CSE 70: Software Development Pipeline
Version Control with Subversion,
Continuous Integration with Bamboo,
Issue Tracking with Jira

Ingolf Krueger
Learning Goals for Today
Learning Goals

- Understand the benefits of Continuous Integration

- Know the capabilities of Continuous Integration Servers (example: Bamboo)

- Understand the benefits of Issue Tracking

- Know the capabilities and key workflows of Issue Tracking Systems (example: Jira)
Key elements of the software development pipeline
Stage 4: Small to Medium Teams w/Automated Build Process

Today’s focus: Version Control (subversion) & Continuous Integration (Bamboo) & Issue Tracking (Jira)
What else can you do with subversion?
Other Features of subversion

- **Create Branches**
  - Isolate changes to *separate lines of development*
  - Example: one branch is the currently deployed version for maintenance, another branch is an experimental development version used for testing
  - Creates a versioned copy of the main branch (typically the trunk), otherwise same procedure for checkout, commit, etc.
  - Branches share common path in the revision tree
  - Can merge branches back together

- **Create Tags**
  - “frozen” branches: no further commits
  - *will use this for grading of labs/projects*

- Examine Repository and its History:
  - `svn log`, `svn diff`
  - `svn cat`, `svn list`
Continuous Integration
“A software development practice where members of a team integrate their work frequently, usually each person integrates at least daily – leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.”

¹from: http://martinfowler.com/articles/continuousIntegration.html
CI - Value Proposition

- Reduce risks
- Reduce repetitive manual tasks
- Generate deployable software continually
- Enhance project status visibility
- Increase confidence in the status of the product

adapted from: Paul M. Duvall: Continuous Integration, Addison Wesley, 2007
CI - Practices

• Commit code frequently

• Write automated tests

• Run local/private tests

• Don’t commit code that doesn’t pass your tests

• Fix broken builds immediately

• Avoid checking out broken code

adapted from: Paul M. Duvall: Continuous Integration, Addison Wesley, 2007
CI – Practices (continued)

- Automate builds
- Make builds single-command and execute fast
- Use a version control system with consistent folder structure
- Fail builds fast
- Use dedicated integration machine
- Use a CI server
  (example: Bamboo, Hudson, CruiseControl, ...)

adapted from: Paul M. Duvall: Continuous Integration, Addison Wesley, 2007
CI – Local Build and Integration Cycle

Update

Make Changes

Update

Resolve Conflicts & Run Local Build

Commit

Continuous Integration
CI – Setup/Deployment

Alice
IDE w/local builds
Ant, jUnit

commits

Subversion

commits

Bob
IDE w/local builds
Ant, jUnit
CI – Setup/Deployment

Alice
IDE w/local builds

commits

Subversion
checks out

Continuous Integration Server

Bob
IDE w/local builds

notifies

Bamboo
builds:
compiles
runs tests
logs
post-processes

Ant, jUnit, Jira
notifies
Example: Bamboo Automation

- Checkout from repository
- Clean-up before build is launched
- Building based on triggers (time, commits, cron, ...)
- Launching of build agents (local, remote, cloud) and builders (Ant, Maven, ...)
- Build artifact delivery
- Post-processing
- Notification
By Hand – Run Ant Tasks: Build

**Notional** example for a “build.xml” file using Ant tasks for sending mail, SVN checkout, running JUnit tests, tests, and directory operations.

```xml
<project name = "w3b" basedir = "." default = "build">

    <target name = "build"
        depends = "clean, checkout, init, compile, tests">
        <mail from = "cse70_buildmaster@ucsd.edu"
            tolist = "me"
            subject="Results of nightly build"
            files = "build.log"/>
    </target>

...  
```

1some parts are simplified/left out for readability, but you should get the basic idea...
By Hand – Run Ant Tasks: Checkout

...  
<target name="checkout" depends="clean">  
  <svn username="guest" password="">  
    <checkout url="https://cse70-server.ucsd.edu/svn/max/trunk/"
      destPath="trunk"/>  
  </svn>  
</target>  
...
By Hand – Run Ant Tasks: Tests

...<target name = "tests" depends = "compile">
   <junit printsummary ="yes"
      fork ="yes"
      haltonfailure="yes">
      <formatter type="xml"/>
      <batchtest fork="yes" todir="trunk/reports/">
         <fileset dir="/trunk/src">
            <include name="**/*Test*.java"/>
         </fileset>
      </batchtest>
   </junit>
</target>
...
By Hand – Run Ant Tasks: Compile

...<target name = "compile" depends = "init, checkout">
    <javac srcdir = "trunk/src"
      destdir = "trunk/bin"/>
</target>

<target name = "clean">
    <delete dir= "trunk"/>
</target>

<target name = "init">
    <mkdir dir = "trunk/bin"/>
</target>
</project>
Issue Tracking
Use Case for Issue Tracking: Bug Tracking (1/2)

- User/developer finds a bug
- Creates an “issue” describing the symptoms
- Submits the issue into the system
- The issue gets a unique “ticket” and is assigned to someone to work on it
- The status of the issue (open, solved, retired, ...) is visible
Use Case for Issue Tracking: Bug Tracking (2/2)

• While the issue is open, additional information can be associated with it:
  – relevant code
  – tests
  – builds, ...

• All communication and information is associated with the ticket

• Productivity views: which projects solve issues quickly? who solves issues? what projects have the fewest issues? Why?
Issue Tracking: Other Applications

- Customer Support/Helpdesk
- Requirements Management
  - Feature Lists
  - Feature Requests
  - Voting
- Project Management
  - Task assignment
  - Task tracking
- Workflow Management
Example: Jira

- Project
- Component
- Version
- Issue

read: a Project has many Components
Example: Properties of Issues in Jira

- Comments
- Priority
- Status
- Type
- Key
- Reporter
- Assignee
- Components
- Versions
- Estimates
- Time Spent
- Description
Jira: Home Screen
Jira: Create Issue (1/2)
Jira: Create Issue (2/2)

- **Reporter:** ikrueger
  
  Start typing to get a list of possible matches.

- **Environment:**

  For example operating system, software platform and/or hardware specifications (include as appropriate for the issue).

- **Description:**

  The maximum file upload size is 10.00 MB. Please zip files larger than this.

- **Attachment:**

  Browse...
## Jira: New Issue Example

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>CSE70 Class</td>
</tr>
<tr>
<td>Issue Type</td>
<td>Task</td>
</tr>
<tr>
<td>Summary</td>
<td>Develop Quiz for Week 3</td>
</tr>
<tr>
<td>Priority</td>
<td>Critical</td>
</tr>
<tr>
<td>Due Date</td>
<td>13/Apr/09</td>
</tr>
<tr>
<td>Component/s</td>
<td>Unknown mmenarini</td>
</tr>
<tr>
<td>Affects Version/s</td>
<td>Unknown</td>
</tr>
<tr>
<td>Fix Version/s</td>
<td>Unknown</td>
</tr>
<tr>
<td>Assignee</td>
<td>Ingolf Krueger</td>
</tr>
<tr>
<td>Reporter</td>
<td>ikruenger</td>
</tr>
<tr>
<td>Environment</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Must include something on build process</td>
</tr>
</tbody>
</table>
Jira: View Issue

CSE70 Class

Develop Quiz for Week 3

Created: Today 04:53 PM  Updated: Today 04:53 PM  Due: 13/Apr/09

Component/s: None
Affects Version/s: None
Fix Version/s: None

Description
Must include something on build process.

All  Comments  Change History  Subversion Commits
There are no comments yet on this issue.

© Ingolf Krueger, 2010  CSE  31
Jira: Workflow Actions

Available Workflow Actions

- Start Progress
- Resolve Issue
- Close Issue

Operations

- Assign this issue
- Attach file to this issue
- Attach screenshot to this issue
- Clone this issue
- Comment on this issue
- Create sub-task
- Delete this issue
- Edit this issue
- Move this issue
- Convert to sub-task

Voting:
You cannot vote for an issue you have reported.

Watching:
You are not watching this issue. Watch it to be notified of changes
Jira: Basic Issue States and Actions
Jira: Advanced Issue States and Actions

What have you learned today?
Learning Goals

- Understand the benefits of Continuous Integration

- Know the capabilities of Continuous Integration Servers (example: Bamboo)

- Understand the benefits of Issue Tracking

- Know the capabilities and key workflows of Issue Tracking Systems (example: Jira)
We have now seen all components of the Software Development Pipeline