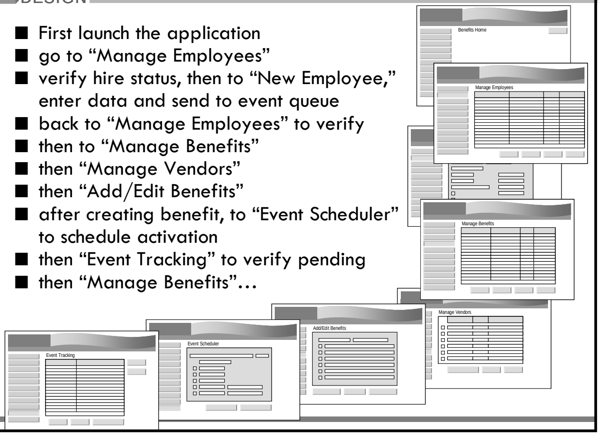
Page Bouncing and Deep Drilling

- Architectural problem: bouncing up and down pages to find information or accomplish task.
- Cause: thinking in categories and hierarchies.
- Solution: organize by tasks.
- A related architectural problem: deep drilling down through many links to enact common tasks or scenarios.
- Solution: reorganize information and links to reduce drilling for common and critical tasks.
- In general, the more alternative routes to a target, the better.



Spraying the Database on the Screen

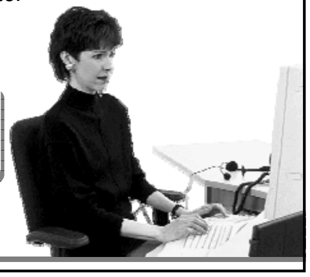
- First launch the application
- go to "Manage Employees"
- verify hire status, then to "New Employee," enter data and send to event queue
- back to "Manage Employees" to verify
- then to "Manage Benefits"
- then "Manage Vendors"
- then "Add/Edit Benefits"
- after creating benefit, to "Event Scheduler" to schedule activation
- then "Event Tracking" to verify pending
- then "Manage Benefits"…



Everything in It's Place

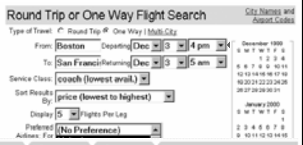
- Good interface architecture gives users what they need, where and when they need it. WYSIWYN!
- It enables users to accomplish their goals without having to bounce around or drill down into the dark depths of the user interface.
- Every time the interaction context refreshes…

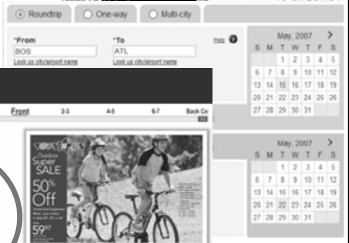
"Windows and dialogue boxes are like rooms. The designer should have a good reason to make the user have to go to another room." —Alan Cooper

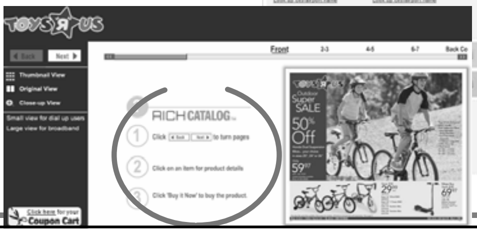


Organization and Efficiency

- Structure Principle: Use proximity and visual connection for related, associated elements.
- Well-structured interfaces can be interpreted, used more effectively.

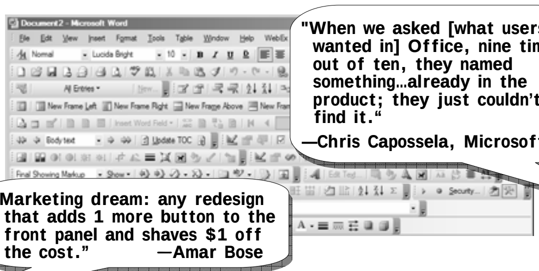






The Cost of Complexity

- Modern user interface architecture is driven by the wants, wishes, and fantasies of management, marketing, developers, and users.



"When we asked [what users wanted in] Office, nine times out of ten, they named something…already in the product; they just couldn't find it."
—Chris Capossela, Microsoft VP

"Marketing dream: any redesign that adds 1 more button to the front panel and shaves $1 off the cost." —Amar Bose

Promote user performance through simplicity!

17

Designing for Success

- Supporting User Performance
- Organizing for User Performance
- Using Instructive interaction
- Reducing Mistakes and Messages
- Eliminating Hidden Inefficiencies

18

Anticipatory Learning

Interfaces can be made self-teaching -
People learn by being told, by being shown, **or** by doing…again and again.
Most learning requires repetition, trial-and-error.
Exceptions: "Sauce Béarnaise" effect*.
Anticipatory learning (the "I-knew-it!" effect):
a form of **single-trial learning** within established prepared learning paradigm.*
❶ **Recognition**: user encounters novel feature,
❷ **Anticipation**: guesses what it does or how it works,
❸ **Action**: tries it, and ❹ **Confirmation**: is rewarded by discovering guess was correct.
To be self-teaching, interfaces must be designed to be
❶ explorable, ❷ intuitable, and ❸ predictable, with ❹ intrinsic guidance.

* Seligman

19

Instructive? Not!

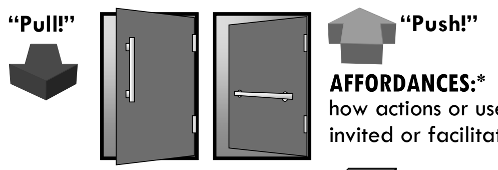
Instructive interaction

- is not based on "real-world" metaphors or silly, simplistic simulations of physical objects.
- does not attempt to analyze and adjust to user behavior or personality.
- does not try to do for users what they can better do for themselves.
- does not make use of universally annoying and usually ineffective pseudo-intelligent agents.

"Interface Hall of Shame" (auf Deutsch) www.frankmahler.de/mshame/ and http://homepage.mac.com/bradster/iarchitect/shame.htm

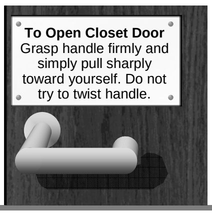
20

Guiding Users

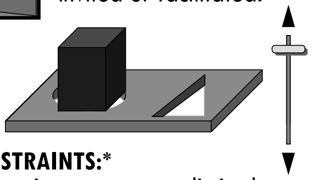


AFFORDANCES:*
how actions or uses are invited or facilitated.

Or, you can get it wrong!



To Open Closet Door
Grasp handle firmly and simply pull sharply toward yourself. Do not try to twist handle.



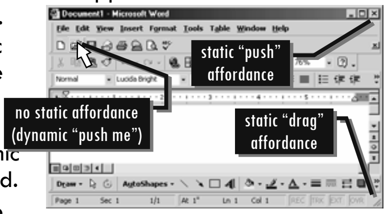
CONSTRAINTS:*
how actions or uses are limited or restricted by physical, semantic, cultural, logical limitations.

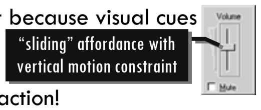
* Norman

21

Visual Guidance

- **Visual affordances** appear to invite or suggest certain actions. Visual constraints appear to limit or control actions.
- Static or dynamic visual cues define affordance or constraint.
- Static and dynamic cues can be mixed.
- Static affordance and constraint are strongest because visual cues are always present without first requiring user action.
- All promote instructive interaction!



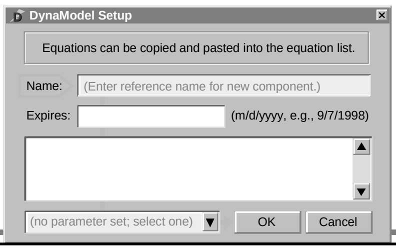


22

Intrinsic Help

- Intrinsic help is built into the interface, not added on.
- It's automatic, no user action is required.
- Can be explicit or implicit, active or passive.

Explicit: ✓ "Getting started" messages and hints.
✓ Embedded prompts. ✓ Examples and templates.
✓ "Tool tips" and pop-ups. ✓ One-time "balloon help."



DynaModel Setup

Equations can be copied and pasted into the equation list.

Name: (Enter reference name for new component.)

Expires: (m/d/yyyy, e.g., 9/7/1998)

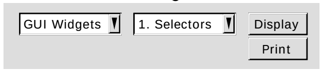
(no parameter set; select one) OK Cancel

Implicit
✓ Starting point highlighted.
✓ Workflow line.
✓ Highlighting (color, shape) of related controls.

23

Progressive Enabling and Disclosure

- Unobtrusively walk users through series of actions.
- Provide an in-context alternative to wizards that is more efficient and promotes skill building.
- Progressive enabling:
- Progressive disclosure exposes facilities as needed.

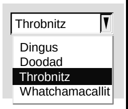


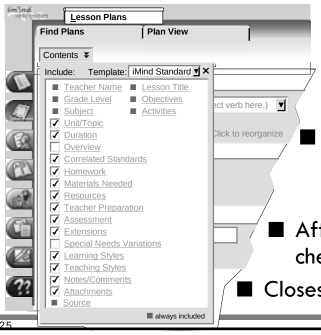
- Progressive enabling and disclosure can cut user errors, such as, in selection or order.
- Use only where required order is both necessary and logical (to users!), as in chapters and sections.
- Either technique can also reduce explorability.
- Progressive disclosure: can be jarring unless done well. (Works best with collections of related controls grouped on drop-down or slide-out panel.)

24

Anticipation

- **Anticipatory action** anticipates probable, necessary, or desirable steps on user's part.
- E.g., first drop-down selection already open.



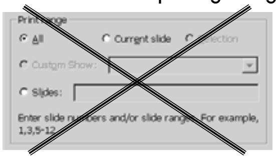


- Anticipatory action can instruct user by exposing options and suggesting possible user actions.
- E.g., novel content navigation control opens automatically when a Lesson Plan opens.
- Affordances (link-like entries, check boxes) invite trial use.
- Closes on "jump" or on blur.

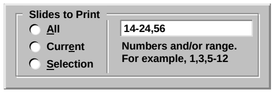
25

Implicit Antecedents

- **Implicit antecedents** avoid imposed order or logic by supplying "missing" steps for the user.
- E.g., clicking a button in a child window that does not have focus implies getting focus and clicking the button.


Print range
All
Current slide
Selection
Custom Show:
Slides:
Enter slide numbers and/or slide ranges. For example, 1,3,5-12

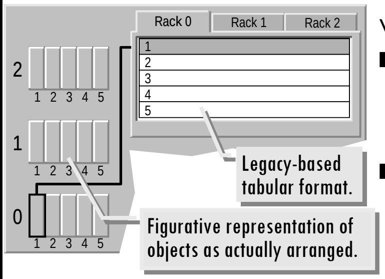
- E.g., here, user entering slide numbers implies selection of the radio button as an antecedent.
- Opportunity for redesign by judiciously violating interaction rules.



- Improved flexibility, efficiency, and ease of learning!

26

Instructive Animation


Rack 0
Rack 1
Rack 2
1
2
3
4
5
2
1
0
1 2 3 4 5
Legacy-based tabular format.
Figurative representation of objects as actually arranged.

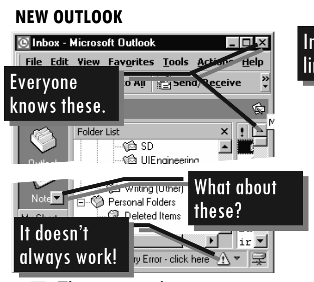
Visual Synchrony -

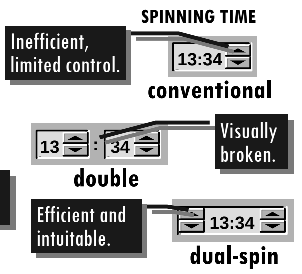
- Two different views of same information in incompatible formats are required.
- User must instantly, correctly understand exactly how related.
- Synchronized views, selection and movement in either.
- Visual link established by colored line.
- Animation dynamically demonstrates exact nature of complicated visual relationship.

27

Familiar Faces, Strange Places

- Standard, well-established controls, visual elements, and interaction idioms can be used in new ways.





- Theme and variation promotes single-trial learning.
- Appeal to familiar without violating expectations.

28

Designing for Success

- Supporting User Performance
- Organizing for User Performance
- Using Instructive interaction
- Reducing Mistakes and Messages
- Eliminating Hidden Inefficiencies

29

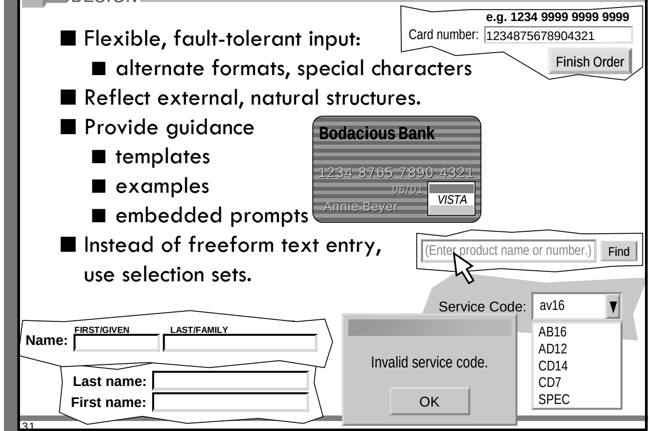
User Errors

- User errors are one of the biggest performance hits.
- Mistakes cost triple:
  - make the mistake
  - discover the mistake
  - correct the mistake
- The most expensive errors are the ones not noticed or never found.
  - lost sales
  - reduced data integrity
  - …
- Prevent errors as much as possible.
- Design tolerant, flexible interfaces.
- Simplify discovery and correction.

30

Error Prevention

- Flexible, fault-tolerant input:
  - alternate formats, special characters
- Reflect external, natural structures.
- Provide guidance
  - templates
  - examples
  - embedded prompts
- Instead of freeform text entry, use selection sets.

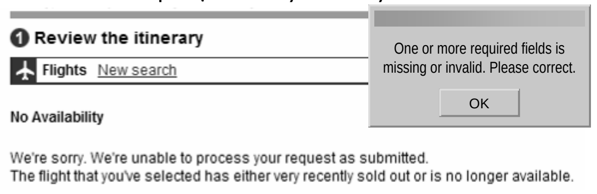


31

Simplified Correction

- Provide notification in context as early as practical.
  - Clearly mark missing or incorrect fields in context.
  - Make it easy for user to get to next error.
- Support edit in place for correction.
- Identify specific cause of problem and how to correct.

Scenario: complex, multi-city itinerary -



1 Review the itinerary

Flights New search

No Availability

We're sorry. We're unable to process your request as submitted.
The flight that you've selected has either very recently sold out or is no longer available.

One or more required fields is missing or invalid. Please correct.

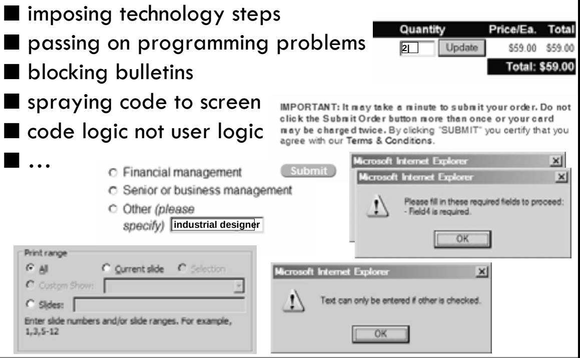
OK

- And never discard or lose or corrupt user-entered data!

32

Inside-Out Design

Technical constraints, coding idiosyncrasies dominate -
- imposing technology steps
- passing on programming problems
- blocking bulletins
- spraying code to screen
- code logic not user logic
- …



Quantity		Price/Ea.	Total
2	Update	$59.00	$59.00
			Total: $59.00

IMPORTANT: It may take a minute to submit your order. Do not click the Submit Order button more than once or your card may be charged twice. By clicking "SUBMIT" you certify that you agree with our Terms & Conditions.

Submit

○ Financial management
○ Senior or business management
○ Other (please specify) industrial designer

Microsoft Internet Explorer
Please fill in these required fields to proceed:
- Field4 is required.
OK

Print range
[✓ All] ○ Current slide ○ Selection
○ Custom Show:
○ Slides:
Enter slide numbers and/or slide ranges. For example, 1,3,5-12

Microsoft Internet Explorer
Text can only be entered if other is checked.
OK

33

Errors, Exceptions, Messages

"Error messages block the progress of work. They indicate failure in the software and by its programmers. Eliminate them all!"*
- Prevent user errors whenever possible.
- Correct or work around user errors within reasonable code. (But not artificial intelligence!)
- Make reasonable assumptions, take appropriate action, report action taken to user where not obvious. (Best shown in context rather than separate message.)
- Relevance, Recognition, and Response: Genuine errors—**relevant** and of interest to user, requiring user **recognition** and user **response**—should be reported by modal messages.
- Remember, modal dialogues (blocking bulletins) stop process until user acknowledges.

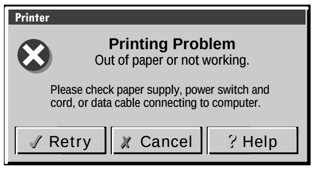
* Alan Cooper

34

Errors, Exceptions, Messages

polite	Do not imply user is wrong, it's the program having a problem.
informative	Say exactly what happened and why, succinctly, in user and domain language.
helpful*	Say what user can or should do to recover.

✓ Introductory and stock phrases are noise, waste time.
✓ Techno-babble and jargon confuse users.
✓ Journalism style is best:
headline,
lead sentence,
story!
✓ Use functional labels, meaningful options.
✓ Provide help whenever possible and appropriate.


Printer
Printing Problem
Out of paper or not working.
Please check paper supply, power switch and cord, or data cable connecting to computer.

* Cooper

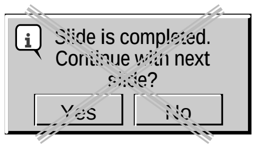
35

Alerts and Confirmations

Alert: Separate modal (sometimes non-modal) message box to inform user of system action, progress, or state.
- Usually obvious from task results, mostly unnecessary.
- Best to inform within interaction context, preferably through feedback intrinsic to task or results.

Confirmation: Modal message box to confirm user action:
- They interrupt progress of work and annoy users.
- They do not work, are ignored or skipped over.
- Often trivial, inconsequential.
- Instead of worthless and annoying confirmation, make all actions reversible!

Almost all are unnecessary!

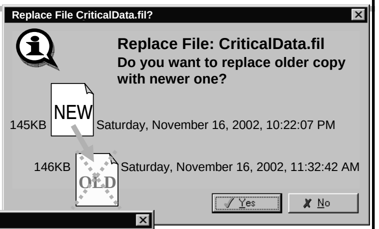

Slide is completed. Continue with next slide?

36

Conditional Confirmation

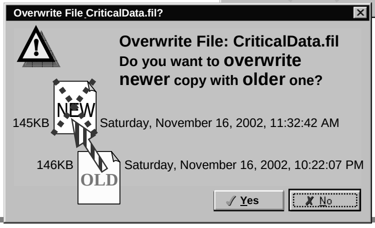
PREVENT ERRORS!

- Highlight key information.
- Emphasize differences between cases.


Replace File CriticalData.fil?

Replace File: CriticalData.fil
Do you want to replace older copy with newer one?

145KB NEW Saturday, November 16, 2002, 10:22:07 PM
146KB OLD Saturday, November 16, 2002, 11:32:42 AM


Overwrite File CriticalData.fil?

Overwrite File: CriticalData.fil
Do you want to overwrite newer copy with older one?

145KB NEW Saturday, November 16, 2002, 11:32:42 AM
146KB OLD Saturday, November 16, 2002, 10:22:07 PM

- Make reasonable assumptions.
- Design based on task cases and analysis of risks!

37

Promoting Reliable Interaction

- Reduce opportunities for mistakes.
- Reduce or eliminate penalties for mistakes.
- Universal, infinite-level undo and redo.
  - (Infinity approximately 12 or less!)
- Promote ease of navigation, exploration, experimentation.
- Do not penalize or punish users for trying things out.
- Communicate clearly with users.
- Remember:
  - unnecessary or confusing messages not only waste time but can cause errors.

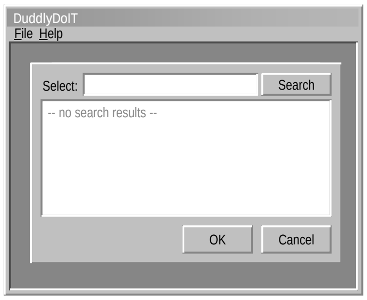
38

Designing for Success

- Supporting User Performance
- Organizing for User Performance
- Using Instructive interaction
- Reducing Mistakes and Messages
- Eliminating Hidden Inefficiencies

39

Workflow Performance Hurdles



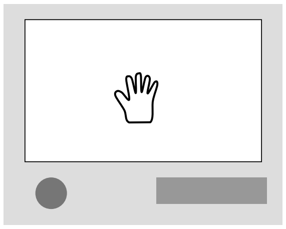
- Users have work to do and just want to get on with it.
- Avoid splash screens, empty space, and other impediments to getting started or getting on with it.
- Even initial dispatch menu or wizard can be a barrier.
- Make reasonable assumptions about most likely starting point, initial state or first user action.
- Offer easy alternatives. Remember preferences.

40

User Performance and Immediacy

- Response time is essential for sense of immediacy and user performance - both efficiency and reliability.
- Essential in Web-deployed applications to identify active objects and provide feedback.
- Intolerable: >500ms
- Slow but tolerable: <500ms
- Desirable: <100ms

(But depends on context.)



41

Component Interaction Design

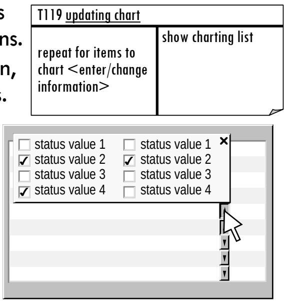
- Interaction design should fit details of the task case and work context in which task is enacted.
- Consider repetitive charting activity.
- Design calls for list of items with embedded drop-downs.
- When a drop-down is open, it blocks view of next items.

SOLUTION

✓ "drop-up" panel

✓ closes automatically on external action

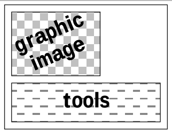
Incremental improvements can add up to dramatically enhanced user performance.

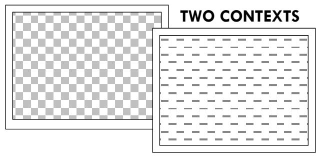


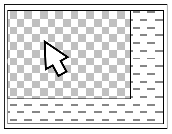
42

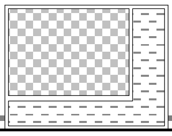
Real Estate and Interaction Design

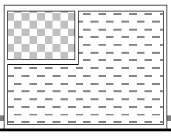
Consider tradeoffs in complex graphic markup and annotation process: image size versus easiest access to most annotation tools.







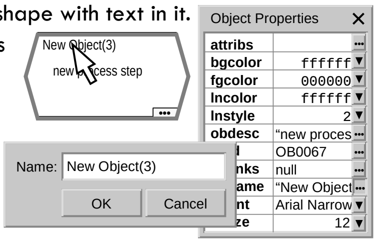




43

Inside-Out or Outside-In Design

- To the programmer, it's an object with properties to be inspected or modified with a property sheet.
- To the user, it's a shape with text in it.
- The user just wants to fix the name and other text.
- Interaction design from perspective of program code slows down and frustrates users.
- Interaction must always be designed from the user perspective, from the outside-inward, supporting "here-and-now" user action, such as edit-in-place.



Object Properties

attribs	
bgcolor	ffffff
fgcolor	000000
lncolor	ffffff
lnstyle	2
obdesc	"new proces
l	OB0067
nks	null
ame	"New Object
nt	Arial Narrow
ze	12

Name: New Object(3)

44

Convenient Design

- Convenience particularly important for frequent, repetitive tasks and for "power users."
- Convenience? "Speed dialing" approximations on the microwave: 88 instead of 130.
- Technology steps: extra actions imposed by the technology, not part of user task perspective.
  E.g., "Recalculate total" or setting quantity to zero on Web forms.
- Use implicit antecedents! Fill in implied but omitted user steps.
- Use default values! Make most likely or logical assumptions, e.g., 100% power.

MY #!%*& MICROWAVE

45

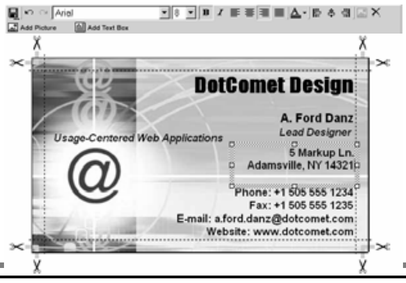
Defaults for Efficiency

- Default values and pre-filled form fields can improve usability by saving time and reducing errors.
- Appropriate choice of default value for parameters and fields include:
  - most likely value
  - last value entered
  - safest value
  - mid-range value
- Inappropriate default values can
  - force users to always enter or change values.
  - increase errors or create hazards.
- Enable user to easily set current values or form contents as default at any time.

46

For User Performance

Take control of presentation and behavior.
With AJAX RIA and SOA, you can do **anything** in a browser instance that you can do in desktop apps.
❶ Design interface right. ❷ Translate to Web.
❸ Adjust design to technology constraints as **necessary**.



- drag-and-drop
- in place editing
- resizable objects
- corral select
- rearrange
- restyle
- insert files
- …

* vistaprint.com
Vista Studio circa 2004

47

Implementing Innovation

- When programmers see the design, the first word out of their mouths is typically…
  "…you can't do that it in .NET (or with AJAX or…)!"
  "…the performance hit will be off the scale."
  "…it won't work."
  "…I hate that kind of thing!"
- Best practice: use programmers who like new problems and respond well to challenges.

Tricks and techniques:
  "Well, go back and think about it."
  "I am confident you can solve it."
  "See how many ways you can find."
- Use the best and brightest for developing custom components.

48