Continuous Integration

C-SPIN 10-Minute Madness

Chris Kaufman

September 3, 2008

Continuous Integration

- What is Continuous Integration
- How does CI work
- How is CI implemented
- How does CI help developers
- How does CI help QA
- Key Practices of Continuous Integration
- How can the benefit of CI be maximized

What is Continuous Integration

- Continuous Integration is a practice of assembling a project's code every time it is changed.

How does CI work

- Developer checks out code
- Developer adds new functionality or fixes defect
  - Developer adds/updates unit tests
  - Developer runs local build/test
  - Developer checks in changes
- CI server detects check in
  - CI server checks out latest code
  - CI server runs build and tests
  - CI server notifies team if build or tests fail

How does CI help developers

- By integrating “early and often” developers know their code works with the rest of the system
- Developers start development from a better code base
- Quick feedback on problems
- No unpredictable, long delays when someone “breaks the build”
- Reinforces unit testing practices

How does CI help QA

- Current software is always available for testing
- Automated unit tests guarantee minimum quality level
- Automated functional/regression/performance tests can be scheduled as part of a daily or periodic build
- QA knows what is in each build

How is CI implemented

- Continuous integration requires four key components
  - An automated build process such as Ant, Make or MSBuild
  - A Source Control Management system such as CVS, Subversion, ClearCase or Perforce
  - A continuous build server such as Cruisecontrol, Apache Gump, Parabuild, Build Forge or Hudson
  - Automated unit tests such as J-Unit or N-Unit

Key Practices of Continuous Integration

- Maintain a code repository
- Automate the build
- Make your build self-testing
- Check in code daily
- Every check-in is built
- Keep the build fast
- Test in a clone of the production environment
- Make it easy to get latest deliverables
- Everyone can see latest build results
- Automate deployment of builds

Maintain a code repository

- Necessary so that multiple developers can work on the same code base
- Allows changes to be tracked
- Allows labeling of good builds so every developer can work from a known good base of code

Automate the build

- An automated build process is the enabling technology for Continuous Integration
- Without an automated build process you can still implement daily check-ins and a daily manual build

Make your build self-testing

- If your build doesn't include unit tests, all you are proving is that the build isn't broken
  - that's valuable,
  - knowing your code works is **more valuable**
- Unit tests (like all regression tests) become more valuable with time
- Beside unit tests you can include other kinds of tests
  - Functional
  - Performance

Check in code daily

- The longer you go without checking in the greater the chance is that your changes won't work with other people's changes
- Daily check-ins prevent accidental code loss
- Daily check-ins promote good coding practices
- Daily check-ins reduce the amount of code that has to be merged.

Every check-in is built

- Provides immediate feedback
  - If the build is broken you know it immediately and can fix it immediately
- Provides confidence that code base is good

Keep the build fast

- To build every check-in the build has to be fast
- Keeping the build fast means keeping code well structured

Test in a clone of the production environment

- This minimizes the chances of masking defects because the test environment is different than production
- Pay attention to
  - Class paths
  - Permissions
  - Database settings
  - Software versions
  - OS versions

Make it easy to get latest deliverables

- Reduce the chance of wasted time testing an "old" version of software
- Tighten the feedback loop from developers to stakeholders. Stakeholders can see what features "look" like.
- Finding issues earlier can reduce rework when features don't meet stakeholder's needs.

Everyone can see latest build results

- IIt should be easy to see:
  - What the latest version of the build is
  - What the latest good build is
  - If the latest build is broken, who and what change broke it.

Automate deployment of builds

- Automating deployment assists in debugging deployment process
  - If you deploy the build every day several times a day you are far less likely to have problems deploying to test or to production.

How Can CI Benefits Be Maximized

- Measure more than just unit test failures
  - Measure number or % of methods with unit test
  - Measure code coverage of tests
  - Measure code complexity
  - Measure code duplication
  - Measure compliance with coding standards
  - Measure code dependencies
- Set quality gates for measures
  - For example
    - build fails if code coverage drops below X%
    - Build fails if cyclomatic code complexity > Y for any method

Tools List

Tool Type	Java Tools	.Net tools
Coverage	Cobertura, Clover	Ncover, Clover.NET
Code Duplication	CPD, Simian	Simian
Code Complexity	JavaNCSS, PMD	CCMetrics, ViL
Coding Standards	Check Style, PMD	FXCop
Code Dependencies	Jdepend	NDepend

Olenick & Associates

205 West Wacker
Suite 2010
Chicago, Il 60606
312-332-0096

Olenick & Associates

www.olenick.com