

psybergate

Continuous Integration Applied



psybergate

Agenda

- What is Continuous Integration?
- Continuous Integration in Context
- Realising Continuous Integration
- Continuous Integration In Practice

What is Continuous Integration?

Integration is hard

- The process of integrating (the various pieces of) software
 - is not a new problem
 - is hard! (but you already knew that 😊)
- Effort increases exponentially with an increase in the
 - number of components/modules
 - number of bugs
 - time since last integration
 - number of team members
- Intent to replace big (and long) integration phases with small and frequent ones
 - nearly instantaneous

What is Continuous Integration?

Rather what it is not?

- **Continuous!**
 - It's really refers to an activity that is frequent or continual
- **Dependent on the adoption of Agile Development**
 - although it enables such an approach
- **A tool or set of tools**
 - although tools are required to implement it
- **Difficult**
 - hey, I can do it!
- **NEW!**
 - sort of doing it for years without knowing it!

What is Continuous Integration?

In the context of agile discipline

- Agile development aims to amplify the effects of standard development practices like
 - Testing through Test Driven Development (TDD)
 - Quality Assurance through Pair Programming and Continuous Integration (CI)
- “If something is hard, do it more often”

What is Continuous Integration?

Straight from the source

'Continuous Integration is 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. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.'

Martin Fowler

What is Continuous Integration?

One line definition

Integrate
working software
early and often

Continuous Integration in Context

Traditional Software Development

- Different modules are developed by different programmers
 - Divide work to allow parallel development
 - Integrate modules at late state of project
 - Integration problems can easily knock project off schedule... or cause it to fail altogether

*The earlier you can detect problems,
the easier it is to resolve them*

Continuous Integration in Context

Agile Software Development

- Any programmer can change any part of the code (collective code ownership)
 - Module clients can change interfaces to suit their needs
 - SCM tools detect conflicting commits
 - Parallel commits can cause *semantic incompatibilities*

*The earlier you can detect problems,
the easier it is to resolve them*

Continuous Integration in Context

Anatomy of a Continuous Integration Solution

- **A central repository for all members of a team, containing**
 - the latest code (at least)
 - the latest executables
- **An automated process for building AND testing all project assets (not just compiling) (does not need to be completely automated)**
 - that can be run many times a day
 - and that is self fulfilling
- **May include:**
 - Continuous Integration tool (hudson, cruise control, bamboo)
 - Testing tools (junit, selenium, jbehave, easyb, cucumber, twist)
 - Bug tracking tools (jira, trac, bugzilla)
 - Project management tools (xplanner, mingle, version one)
 - Quality assurance tools (checkstyle, sonar, structure101)



Continuous Integration in Context

Typical developer workflow

1. Developers run private builds in their own workspaces to ensure their changes don't break the integration build
2. Developers commit their code to a version control repository *at least once a day (ideally after every meaningful increment of work)*
3. Integration builds occur several times a day on a separate build machine
4. 100% of tests (unit, integration, load, functional, etc.) must pass for every build
5. A product is generated (e.g., WAR, assembly, executable, etc.) that can be functionally tested
6. Fixing broken builds is of the highest priority
7. Review reports generated by the build (coding standards, etc.) to seek areas for improvement

Continuous Integration in Context

Benefits of Continuous Integration

- Rapid feedback on
 - build breakages
 - defects introduced
 - quality control
- Finding defects (because of integration or oversight) is easier
 - builds are self-testing
 - defect are detected earlier
 - defects do not accumulate
 - small integration failures are easier to diagnose than large ones
- Helps flush out risk earlier in lifecycle while there is more time to respond
- Encourages more iterative development practices including more frequent releases
- Developers concentrate on coding

Realising Continuous Integration

Hudson – what a tool ☺

- Open-source CI server written in java by Kohsuke Kawaguchi (<http://hudson-ci.org>)
- Emphasis on ease of installation and use
 - “java -jar hudson.war” command line execution
 - Test drive via webstart (<https://hudson.dev.java.net/hudson.jnlp>)
 - Or your OS-specific package
 - Configure everything from browsers
- Extensibility
 - 230+ community-developed public plugins by 230+ contributors
- Great support
- Estimated 13,000 installations, adopted almost “everywhere”
- Community releases every week
 - Over 340 releases to date
 - Rapid new features & bug fixes delivery

Realising Continuous Integration

Basic feature set in Hudson

- Hudson is configured to monitor repository changes
- Check out the source code
 - CVS, Subversion, Git, ClearCase, Mercurial, Accurev, Perforce, StarTeam, ...
- Do builds
 - Ant, Maven, NAnt, MSBuild, Rake, GAnt, Gradle, Shell script, ...
- Record and publish results
- Close the feedback loop (build can be successful, unstable, failed)
 - RSS feeds
 - E-mail/IM notification
 - Tray application to notify developers
- Tool integration
 - Findbugs, cobertura, IDE plugins, ...

Realising Continuous Integration

Hudson's Features in depth

- Every commit triggers a build
 - Commit at least daily (*ideally after every meaningful increment of work*)
 - A clean, full build on an independent “empty” integration machine
 - Triggered by SCM polling, cron-like schedulers, SCM post commit notifications
- Supports self testing builds
 - Directly from source to running tests (no manual intervention necessary)
- Record and publish build results
 - xUnit reports
 - code coverage reports
 - open tasks
- Testing on a clone of the production machine
 - SCP plugin supports copying files to a remote machine
 - VMWare, VirtualBox plugins support managing of virtual instances
 - Deployment plugins support remote java (WAR) deployment

Realising Continuous Integration

Hudson's features in depth



















- Everybody sees what's happening
 - Easy to use web based interface
 - Notifications via email, IM, RSS
 - IDE plugins (Eclipse, IntelliJ)
- Artifact publishing – define the artifacts to publish so that anyone can get them easily
 - WAR/JAR archives
 - Installers
 - Documentation,
 - etc.
- Keep the build fast
 - Supports pipelined builds to split builds across
 - architectural/component boundaries
 - testing boundaries
 - release boundaries
 - Supports clustered builds
- And much, much more...

Realising Continuous Integration

Matrix projects in Hudson

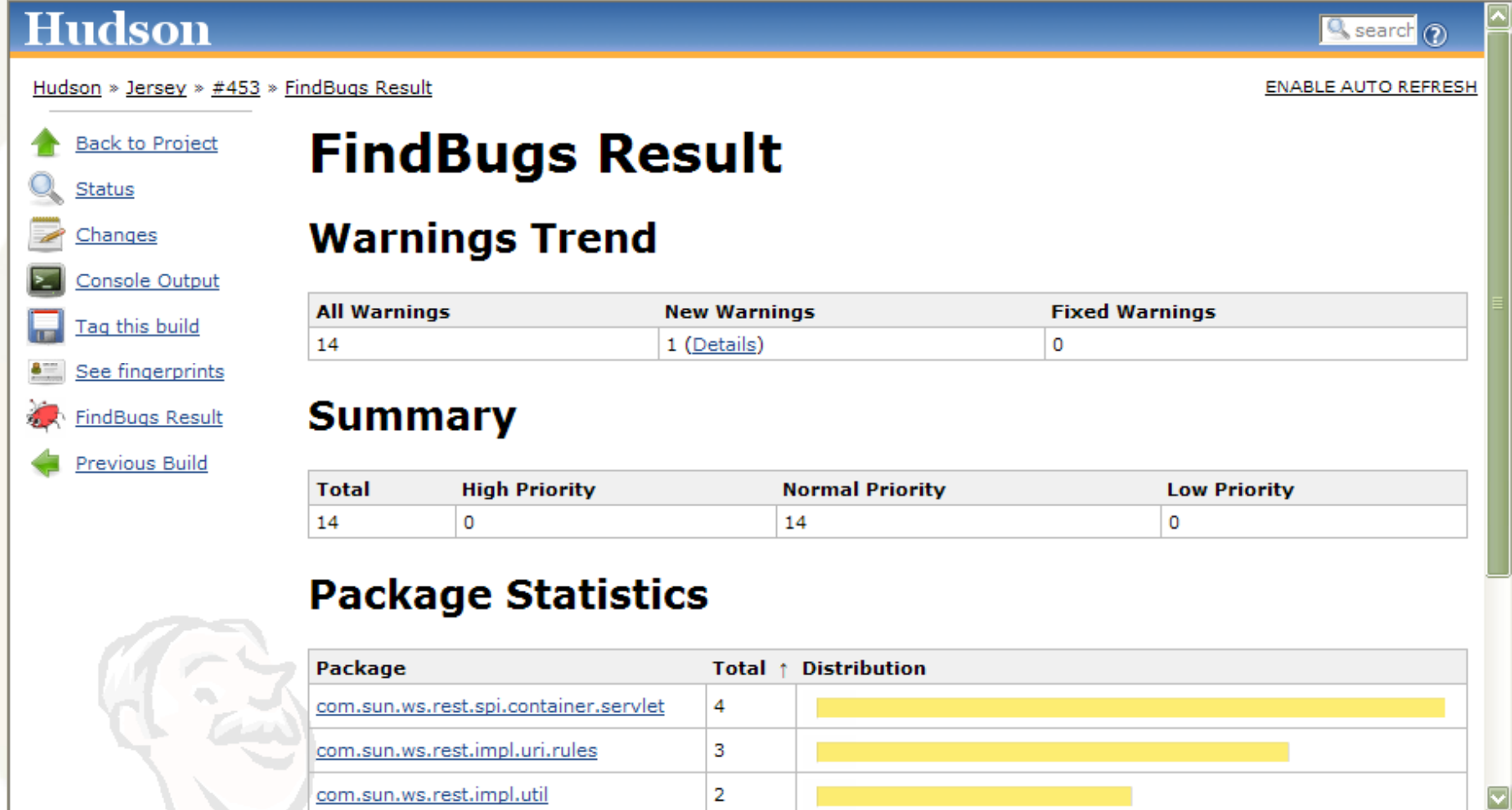
- Run the same thing on different environments
 - such as different JDKs, different databases, and different OSes
- Aggregate the results

Configuration Matrix

		JDK6.0	JDK5.0_12
jaxws-ri-2.1.2	linux		
	windows		
	solaris		
jaxws-ri-2.1.3	linux		
	windows		
	solaris		
jaxws-ri-2.2	linux		
	windows		
	solaris		

Realising Continuous Integration

Track bugs using Hudson



The screenshot shows the Hudson web interface for a specific build. The breadcrumb trail is "Hudson » Jersey » #453 » FindBugs Result". The main heading is "FindBugs Result". Below it is a "Warnings Trend" table showing 14 All Warnings, 1 New Warning (with a link to details), and 0 Fixed Warnings. A "Summary" table shows 14 Total warnings, 0 High Priority, 14 Normal Priority, and 0 Low Priority. The "Package Statistics" table lists three packages: com.sun.ws.rest.spi.container.servlet (4), com.sun.ws.rest.impl.uri.rules (3), and com.sun.ws.rest.impl.util (2). The interface includes a sidebar with navigation links like "Back to Project", "Status", "Changes", "Console Output", "Tag this build", "See fingerprints", "FindBugs Result", and "Previous Build". A search bar and "ENABLE AUTO REFRESH" link are in the top right. A watermark of a man's face is visible in the bottom left.

Hudson

Hudson » Jersey » #453 » FindBugs Result ENABLE AUTO REFRESH

[Back to Project](#)
[Status](#)
[Changes](#)
[Console Output](#)
[Tag this build](#)
[See fingerprints](#)
[FindBugs Result](#)
[Previous Build](#)

FindBugs Result

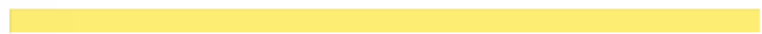
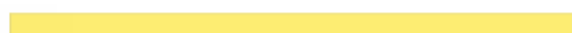
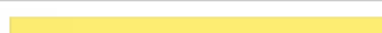
Warnings Trend

All Warnings	New Warnings	Fixed Warnings
14	1 (Details)	0

Summary

Total	High Priority	Normal Priority	Low Priority
14	0	14	0

Package Statistics

Package	Total	Distribution
com.sun.ws.rest.spi.container.servlet	4	
com.sun.ws.rest.impl.uri.rules	3	
com.sun.ws.rest.impl.util	2	

Realising Continuous Integration

Tracking changes using Hudson

[#400 \(Nov 23, 2007 9:39:24 AM\)](#)

1. Removed print statements — [Paul Sandoz / detail](#)

[#399 \(Nov 23, 2007 9:30:35 AM\)](#)

1. Fixed really stupid bug in annotation checking. — [Paul Sandoz / detail](#)

[#396 \(Nov 23, 2007 7:06:10 AM\)](#)

1. Silly bug, the base URI was not set. — [Paul Sandoz / detail](#)

[#395 \(Nov 23, 2007 6:29:21 AM\)](#)

1. When creating a container ignore any container providers that cannot be loaded. — [Paul Sandoz / detail](#)

[#394 \(Nov 23, 2007 6:06:34 AM\)](#)

1. Throw meaningful exceptions when resource class scanning errors occur. This important to for tracking down deployment errors. Log a warning for a path that is not a directory or a jar file. — [Paul Sandoz / detail](#)

[#379 \(Nov 21, 2007 10:55:06 AM\)](#)

1. Updated EntityProvider sample to show support for other Java types, added README. — [Marc Hadley / detail](#)

[#377 \(Nov 21, 2007 10:05:15 AM\)](#)

1. Fixed a bug where premature decoding of the message body could lead to errors in parsing. Escaped = and & characters were not handled correctly. — [Marc Hadley / detail](#)

[#362 \(Nov 16, 2007 9:02:37 AM\)](#)

1. Refactored to remove BaseResourceClass and RootResourceClass. This are no longer needed now that the rules have been moved to the compiler. — [Paul Sandoz / detail](#)

[#361 \(Nov 16, 2007 8:57:22 AM\)](#)

1. Refactored to remove BaseResourceClass and RootResourceClass. This are no longer needed now that the rules have been moved to the compiler. — [Paul Sandoz / detail](#)

Realising Continuous Integration

Build promotion using Hudson

- CI produces a lot of “successful” builds
 - Often overwhelming to downstream consumers
 - Helps in partitioned teams following strict Development and QA processes
- Run tests as fast as you can
- If a build passes tests, promote it

Promotions

 **Good for jax-ws**

Qualification (promoted 12 days ago — 45 minutes after build)

Downstream builds succeeded: [jaxb-sqe-ri-2.1 #356](#) [jaxb-tck-ri-2.1 #346](#) [jaxb-unit-test-2.1 #444](#)
[jaxws-2.1.x-bleedingedge #203](#)

Status

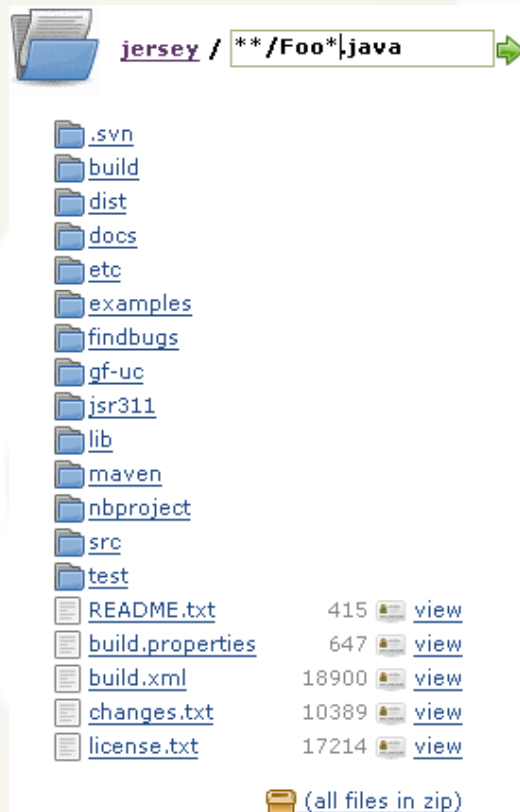
Successfully promoted ([record](#))

Re-execute promotion

Realising Continuous Integration

Other features in Hudson

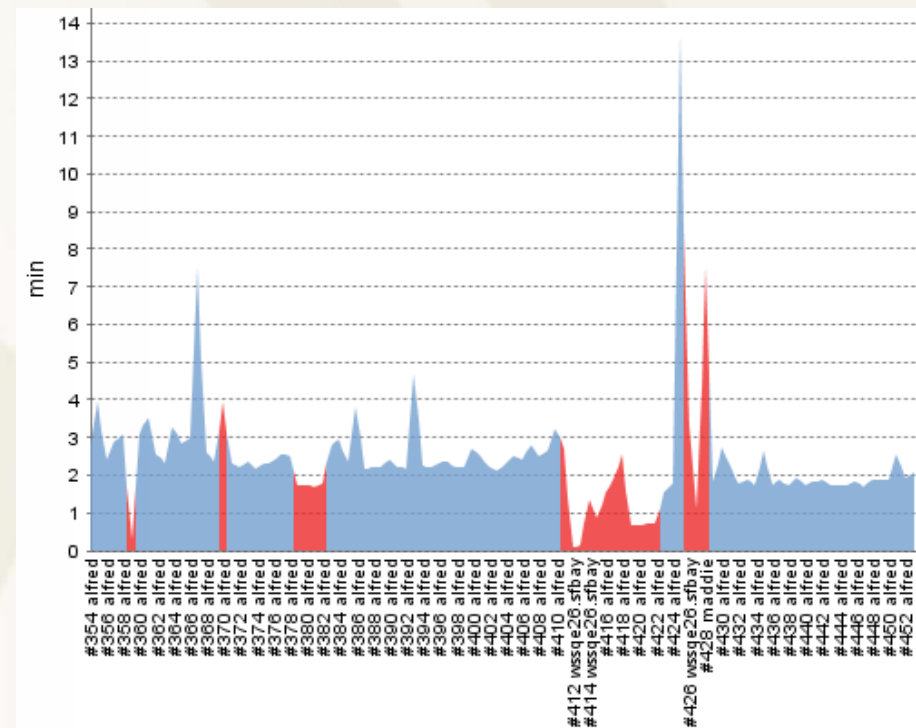
- Browse workspaces
- Build time trend report



jersey / →










- .svn
- build
- dist
- docs
- etc
- examples
- findbugs
- gf-uc
- jsr311
- lib
- maven
- nbproject
- src
- test
- README.txt 415 [view](#)
- build.properties 647 [view](#)
- build.xml 18900 [view](#)
- changes.txt 10389 [view](#)
- license.txt 17214 [view](#)

[\(all files in zip\)](#)



Realising Continuous Integration

Quality Measurement using Sonar – Dashboard

Rules compliance	Project	Code coverage ▲	Build time	Links
90.1%	ceds-parent	0.0%	09.10.2009	  
78.0%	wits-alp	57.8%	10.10.2009	
98.5%	aidsmis-parent	68.8% ▼	08.09.2009	  
77.9%	wits-alp-maven Maven Webapp	71.7%	29.07.2009	
94.0%	crm-batch	88.3%	08.10.2009	

 [Alerts feed](#)



Size

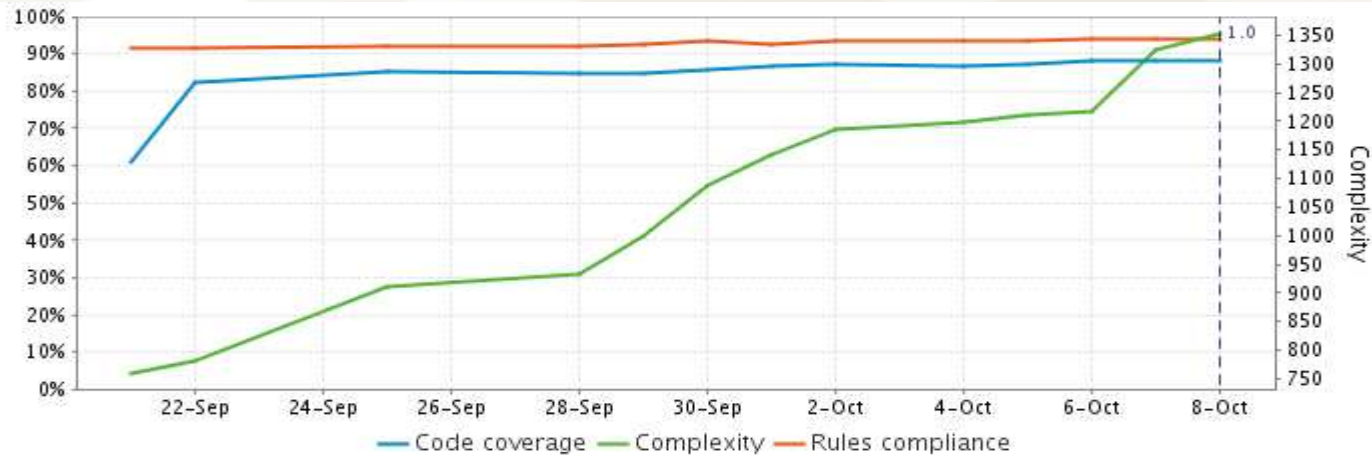
Complexity

Color 0%  100%

Code coverage

Realising Continuous Integration

Quality Measurement using Sonar – Time machine



Show date	Sep. 21, 2009 hide	Oct. 8, 2009 Version 1.0 hide	
Complexity			
<input type="checkbox"/> Complexity/class	8.6	8.2	
<input type="checkbox"/> Complexity/method	1.3	1.4	
<input checked="" type="checkbox"/> Complexity	761	1'353	
Documentation			
<input type="checkbox"/> Comments (%)	2.4%	2.0%	
<input type="checkbox"/> Comment lines	57	80	
Rules			
<input checked="" type="checkbox"/> Rules compliance	91.4%	94.0%	
<input type="checkbox"/> Rules compliance, incl. opt.	0.0%	0.0%	
<input type="checkbox"/> Opt. rules compliance	0.0%	0.0%	
<input type="checkbox"/> Violations	65	80	
<input type="checkbox"/> Opt. rules violations	3'815	6'986	
<input type="checkbox"/> Violations, incl. opt. rules	3'880	7'066	

Realising Continuous Integration

Quality Measurement using Sonar – Code coverage

test.jdbc.datasource	60.0%	UpdateApplicantStudentNumberDao	62.5%
za.ac.wits.crm.reader	65.4%	AbstractIWitsDao	81.5%
za.ac.wits.crm.connector	65.8%	UpdateEnquirerDao	85.7%
za.ac.wits.crm.storedproc	78.7%	CreateApplicantDao	85.7%
za.ac.wits.crm.dao	83.1%	GetStudentNoDao	88.9%
za.ac.wits.crm.processor	87.1%	AbstractIWitsDao\$4	100.0%

CreateApplicantDao

Lines of code: 10
Methods: 4
Duplicated lines: -

Complexity: 4
Complexity/method: 1.0

Code coverage: 85.7%

Violations: 0
Opt. rules violations: 19

Sources Code coverage Violations

```
1 package za.ac.wits.crm.dao;
2
3 import org.springframework.util.Assert;
4
5 import za.ac.wits.crm.domain.Message;
6 import za.ac.wits.crm.storedproc.AbstractIWitsStoredProcedure;
7
8 public class CreateApplicantDao extends AbstractIWitsDao<Message> {
9
10     private AbstractIWitsStoredProcedure<Message> storedProcedure;
11
12     public void init() {
13         Assert.notNull(storedProcedure);
14     }
15
16     @Override
17     protected Message doCreate(Message message) {
18         return executeStoredProcedure(message);
19     }
20
21     protected Message executeStoredProcedure(Message message) {
22         return storedProcedure.execute(message);
23     }
24
25     public void setStoredProcedure(
26         AbstractIWitsStoredProcedure<Message> storedProcedure) {
27         this.storedProcedure = storedProcedure;
28     }
29
30 }
```

Realising Continuous Integration

Quality Measurement using Sonar – Code violations

test.jdbc.datasource	60.0%	UpdateApplicantStudentNumberDao	62.5%
za.ac.wits.crm.reader	65.4%	AbstractIWitsDao	81.5%
za.ac.wits.crm.connector	65.8%	UpdateEnquirerDao	85.7%
za.ac.wits.crm.storedproc	78.7%	CreateApplicantDao	85.7%
za.ac.wits.crm.dao	83.1%	GetStudentNoDao	88.9%
za.ac.wits.crm.processor	87.1%	AbstractIWitsDao\$4	100.0%

CreateApplicantDao

Lines of code: 10
Methods: 4
Duplicated lines: -

Complexity: 4
Complexity/method: 1.0

Code coverage: 85.7%

Violations: 0
Opt. rules violations: 19

Sources Code coverage Violations

1	package za.ac.wits.crm.dao;	
2		
3	import org.springframework.util.Assert;	
4		
5	import za.ac.wits.crm.domain.Message;	
6	import za.ac.wits.crm.storedproc.AbstractIWitsStoredProcedure;	
7		
8	public class CreateApplicantDao extends AbstractIWitsDao<Message> {	
9		
10	1 private AbstractIWitsStoredProcedure<Message> storedProcedure;	Tab Character : Line contains a tab character.
11		
12	2 public void init() {	Tab Character : Line contains a tab character.
		Design For Extension : Method 'init' is not designed for extension - needs to be abstract, final or empty.
13	1 Assert.notNull(storedProcedure);	Tab Character : Line contains a tab character.
14	1 }	Tab Character : Line contains a tab character.
15		
16	2 @Override	Tab Character : Line contains a tab character.
		Design For Extension : Method 'doCreate' is not designed for extension - needs to be abstract, final or empty.
17	1 protected Message doCreate(Message message) {	Tab Character : Line contains a tab character.
18	1 return executeStoredProcedure(message);	

Realising Continuous Integration

Other CI tools

- Cruise Control (<http://cruisecontrol.sourceforge.net>) (the first)
- Apache Continuum (<http://continuum.apache.org>)
- Atlassian Bamboo (<http://www.atlassian.com/software/bamboo>)

Continuous Integration in Practice

Adoption

**Continuous Integration is only one aspect
of an overall process,
for it to work best, you need to:**

- Plan iteratively
 - schedule regular releases with evolving levels of functionality
- Implement iteratively
 - identify and implement small work tasks
 - refactor if necessary!
- Report proactively
 - identify exactly the contents of any build
 - automate reports

Continuous Integration in Practice

Tips

- Define what it means for your build to be successful?
 - When it compiles?
 - When all the unit-tests have run?
 - When all the integration-tests have run?
 - When all the functional-tests have run?
 - When it has been deployed?
- Remember, every build failure is a success
 - You have exposed a potential problem – early!
- Raise build visibility by using something like the Ambient Orb™
- Scale up Continuous Integration by pipelining and splitting along architectural boundaries

Continuous Integration in Practice

Policies and Practices

- Continuous Integration supports team communication, it does not replace it
- Supported by policies and practices
 - Check in frequently
 - Don't check in broken code
 - Don't check in untested code
 - Don't check in when the build is broken
 - Don't go home after checking in until the system builds
 - ...
- Peer pressure within team ensures policies are followed
- No need to enforce policies from on high

Summary

What have we learned?

- Continuous Integration is a variation on an old theme
- It works best with an incremental approach to software development
- CI provides rapid feedback on quality control
- The sky is the limit with regards to what can be achieved once basic automation has been achieved
- It makes software development fun!

Question

What prevents
you from working
iteratively?

Questions



References

- Continuous Integration (according to Martin Fowler) (<http://www.martinfowler.com/articles/continuousIntegration.html>)
- Agile SCM: Realising Continuous Integration, Kevin Lee
- Continuous Integration – Improving Software Quality and Reducing Risk, Paul M Duvall