Lecture 10: Incremental Releases

Valentin Razmov

Outline

- Why Incremental Releases?
  - Deliverables for Zero-Feature, Alpha/Beta, and Final Releases
  - "The Joel Test: 12 Steps to Better Code"

Resources

- Rapid Development, by Steve McConnell
- Lectures from Winter 2004, by David Socha
- Discussions with James Bullock

Why Incremental Releases?

- To reduce the risks
  - Manage customer expectations
  - Have something shippable / shipped at all times
  - Allow the development team to quickly identify and correct/recover from early failures

- Closely related to evolutionary prototyping, evolutionary delivery, staged delivery, and design-to-schedule

Deliverables: Zero-Feature Release

- Build process, installation process, code repository, automated testing framework, bug tracking system
  - Maybe no tests yet and no tickets in the bug tracking system
  - Installation package
  - Includes all of the items below
  - Demo of one-step build and component communication
    - Checks out all sources from repository, compiles and builds binaries, packages them along with all existing documentation and automated tests, and places the result on a known site ready for downloading
    - Shows that your main components identified in the design can successfully communicate (be integrated) with each other
  - Latest specification, design, and test plan documents
    - Keep them short! Consider what is / isn’t important for customers / devs.
    - Up-to-date schedule
      - Includes what has been done and what remains to be done
  - Release notes
    - Detailed instructions on how to run the demo
    - Known issues with prioritization

Deliverables: Alpha/Beta Releases

- Installation package
  - Application sources and binaries
    - One-step build for all sources
    - Latest spec & design documents
      - Keep it short! Consider what is/isn’t important for customers/devs.
      - Release notes
        - Detailed instructions on how to run a (small) demo of your app
        - Known issues with prioritization, expressed in a bug tracking system
      - Up-to-date test plan
        - Automated tests (unit and acceptance)
        - Up-to-date schedule
          - Including what has been done and what remains to be done

Issues to consider:

- Who is your audience – customers or developers? What do they expect from this release? What defines success for them?
The Joel Test:
12 Steps to Better Code

- Which of the 12 steps will your team follow?
  - Do you use source control? ...........................................
  - Can you make a build in one step? ...............................  
  - Do you make daily builds? ...........................................
  - Do you have a bug database? ......................................
  - Do you fix bugs before writing new code? ......................
  - Do you have an up-to-date schedule? ...........................
  - Do you have a spec? .................................................
  - Do programmers have quiet working conditions? ..........
  - Do you use the best tools money can buy? ....................
  - Do you have testers? ................................................
  - Do new candidates write code during their interview? ....
  - Do you do hallway usability testing? ...........................

15 Jul 2005  CSE440, Summer'05, Lecture 10

The Joel Test:
12 Steps to Better Code

- Are there important missing steps (that you do or need to do, but that aren’t mentioned among the 12 Joel talks about)?

15 Jul 2005  CSE440, Summer'05, Lecture 10