GUI Bloopers

Responsiveness Bloopers

Responsiveness

- Responsiveness not the same as performance or speed
- Highly responsive software
  - Lets you know immediately that your keystrokes, mouse, and clicks were received
  - Estimates how long operations will take
  - Frees you to do other things while waiting
  - Manages queued events intelligently
  - Performs housekeeping and low-priority tasks in the background
  - Anticipates your requests
- Can be highly responsive but slow

Responsiveness Bloopers

- Hard to show screenshots since responsiveness requires a time-lapse capture or movie
- Bloopers are closely related, so they are listed and discussed together instead of one at a time

Bloopers 52-55

- Blooper 52: Cursor doesn't keep up with you; it jumps around and keeps moving after you stop the mouse
- Blooper 53: On-screen buttons acknowledge clicks too late or not at all
- Blooper 54: Menus, sliders, scrollbars lag behind actions, destroying hand-eye coordination for successful operation
- Blooper 55: Moving and sizing operations don't keep up with your actions and don't provide temporary "rubber-band" feedback

Bloopers 56-59

- Blooper 56: Application doesn't indicate that it is busy; it just ignores you
- Blooper 57: Application occasionally – and unpredictably – is unresponsive while it does internal housekeeping
- Blooper 58: Long operations don't display progress
- Blooper 59: Long operations provide no way to cancel

Bloopers 60-63

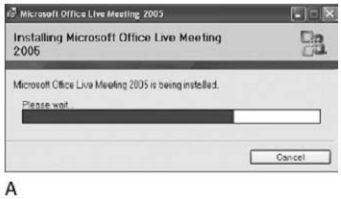
- Blooper 60: Application wastes idle time, and when you finally give a predictable command it takes a long time to finish
- Blooper 61: Application gives no feedback when it hangs, with no indication of what is or is not happening
- Blooper 62: Website has huge images and animations, so it is viewable only with a high-speed Internet connection
- Blooper 63: Website always reloads whole page in response to small edits

Responsiveness Blooper Example : Waiting

- Let the user know when the system is busy

- Without it the user may think the application is locked up, submit data multiple times, kill it

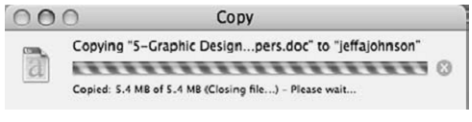
Phony Progress Bars



A

B

False progress bars that fill up repeatedly. (A) Windows Installer. (B) MacOS CD utility.



Apple iDisk file copy: false progress bar, no time estimate, and cancel button is disabled.

Reasons for poor responsiveness

- UI designers rarely consider responsiveness during design
- Programmers don't know how important responsiveness is
- Programmers equate responsiveness with performance
  - e.g. better algorithms or data structures
- Programmers treat user input like machine input
  - Doesn't always have to be processed in the order received
- Developers use simple implementations

Reasons for poor responsiveness

- GUI software tools, components, and platforms are inadequate
  - Normally not the default for a multi-threaded wait cursor
- Managers hire GUI programmers who lack the required skill

Avoiding Responsiveness Bloopers

- Principle 1: Responsiveness is not the same as performance
- Principle 2: Processing resources are always limited
  - Users try to do more as CPU speeds increase
  - Customers probably have slower computers than developers
- Principle 3: The user interface is a real-time interface

Real Time Interface

- 0.1 seconds
  - Limit for perception of cause-and-effect between events
  - Software that waits longer than 0.1 seconds to register a reaction to a user action appears “broken”
  - Limit for perception of smooth animation
- 1 second
  - Maximum comfortable gap in a conversation
  - If displaying information on the screen the user is unlikely to react until at least one second
- 10 seconds
  - Unit of time into which people break down their planning and execution of larger tasks
  - Every ten seconds user like to look up and reassess their task status, relax, etc.
  - Like to mark a task complete and move onto the next one
  - Amount of time a user is willing to spend to set up and operation and start it before losing patience (operation can take longer)

Avoiding Responsiveness Bloopers

- Principle 4: All delays are not equal: software need not do everything immediately
- Principle 5: Software need not do tasks in the order in which they were submitted
- Principle 6: Software need not do everything it was asked to do
  - Sometimes an operation is not necessary; e.g. if told to save but nothing has changed there is no need to waste time re-saving it
  - Queued task may become moot

Avoiding Responsiveness Bloopers

- Principle 7: Human users are not computer programs
  - Cannot sustain high rates of input for very long; can keep the system busy for several seconds but then must pause to think or rest
  - Can multi-task depending on tasks
  - When buttons don't acknowledge a click immediately, users assume they missed and click again

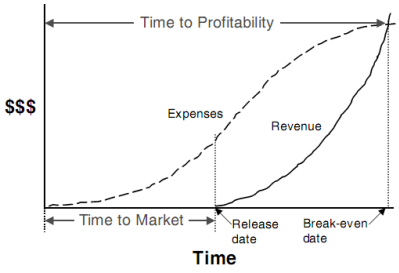
Management Bloopers

Management Bloopers

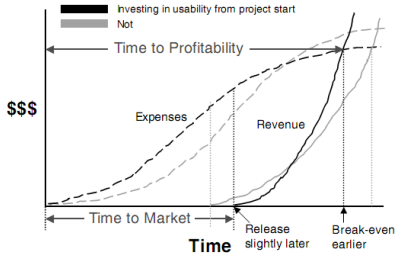
- GUI bloopers are not always the programmers fault; sometimes management is to blame for creating adverse circumstances
- Example: Smooth over problem of the moment
  - Call in a UI consultant with no mandate or resources to correct a flawed process and attitudes
  - “Smearing lipstick on a bulldog”

Blooper 64: Treating UI as a low priority

- Usability often has a lower priority over other tasks
- Variation A: Assuming that usability has low impact on market success by focusing on time to market, not time to profitability



Total costs and revenue over time for a normal software development project.



Total costs and revenue over time with and without early usability investment.

DILBERT © Scott Adams/Dist. by United Feature Syndicate, Inc.

Blooper 64

- Variation B: Assuming that the UI is only “fonts and color”
- Variation C: Assuming that users can adapt to anything
  - Just because people can doesn’t mean they will
- Variation D: Rationalizing
  - UI not a product feature that can be dropped to meet a deadline
- Variation E: Assigning the GUI to junior programmers

Avoiding Blooper 64

- Management should make it a high priority to develop products that have high-quality user interfaces
  - Usability has a powerful impact on the product’s success
  - The UI is about “deep” issues not just fonts and colors
  - Users can adapt to bad UI’s but banking on that is foolish
  - The UI can’t be dropped to meet a schedule or budget constraint
  - Experience matters

Blooper 65 : Misunderstanding what user interface professionals do

- Many people don't know what usability professionals actually do
- GUI programmers are GUI designers?
  - Programmers know how to write code using controls, menus, etc. but not necessarily how to design the interface and in fact can produce bad GUI's
- Graphic designers are GUI designers?
  - Appearance vs. usability

Avoiding Blooper 65

- Know the roles of different designers and programmers

Role	*Skills*
GUI designer	• Task analysis, conceptual design • Interaction design: context, high-level organization, task flow • UI design: input and output • Real-time responsiveness goals • Usability evaluation, usability testing • Assessing conformance to usability standards • Layout
Graphic designer	• Creating recognizable images, intuitive symbols • Production values, aesthetic appeal, brand awareness • Making best use of the available display medium • Conveying function graphically • Layout, visual hierarchy • Visual consistency
GUI programmer	• Dynamic prototypes • Implementing specified design: internal architecture, programming • Knowledge of GUI toolkit • Maximizing performance, meeting real-time goals • Assessing and explaining technical constraints, costs, and risks

Blooper 66: Discounting the value of testing and iterative design

- Some managers don't see the need for usability testing or significant UI revisions
- Variation A: Agile/XP in name only
- Variation B: Good designers don't need iteration
  - Testing and revision best way to reduce risk
- Variation C: We don't have the luxury of usability testing
  - Myths: expensive, skipping testing will save money
- Variation D: Allowing no time to fix usability problems

Avoiding Blooper 66

- UI design is not a mystical art based on innate talent and flashes of creativity but a learned engineering discipline.
  - Industry standards, best practices
  - Scientific basis in human perception, motivation, information processing
  - Need for clear requirements
  - Working with constraints and trade-offs
  - Generation and consideration of design alternatives
  - A need to test, evaluate, and revise

Blooper 67: Anarchic Development

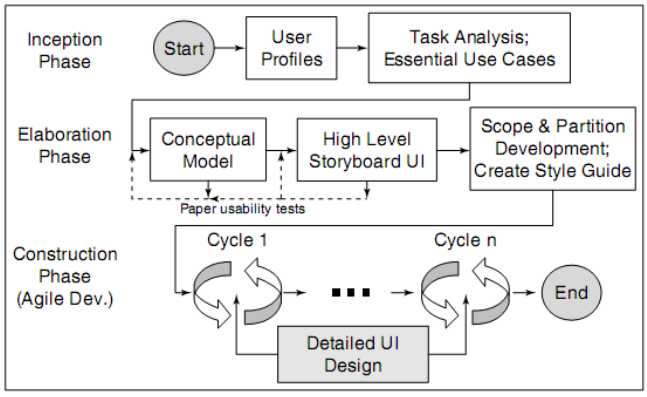
- Anarchic: uncontrolled, unrepeatable, driven by individual whim and crisis of the moment rather than proven, repeatable practices
- Variation A: No design
  - Fooling self that you're doing Agile/XP but not doing weekly scrums, discussing/testing designs for the next cycle, no quick tests, not getting feedback
- Variation B: No standards or guidelines

Blooper 67: Anarchic Development

- Variation C: No oversight
  - "Hire nerds, tell 'em what you want, lock 'em in their offices, and throw in pizza and t-shirts every few weeks"

Avoiding Blooper 67

- User-centered design and Agile/XP coexist



User-centered design with Agile development [adapted from Meads, 2007].

Avoiding Blooper 67

- Quality UI's require investment
  - Training, hiring, developing UI style guides or standards, usability tests, etc.
- Give UI experts more clout
- Take responsibility

Blooper 68: No task expertise on the team

- Projects require someone with a solid understanding of the target task
- Developers may assume they are task-domain experts
- Developers sometimes discount users' task knowledge
- Importing task expertise is hard

Avoiding Blooper 68

- Users' task-domain expertise is a crucial ingredient
  - Key in XP/Agile methods
  - Overcome any organizational obstacles to user involvement
- Use dedicated designers for complex, specialized applications
- Hire dual experts if you can find them

Blooper 69: Using poor tools and building blocks

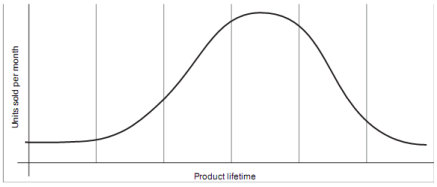
- If using a GUI toolkit, does it really include usability for the GUIs that can be built? Managers and developers focus on:
  - Ease of use
  - How quickly GUIs can be developed
  - How easy the resulting GUI is to maintain
  - Compatibility with existing tools, development process
  - Cost
  - Prior experience with the tool or similar tools

Blooper 69

- Managers should also consider the usability list as factors in adoption of a tool:
  - How compliant are GUIs developed with the tool for standards on the target platform?
  - Are the GUIs developed sufficiently responsive?
  - Does the tool allow appearance details to be fine-tuned to conform to an apps look and feel?
  - How easy GUIs can be internationalized and localized?
  - How accessible GUIs can be to various users such as the disabled?

Blooper 70: Giving programmers the fastest computers

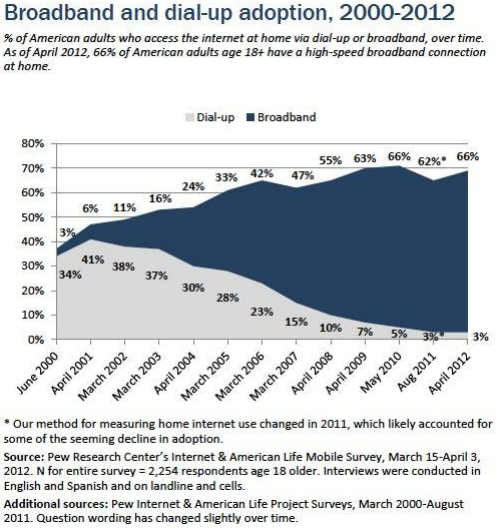
- A cause of responsiveness bloopers
- Programmers' systems will be faster than those of most customers, giving the latest hardware and net connections gets them accustomed to frequent upgrades and view performance/response problems as temporary until "the next upgrade"



Market adoption of new computer models over time.

Net connections

- The masses are behind the technology elite
- Most people finally have broadband though but 30% still have none



* Our method for measuring home internet use changed in 2011, which likely accounted for some of the seeming decline in adoption.

Source: Pew Research Center's Internet & American Life Mobile Survey, March 15-April 3, 2012. N for entire survey = 2,254 respondents age 18 older. Interviews were conducted in English and Spanish and on landline and cells.

Additional sources: Pew Internet & American Life Project Surveys, March 2000-August 2011. Question wording has changed slightly over time.

Avoiding Blooper 70

- Don’t be too quick to upgrade programmers’ computers
- Test on slower computers
- Test on slow network connections
- Compromise: Get developers a fast development machine and a “slow” test machine