

Desarrollo y Pruebas de Proyectos Java en un Entorno Ágil

Acerca de Mi

- Amante del Software
- Expatriado y Retornado
- 10 años peleando con líneas de código
- Blogger aficionado: <http://brigomp.blogspot.com>
- Co-fundador de Jobsket
- <http://www.jobsket.es/cv/martin>

El índice

- Historietas de Guerra
- Nociones básicas sobre Desarrollo Ágil
- Buenas Prácticas
- Testing y QA
- Integración Continua

Historias de Guerra



300

Historietas de Guerra

- Iteraciones frecuentes
- Desplegar software rápidamente
- Hacer que el cliente participe en el diseño
- No hace falta rodearse de los mejores.
- Rodearse de gente trabajadora suele ser suficiente.

AQUOS

TVODD: TV Oriented Based Development
ESTAS RETRASANDO EL PROYECTO

SHARP

Historias de Guerra

- Las televisiones no motivan (a no ser que te gusten apostar a las carreras de caballos)
- Determinadas acciones son golpes a la moral
- Los proyectos retrasados no van a remontar el vuelo por poner más presión a los empleados.
- Demuestran que son necesarios cambios en la forma de trabajar de la empresa.



iii A TESTEAR !!!

Historias de Guerra

- No por tener más testers tu software va a estar mejor testeado.
- Es necesaria una estrategia de automatización.
- El equipo de desarrollo debe ayudar al equipo de testers.
- Pero El equipo de testers debe dejarse ayudar.
- Tener testers no informáticos es bueno.

Nociones básicas (y muy breves) sobre Desarrollo Ágil

Manifiesto Ágil

Nuevos Valores

Individuos e integración

Software que funciona

Colaboración con el cliente

Respuesta al cambio

Antiguos Valores

Procesos y herramientas

Documentación exhaustiva

Negociación contractual

Seguimiento de un plan

Desarrollo ágil

- Pragmatismo
- Iteraciones cortas
- Software funcional disponible frecuentemente
- Reuniones frecuentes
- Reparto de conocimiento y responsabilidades
- Documentación ágil
- Demostración continua
- Retrospectivas
- Personas y relaciones frente a herramientas

Buenas Prácticas

Programación en Parejas

- Juntar dos desarrolladores frente a una pantalla.
- A la larga, mayor productividad.
- Incrementa la propiedad colectiva del código.
- Combinación perfecta: desarrollador experto + desarrollador normal/junior.
- Juntar a 2 juniors, 2 expertos no aporta demasiado
- El programador inexperto aprende mucho más.
- Quizás no una práctica para el día a día pero sí para hacer de cuando en cuando.

Revisiones de Código

- Muy útil para detectar errores.
- Muy importante con nuevas incorporaciones en los equipos.
- Es necesario alternar la responsabilidad. Si no, “el revisor” será un cuello de botella.
- Aumenta la responsabilidad colectiva del código.
- Centrarse en errores de alto nivel. El resto se puede detectar fácilmente con análisis estático.

Análisis estático

- Las herramientas de análisis estático detectan errores sin tener que ejecutar código.
- Se integran fácilmente en el proceso de build
 - Si se detectan X errores críticos la build falla.
- Algunas veces hay que mitigar el ruido
- Findbugs, PMD, Sonar.
- Alternativas comerciales

Análisis estático: Errores Imprevisibles

- Visto en Eclipse 3.5RC2
 - if (adapters == null && adapters.length == 0) return;
- Error desde Eclipse 3.2
 - Probablemente nunca se cumplió la condición
 - Si se cumpliera habría saltado una NPE

Análisis estático: Seguridad

- Ejemplo típico. SQL Injection

```
HttpServletRequest request = ...
```

```
String usuario = request.getParameter("user")
```

```
String query = "select * from account_numbers where nombre='  
" + usuario + "'";
```

```
con.execute(query)
```

- ¿Y si user es " ' OR 1==1-- " ?

Análisis estático: Escalabilidad

- Resource leaking

```
try {
```

```
    Connection c = ...
```

```
    String query = ...;
```

```
    c.execute(query);
```

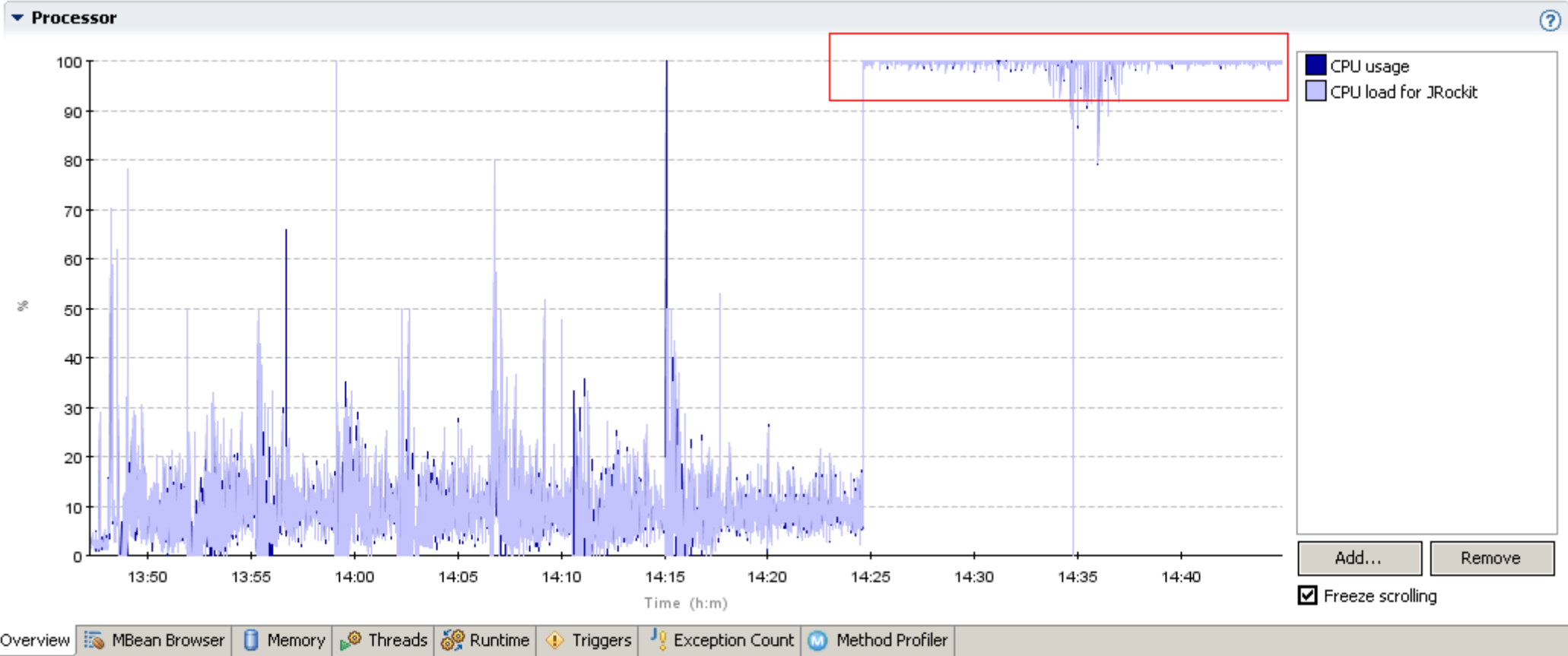
```
    c.close()
```

```
} catch (Exception e) { e.printStackTrace() }
```

- ¿ Qué pasa si falla la consulta ?

Análisis estático: Concurrencia

- Ejemplo, uso de estructuras de datos no Thread-safe en código multihilo
- ¿Inocente? Esto es un gráfico de un servidor real. 4 CPUs 8Gb RAM.



Análisis estático: Concurrencia

- Parece grave, ¿no? ¿Cuál fue la causa?
- Dos llamadas a `HashMap.put()` y mala suerte

FindBugs: jobsket crawlers

Fichero Editar View Navigation Designation Ayuda

Class search strings:

Category Bug Kind Bug Pattern ↔ Bug Rank

Bugs (15)

- Correctness (5)
 - Bad use of return value from method (1)
 - Method ignores return value (1)
 - com.jobsket.core.engine.crawlers.ie.IrishjobsExtractorImpl**
 - Null pointer dereference (1)
 - Redundant comparison to null (3)
- Bad practice (1)
- Malicious code vulnerability (4)
- Dodgy (5)

Classify: unclassified

```
IrishjobsExtractorImpl.java in com.jobsket.core.engine.crawlers.ie
75 //println "Found title: " + title;
80 }
81 } else {
82     //println "No title has been found"
83 }
84
85 //SALARY
86 int ipayment = html.lastIndexOf("Salary:");
87 if(ipayment != -1) {
88     ipayment = html.indexOf("class=\"text\"", ipayment) + 13;
89     int fpayment = html.indexOf("<", ipayment);
90     if (fpayment != -1) {
91         String strSalary = html.substring(ipayment, fpayment).trim();
92         strSalary.replaceAll("€", "");
93         salary = (long)parseSalary(html.substring(ipayment, fpayment));
94         //println "Found salary : ${salary}"
95     }
96 } else {
97     //println "No salary has been found"
98 }
99
100 //DESCRIPTION
101 int iDescription = html.indexOf("Description</div>", ititle);
102 if(iDescription != -1) {
103     iDescription = html.indexOf("<div class=\"text\"", iDescription) + 18;
104 }
```

com.jobsket.core.engine.crawlers.ie.IrishjobsExtractorImpl.fetch(String) ignores return value of String.replaceAll(String, String)
At IrishjobsExtractorImpl.java:[line 92]
In method com.jobsket.core.engine.crawlers.ie.IrishjobsExtractorImpl.fetch(String) [Lines 67 - 122]
Called method String.replaceAll(String, String)

Method ignores return value

The return value of this method should be checked. One common cause of this warning is to invoke a method on an immutable object, thinking that it updates the object. For example, in the following code fragment,

```
String dateString = getHeaderField(name);
dateString.trim();
```

the programmer seems to be thinking that the trim() method will update the String referenced by dateString. But since Strings are immutable, the trim() function returns a new String value, which is being ignored here. The code should be corrected to:

```
String dateString = getHeaderField(name);
dateString = dateString.trim();
```

<http://findbugs.sourceforge.net>

UNIVERSITY OF MARYLAND

Class search strings:

Category Bug Kind Bug Pattern Bug Rank

- Bugs (22)
 - Correctness (1)
 - Useless/non-informative string generated (1)
 - Invocation of toString on an array (1)
 - Invocation of toString on buffer in com.jobsket.text.extractors**
 - Bad practice (7)
 - Bad implementation of cloneable idiom (3)
 - Confusing method name (1)
 - Method ignores results of InputStream.read() (2)
 - Null pointer dereference (1)
 - Malicious code vulnerability (10)

Classify: unclassified

```
TextExtractor.java in com.jobsket.text.extractors
70     }
71   }
72
73   /**
74    * @see org.jlibrary.core.indexer.extractors.Extractor#extractText(java.io.InputStream)
75    */
76   public String extractText(InputStream is) throws ExtractionException {
77
78     try {
79       StringBuffer sb = new StringBuffer();
80       byte[] buffer = new byte[1024];
81       while (is.read(buffer) != -1) {
82         sb.append(buffer);
83       }
84       is.close();
85       return sb.toString();
86     } catch (Exception e) {
87       throw new ExtractionException(e);
88     }
89   }
90
91   /**
92    * @see org.jlibrary.core.search.extraction.Extractor#extractText(byte[])
93    */
94   public String extractText(byte[] content) throws ExtractionException {
```

Invocation of toString on buffer in com.jobsket.text.extractors.TextExtractor.extractText(InputStream)
At TextExtractor.java:[line 82]
In method com.jobsket.text.extractors.TextExtractor.extractText(InputStream) [Lines 79 - 87]
Local variable named buffer

Invocation of toString on an array

The code invokes toString on an array, which will generate a fairly useless result such as [C@16f0472. Consider using Arrays.toString to convert the array into a readable String that gives the contents of the array. See Programming Puzzlers, chapter 3, puzzle 12.

```
1440
1441 javax.jcr.Node root = JCRUtils.getRootNode(session);
```

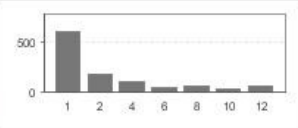
Dashboard

- Components
- Violations drilldown
- Time machine
- Clouds
- Hotspots



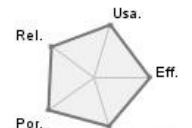
Version 1.2 - Domingo, 8 de Noviembre de 2009 18:54:24 +0100 - profile [Sonar way](#)

Lines of code 17.727 31.431 lines	Classes 145 23 packages 1.128 methods +337 accessors
-------------------------------------------------------	--------------------------------------------------------------------------------

Complexity 3,5 / method 27,3 / class 3.963 cmpx 8.501 statements	Methods Classes 
---------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------

Comments 17,6% 3.782 lines 53,2% docu. API 548 undocu. API 90 commented LOCs	Duplications 2,4% 762 lines 24 blocks 9 files
-----------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------

Code coverage 0,0% 0,0% line coverage 0,0% branch coverage 0 tests

Rules compliance 80,8% 	Violations 2.209 <ul style="list-style-type: none"> Blocker 0 Critical 15 Major 631 Minor 1.432 Info 131
------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------


Efficiency	99,4%
Maintainability	92,9%
Portability	99,0%
Reliability	92,9%
Usability	96,6%

Version 1.2

jLibrary Document Management System server core
 Key : org.jlibrary:jlibrary-server
 Language : java

[Developer connection](#) [Home](#) [Sources](#)

Dashboard
Components
Violations drilldown
Time machine
Clouds
Hotspots



Version 1.2 - Domingo, 8 de Noviembre de 2009 18:54:24 +0100 - profile [Sonar way](#)

Priority	Category	Rule	Count
Blocker	0		
Critical	15	Avoid Catching Throwable	6
Major	631	Security - Array is stored directly	4
Minor	1.432	Empty If Stmt	3
Info	131	Equals Hash Code	2

org.jlibrary.core.jcr	4	JCRRepositoryService	3
org.jlibrary.core.util.zip	3	UnrecognizedExtraField	2
org.jlibrary.core.util	3	FlexibleProperties	2
org.jlibrary.core.entities	2	ZipEntry	1
org.jlibrary.core.jcr.modules	2	UtilCache	1
org.jlibrary.core.properties	1	CustomPropertyDefinition	1

Path: CRITICAL [clear](#) » Any rule »

JCRRepositoryService

Sources Coverage **Violations** Duplications [\[Permalink\]](#)

0 3 38 111 0 Filter: CRITICAL (3) Expand:


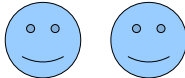


```

772
773         child.setProperty(JLibraryConstants.JLIBRARY_SIZE, resNode.getProperty(JCRConstants.JCR_DATA).getLength());
774
775         return child;
776     } catch (Throwable e) {
777         logger.error(e.getMessage(), e);
778         throw new RepositoryException(e);
779     } finally {
780         if (bais != null) {
781             try {
1439
1440
1441         javax.jcr.Node root = JCRUtils.getRootNode(session);

```

Testing y QA

Testing

- Todos somos humanos. El software inevitablemente tiene muchos errores.
- Bienvenidos sean los fallos.
- Por eso mismo debemos hacer pruebas.
 - Unitarias 
 - Integración 
 - Funcionales 
 - Stress 

Tests Unitarios

- Los hacen los desarrolladores
- Prueban la funcionalidad de tu código
- Introducidos ya en todos los lenguajes
- Junit, Nunit, PHPUnit, Test::Unit,...
- Prueban partes de código que no interactuen con recursos externos
 - No acceden a bases de datos, no acceden a la red, etc.

Tests Unitarios

```
public class LanguageDetectorTests extends TestCase {  
  
    @Test  
  
    public void testLanguageDetection() throws Exception {  
  
        LanguageIdentifier identifier = new LanguageIdentifier();  
        assertEquals(identifier.identify("This is a test"), "en");  
        assertEquals(identifier.identify("Esto es un test"), "es");  
    }  
}
```

Tests Unitarios

```
public void testExtractExperience() {  
  
    List<ExperienceEntry> entries =  
    extractor.extractExperience("Julio 2006 - Septiembre 2006 Vuelvo a  
    Sertec Soluciones informáticas Trabajando como Operador de  
    Sistemas dentro del Proyecto CRONOS de la Editorial  
    Planeta.\nSeptiembre 2007 - Julio 2008 Becario en el departamento  
    de Matemàtica Aplicada I de la Universitat Politècnica de  
    Catalunya dentro de los servicios informáticos dando soporte a los  
    profesores del departamento.", "es");  
  
    assertEquals(entries.size(), 2);  
  
    assertEquals(entries.get(0).getExperience(), 2);  
  
    assertEquals(entries.get(1).getExperience(), 10);  
  
}
```


Java - Eclipse SDK

File Edit Navigate Search Project Run Window Help

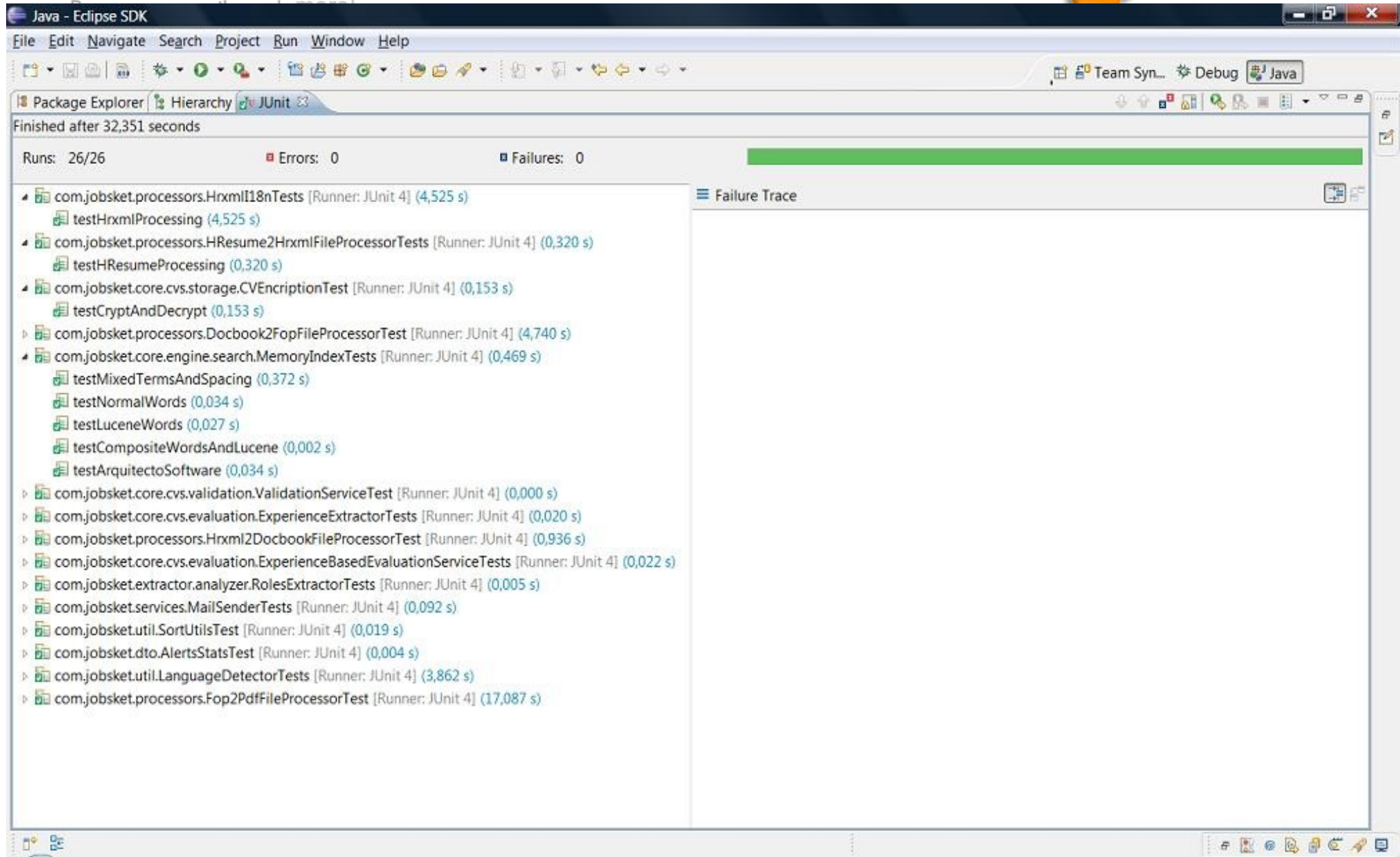
Package Explorer Hierarchy JUnit

Finished after 32,351 seconds

Runs: 26/26 Errors: 0 Failures: 0

- com.jobsket.processors.Hrxml118nTests [Runner: JUnit 4] (4,525 s)
 - testHrxmlProcessing (4,525 s)
- com.jobsket.processors.HResume2HrxmlFileProcessorTests [Runner: JUnit 4] (0,320 s)
 - testHResumeProcessing (0,320 s)
- com.jobsket.core.cvs.storage.CVEncriptionTest [Runner: JUnit 4] (0,153 s)
 - testCryptAndDecrypt (0,153 s)
- com.jobsket.processors.Docbook2FopFileProcessorTest [Runner: JUnit 4] (4,740 s)
- com.jobsket.core.engine.search.MemoryIndexTests [Runner: JUnit 4] (0,469 s)
 - testMixedTermsAndSpacing (0,372 s)
 - testNormalWords (0,034 s)
 - testLuceneWords (0,027 s)
 - testCompositeWordsAndLucene (0,002 s)
 - testArquitectoSoftware (0,034 s)
- com.jobsket.core.cvs.validation.ValidationServiceTest [Runner: JUnit 4] (0,000 s)
- com.jobsket.core.cvs.evaluation.ExperienceExtractorTests [Runner: JUnit 4] (0,020 s)
- com.jobsket.processors.Hrxml2DocbookFileProcessorTest [Runner: JUnit 4] (0,936 s)
- com.jobsket.core.cvs.evaluation.ExperienceBasedEvaluationServiceTests [Runner: JUnit 4] (0,022 s)
- com.jobsket.extractor.analyzer.RolesExtractorTests [Runner: JUnit 4] (0,005 s)
- com.jobsket.services.MailSenderTests [Runner: JUnit 4] (0,092 s)
- com.jobsket.util.SortUtilsTest [Runner: JUnit 4] (0,019 s)
- com.jobsket.dto.AlertsStatsTest [Runner: JUnit 4] (0,004 s)
- com.jobsket.util.LanguageDetectorTests [Runner: JUnit 4] (3,862 s)
- com.jobsket.processors.Fop2PdfFileProcessorTest [Runner: JUnit 4] (17,087 s)

Failure Trace



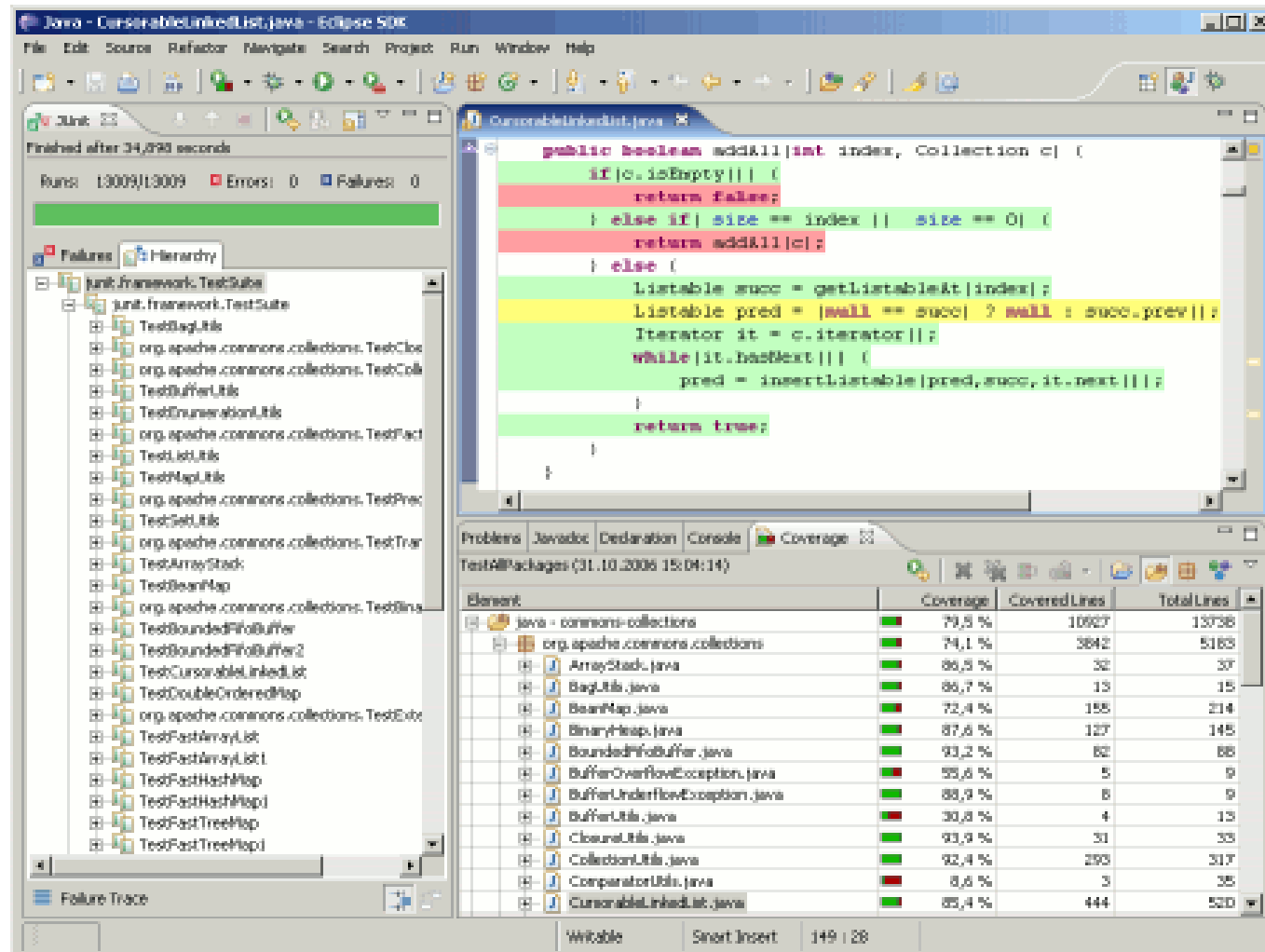
Tests Unitarios

- Típico: Tengo prisa. No tengo tiempo para hacer tests unitarios
- Los tests unitarios se hacen para ahorrar tiempo
 - Software más sólido y probado.
 - Menos cosas que probar manualmente.
 - Menos bugs. Los bugs se encuentran antes.
 - Menos presión.
- No hacer tests unitarios es no ser un buen programador

Cobertura de Código

- Es importante saber cuanto código estamos realmente probando.
 - Ideal 70%-80%
 - 100% es perder el tiempo. Demasiado caro.
- Muy útil para contrastar que nuestros tests están realmente haciendo lo que queremos.
- Muchísimas herramientas
 - Lo bueno es que no hay que configurar nada.
 - Se pueden integrar en el proceso de build.
 - Cobertura, EclEmma, Sonar, etc.

Cobertura de Código



The screenshot shows the Eclipse IDE interface with the following components:

- JUnit Console:** Shows a successful test run: "Finished after 34,098 seconds", "Runs: 13009/13009", "Errors: 0", "Failures: 0".
- Project Hierarchy:** Lists various test classes under the package `org.apache.commons.collections`, including `TestCursorableLinkedList`.
- Source Editor:** Displays the `addAll(int index, Collection c)` method in `CursorableLinkedList.java`. The code is color-coded to show coverage: green for covered lines and red for uncovered lines. The `while` loop body is highlighted in yellow.
- Coverage View:** A table showing coverage statistics for the project and its sub-packages.

Element	Coverage	Covered Lines	Total Lines
java - commons-collections	79,5 %	10927	13738
org.apache.commons.collections	74,1 %	3842	5183
ArrayStack.java	86,5 %	32	37
BagUtil.java	86,7 %	13	15
BeanMap.java	72,4 %	158	214
BinaryHeap.java	87,8 %	127	145
BoundedPriorityBuffer.java	93,2 %	62	66
BufferOverflowException.java	55,8 %	5	9
BufferUnderflowException.java	88,9 %	8	9
BufferUtil.java	30,8 %	4	13
CloneUtil.java	93,9 %	31	33
CollectionUtil.java	92,4 %	293	317
ComparatorUtil.java	8,8 %	3	35
CursorableLinkedList.java	85,4 %	444	520

Dashboard

Components

Violations drilldown


































Time machine

Clouds

Hotspots



Name	Rules compliance	Coverage	Build time	Links
jlibrary Server Core	80,8%	0,0%	18:54	  

Name	Rules compliance	Coverage	Build time	Links
 org.jlibrary.core.i18n	88,0%	0,0%	18:54	
 org.jlibrary.core.jcr.compatibility	84,0%	0,0%	18:54	
 org.jlibrary.core.jcr	89,0%	0,0%	18:54	
 org.jlibrary.core.jcr.modules	87,8%	0,0%	18:54	
 org.jlibrary.core.jcr.nodetypes	90,4%	0,0%	18:54	
 org.jlibrary.core.config	80,2%	0,0%	18:54	
 org.jlibrary.core.entities	65,1%	0,0%	18:54	
 org.jlibrary.core.factory	79,1%	0,0%	18:54	
 org.jlibrary.core.jcr.security	79,7%	0,0%	18:54	
 org.jlibrary.core.jcr.security.test	90,6%	0,0%	18:54	
 org.jlibrary.core.jcr.webdav	88,2%	0,0%	18:54	
 org.jlibrary.core.locking	88,9%	0,0%	18:54	
 org.jlibrary.core.profiles	72,1%	0,0%	18:54	
 org.jlibrary.core.properties	82,4%	0,0%	18:54	
 org.jlibrary.core.repository	68,3%	0,0%	18:54	
 org.jlibrary.core.repository.exception	100,0%	0,0%	18:54	
 org.jlibrary.core.search.algorithms	88,9%	0,0%	18:54	
 org.jlibrary.core.search	72,8%	0,0%	18:54	
 org.jlibrary.core.security.exception	100,0%	0,0%	18:54	
 org.jlibrary.core.security	66,3%	0,0%	18:54	
 org.jlibrary.core.util	66,0%	0,0%	18:54	
 org.jlibrary.core.util.xml	82,3%	0,0%	18:54	
 org.jlibrary.core.util.zip	71,5%	0,0%	18:54	
 org.jlibrary.test			18:54	
 org.jlibrary.test.authors			18:54	
 org.jlibrary.test.bookmarks			18:54	
 org.jlibrary.test.categories			18:54	
 org.jlibrary.test.content			18:54	
 org.jlibrary.test.content.versions			18:54	
 org.jlibrary.test.export			18:54	
 org.jlibrary.test.favorites			18:54	
 org.jlibrary.test.locking			18:54	
 org.jlibrary.test.notes			18:54	



Size

Lines of code

Color 0%  100%

Rules compliance

Tests de Integración

- Los hacen los desarrolladores
- Prueban la funcionalidad de tu código que depende de recursos externos
 - Pruebas de bases de datos, recursos en red, servicios web, EJBs, disco, etc.
- Requieren un setup. Inicialización de recursos, ficheros de configuración diferentes a producción, etc.
- Herramientas como Spring te lo hacen más sencillo

Tests de Integración

- Existen frameworks para todos los gustos
 - XMLUnit: Xmls, XSLTs, XSDs, ...
 - DBUnit: Acceso a bases de datos
 - HtmlUnit: Páginas web
 - SoapUI: Servicios web
 - Cactus: JSPs, Servlets, EJBs,...

```
@Test
public void homePage() throws Exception {
    final WebClient webClient = new WebClient();
    final HtmlPage page = webClient.getPage("http://htmlunit.sourceforge.net");
    assertEquals("HtmlUnit - Welcome to HtmlUnit", page.getTitleText());
}
```

HTMLUnit

```
...

public void testMe() throws Exception
{
    // Execute the tested code that modify the database here
    ...

    // Fetch database data after executing your code
    IDataset databaseDataSet = getConnection().createDataSet();
    ITable actualTable = databaseDataSet.getTable("TABLE_NAME");

    // Load expected data from an XML dataset
    IDataset expectedDataSet = new FlatXmlDataSetBuilder().build(new File("expectedDataSet.xml"));
    ITable expectedTable = expectedDataSet.getTable("TABLE_NAME");

    // Assert actual database table match expected table
    Assertion.assertEquals(expectedTable, actualTable);
}
}
```

DBUnit

```
public void testXSLTransformation() throws Exception {
    String myInputXML = "...";
    File myStylesheetFile = new File("...");
    Transform myTransform = new Transform(myInputXML, myStylesheetFile);
    String myExpectedOutputXML = "...";
    Diff myDiff = new Diff(myExpectedOutputXML, myTransform);
    assertTrue("XSL transformation worked as expected", myDiff.similar());
}
```

XMLUnit

Tests de Integración

- Hacer tests de integración requieres más esfuerzo que el hacer tests unitarios.
- Sin embargo, es muy necesario.
- Este esfuerzo es sólo de configuración.
 - ¿Cómo inicializo mi base de datos?
 - ¿Cómo inicializo mi servidor web?
- Muchos servidores pueden ser embebidos: Jetty, Tomcat, HSQLDB, ...

Tests de Integración

- Proceso típico:
 - 1) Inicializar base de datos, servidor web, ...
 - 2) Inicializar nuestros servicios.
 - 3) Ejecutar el test
- Si nos encontramos haciendo nuestro propio framework de integración: Mal camino.
 - Spring hace todo más fácil
- En Grails por ejemplo es ridículamente simple.
- Una vez está todo configurado. Vale la pena. Los tests son mucho más significativos.

Tests de Integración

```
void testFindAllByUsername() {  
  
    CV cv1 = buildCV()  
    cvService.save(cv1)  
    CV cv2 = buildCV()  
    cvService.save(cv2)  
  
    def cvs = cvService.findAllByUsername("test")  
    assertEquals cvs.size(), 2  
  
}
```

```
void testDeleteInvoice() {  
  
    def invoiceData = new InvoiceData()  
    invoiceData.user = recruiter  
  
    def invoice = billingService.generateInvoice(invoiceData)  
    assertEquals billingService.findAllInvoicesByUser(recruiter).size, 1  
  
    billingService.deleteInvoice(recruiter, invoice)  
    assertEquals billingService.findAllInvoicesByUser(recruiter).size, 0  
  
}
```

Tests de Integración

- Ojo con el solapamiento entre tests de integración y tests funcionales
- Si:
 - No usamos un framework que nos haga las cosas más fáciles
 - Nos está costando bastante el configurar el sistema (base de datos embebida, gestión de transacciones, limpieza entre tests, etc.)
- Entonces quizás sea mejor saltar directamente a hacer tests funcionales

Tests Funcionales

- No los deberían hacer los desarrolladores.
- Se comportan como si fueran usuarios reales.
- No conocen el sistema.
- Simplemente acceden a las páginas web, aplicación.
- Son con diferencia el mejor tipo de pruebas
 - ¡¡ El software funciona de verdad !!

QA

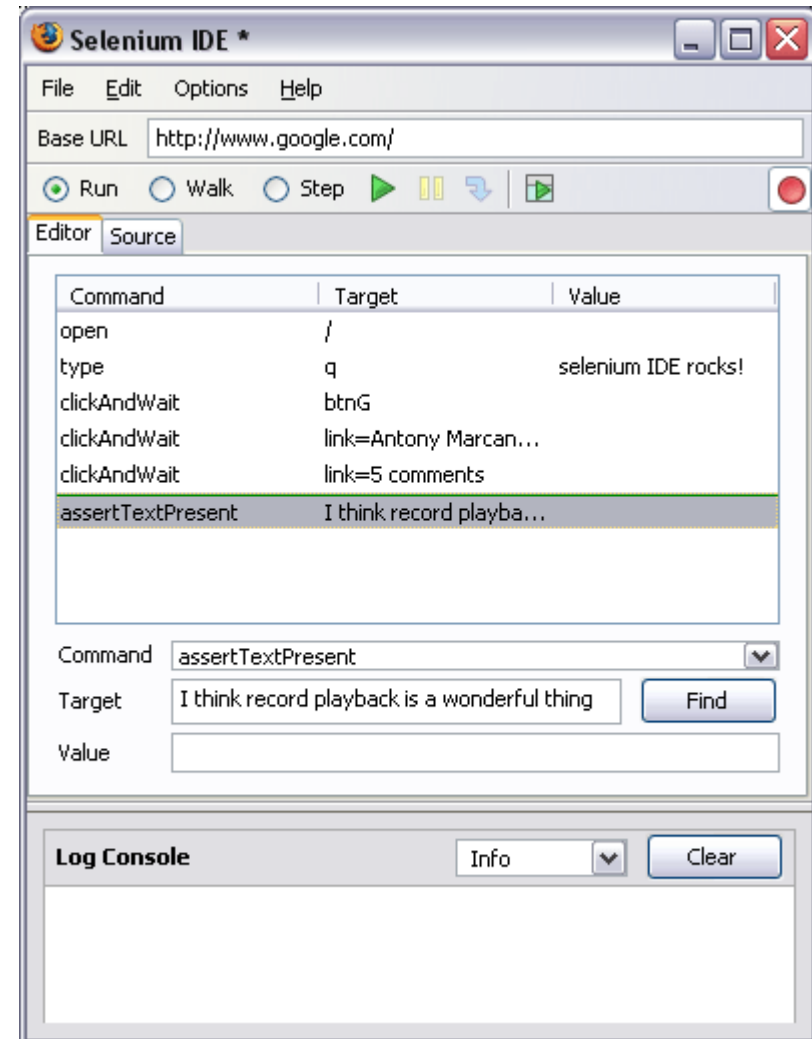
- Rara avis en España. Muy común en Europa.
- No informáticos que se dedican a probar las aplicaciones.
- Son personas muy útiles, pero es necesario guiarles. Es importante que se dejen guiar. Es gente que **no tiene experiencia en desarrollo de software**.
- ¿Deberían hacer tests funcionales los programadores?
 - Idealmente, no. QA mira el software de un modo completamente diferente.

Tests Funcionales

- Hoy por hoy hay muchas herramientas
 - Selenium
 - Canoo WebTest
 - HTMLUnit
- ¿Y si mi aplicación es Swing?
 - Abbot, Frankenstein,...
- Muchas herramientas comerciales
 - En una empresa usabamos HP QuickTest
 - QA grababa los tests desde un applet y se generaban scripts en Excel. ¡Les encantaba!

Tests Funcionales

- Selenium es enormemente sencillo
- Tiene un IDE para grabar tests.
- Los tests se pueden reproducir.
- Incluso genera código en diferentes lenguajes.
- Cualquiera puede grabar estos tests. Ideal para no desarrolladores.



Tests Funcionales

- Pero, lanza una instancia del navegador con cada tests, es decir, consume bastantes recursos.
- Los tests pueden ser complicados de mantener.
- Canoo WebTest

Tests Funcionales

- Ojo, a veces lo simple puede ser lo más potente
- HtmlUnit es realmente sencillo, pero no es nada amigable para no desarrolladores.

Tests Funcionales

- Ojo, a veces lo simple puede ser lo más potente
- HtmlUnit es realmente sencillo y no requiere navegadores abiertos.
- Pero sólo lo entendemos los programadores.

```
@Test
public void submittingForm() throws Exception {
    final WebClient webClient = new WebClient();

    // Get the first page
    final HtmlPage page1 = webClient.getPage("http://some_url");

    // Get the form that we are dealing with and within that form,
    // find the submit button and the field that we want to change.
    final HtmlForm form = page1.getFormByName("myform");

    final HtmlSubmitInput button = form.getInputByName("submitbutton");
    final HtmlTextInput textField = form.getInputByName("userid");

    // Change the value of the text field
    textField.setValueAttribute("root");

    // Now submit the form by clicking the button and get back the second page.
    final HtmlPage page2 = button.click();
}
```

Tests Funcionales

- Pero con un buen DSL cualquiera puede hacer tests.
- En uno de mis trabajos lo probé en una de mis empresas y se comenzaron a hacer tests como churros. 36 Qas por fin productivos.

```
Module: signin  
  
Goto '/home'  
ClickByText 'Sign In'  
Type 'Login','martin'  
Type 'Password','test'  
ClickButton 'Sign In'  
VerifyText 'Welcome Martin'
```

```
run signin  
  
Goto '/invoices'  
ClickButton 'New Invoice'  
VerifyText 'Create...'
```

Test Driven Development

- Crear primero el test y después el código.
- Difícil de comenzar, cambio completo de mentalidad.
- Tras acostumbrarse, muy productivo. Ya que:
 - Te obliga a enfrentarte al problema desde un punto de vista diferente.
 - Tras acabar la funcionalidad, ya tienes los tests.
- Fácil acostumbrarse al empezar con bugs
 - ¿Te pasan un bug? Test que lo reproduzca.

Lo que ningún libro te contará sobre los tests



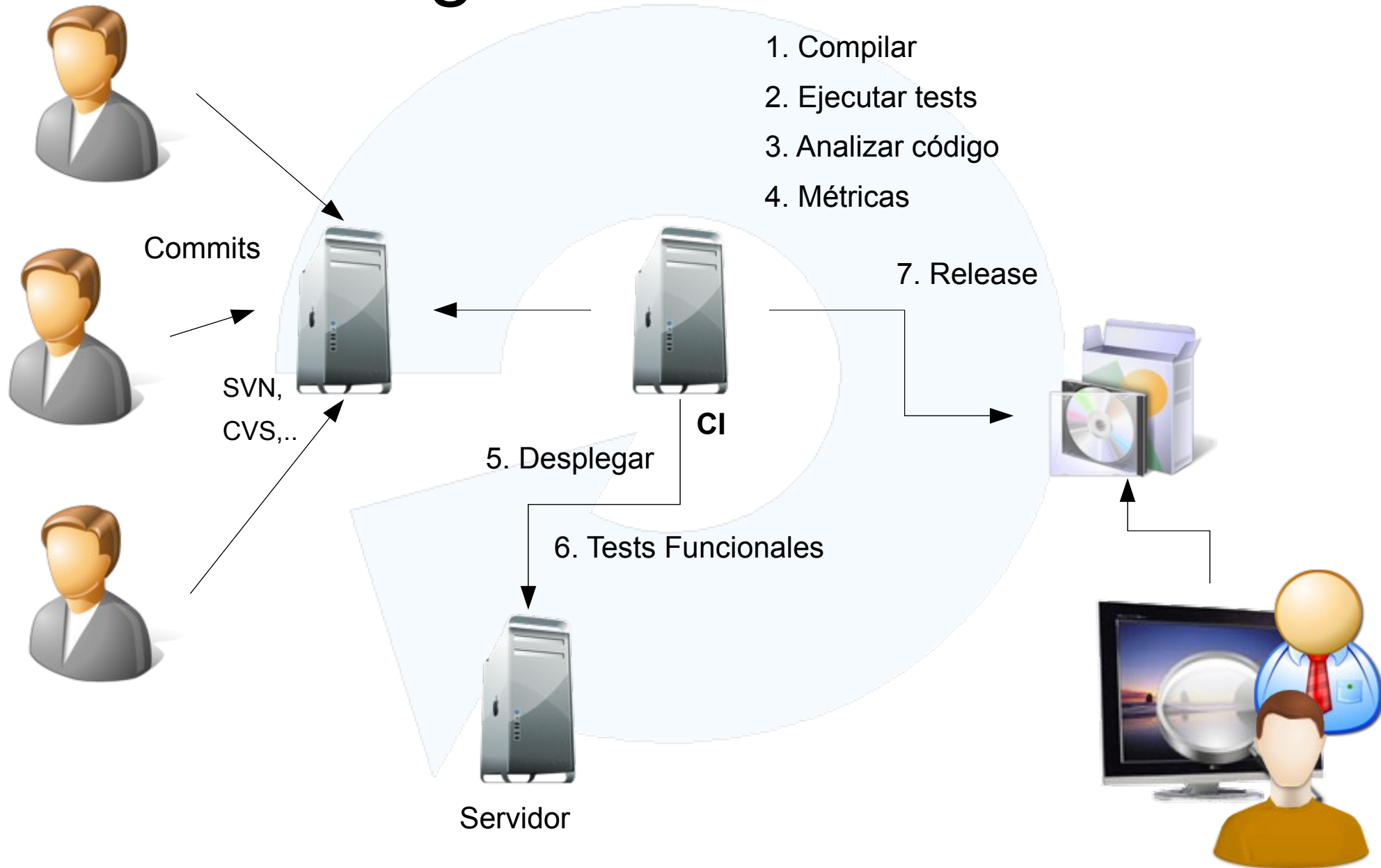
Integración Continua

¡Sálvame!

- Iteraciones cada dos semanas.
- Demos a final de cada iteración.
- Despliegue en producción cada mes.
- Un departamento de QA que necesita software que funcione.
- Un departamento Comercial que quiere hacer demos a clientes.

¿Imposible?

Integración Continua



Integración Continua

- Cruise Control, Hudson, Apache Continuum, Bamboo, Team Foundation Server,...
- Para mi, el mejor es Hudson
 - Instalación trivial
 - Enormemente sencillo de utilizar
 - Multitud de plug-ins

Integración Continua

Hudson

Hudson

-  [New Job](#)
-  [Manage Hudson](#)
-  [People](#)
-  [Build History](#)
-  [Project Relationship](#)
-  [Check File Fingerprint](#)
-  [My Views](#)

Build Queue
No builds in the queue.

Build Executor Status	
#	Status
1	Idle



All +		S	W	Job ↓	Last Success	Last Failure	Last Duration
				Docsket API	1 mo 23 days (#1)	N/A	24 sec
				Docsket Build	4 days 22 hr (#11)	N/A	3 min 41 sec
				Jobsket Build	12 hr (#2081)	13 hr (#2079)	6 min 32 sec
				Jobsket Crawlers	5 days 13 hr (#43)	17 days (#41)	39 sec
				Jobsket Extractors	5 days 13 hr (#13)	4 mo 13 days (#9)	46 sec
				Jobsket Functional Tests	2 mo 13 days (#520)	2 mo 13 days (#519)	4 min 17 sec
				Jobsket Storage	4 days 22 hr (#23)	2 mo 4 days (#18)	48 sec
				Jobsket WAR	12 hr (#956)	N/A	2 min 21 sec

Icon: [S](#) [M](#) [L](#)

[Legend](#)  [for all](#)  [for failures](#)  [f](#)

Hudson

Hudson » [Jobsket Build](#)

[ENABLE AUTO REFRESH](#)

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

[Subversion Polling Log](#)

Build History [\(trend\)](#)

#2081	13-nov-2009 19:25:43
#2080	13-nov-2009 18:55:43
#2079	13-nov-2009 18:46:39
#2078	13-nov-2009 17:50:43
#2077	13-nov-2009 17:10:43
#2076	13-nov-2009 16:20:43
#2075	13-nov-2009 16:10:43
#2074	13-nov-2009 15:35:43
#2073	13-nov-2009 12:30:43
#2072	13-nov-2009 11:05:43
#2071	13-nov-2009 10:45:45
#2070	13-nov-2009 9:50:42
#2069	12-nov-2009 19:20:42
#2068	12-nov-2009 18:40:42
#2067	12-nov-2009 18:02:20
#2066	12-nov-2009 17:55:42
#2065	12-nov-2009 17:15:42
#2064	12-nov-2009 16:45:42
#2063	12-nov-2009 16:35:42
#2062	12-nov-2009 16:15:53
#2061	12-nov-2009 16:05:42
#2060	12-nov-2009 15:37:32
#2059	12-nov-2009 15:30:44
#2058	12-nov-2009 15:12:26

Project Jobsket Build

Main build of Jobsket



[Workspace](#)



[Recent Changes](#)



[Latest Test Result](#) (no failures)

Upstream Projects

- [Docsket API](#)
- [Jobsket Crawlers](#)
- [Jobsket Extractors](#)
- [Jobsket Storage](#)

Downstream Projects

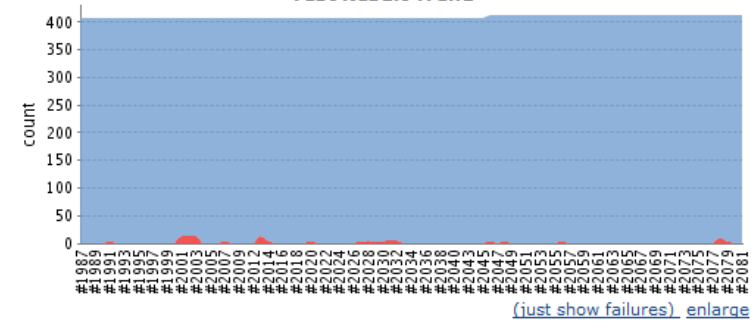
- [Jobsket WAR](#)

Permalinks

- [Last build \(#2,081\), 15 hr ago](#)
- [Last stable build \(#2,081\), 15 hr ago](#)
- [Last successful build \(#2,081\), 15 hr ago](#)
- [Last failed build \(#2,079\), 16 hr ago](#)


Test Result Trend

[edit description](#)



Hudson

Hudson » Jobsket Build » #2079

 [Back to Project](#)

 [Status](#)


 [Changes](#)

 [Console Output \[raw\]](#)

 [Tag this build](#)

 [Test Result](#)

 [Previous Build](#)

 [Next Build](#)

Build #2079 (13-nov-2009 18:46:39)



Revision: 4799

Changes

1. try to fix tests ([detail](#))



Started by user [jobsket](#)



[Test Result](#) (2 failures / -8)

[AdminControllerTests.testAffiliatePayments](#)
[AdminControllerTests.testAffiliatePaymentWithExpiredPromotionCode](#)

Failed

AdminControllerTests.testAffiliatePayments

Failing for the past 2 builds (Since

[Tool](#)
[add...](#)

Error Message

No signature of method: com.jobsket.services.BillingService.generateInvoice() is applicable for argument types: (com.jobsket.core.model.User, java.util.Date, java.lang.Boolean) values: {com.jobsket.core.model.User@1c226, Fri Nov 13 18:52:52 GMT 2009, true}

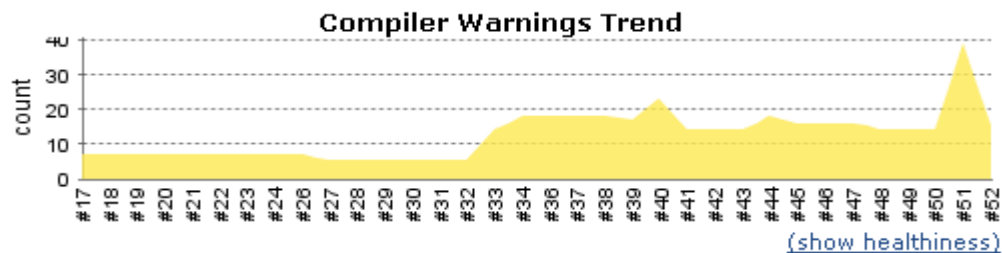
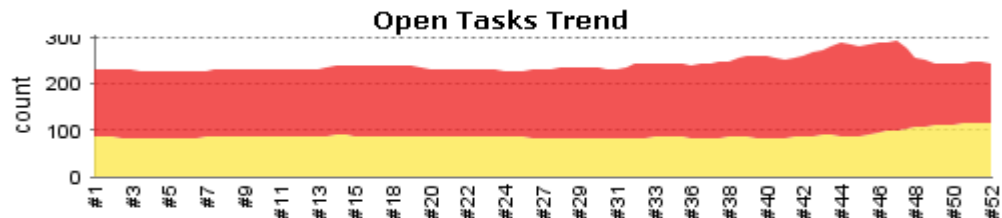
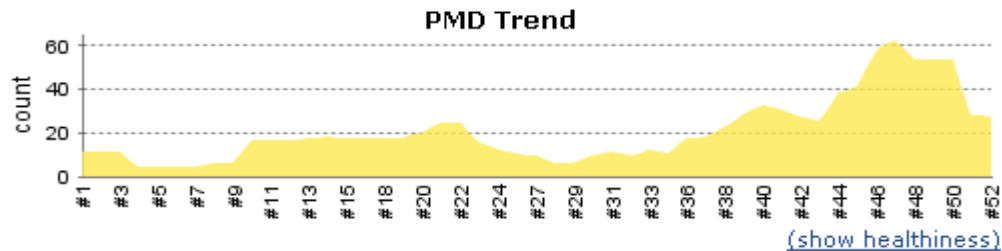
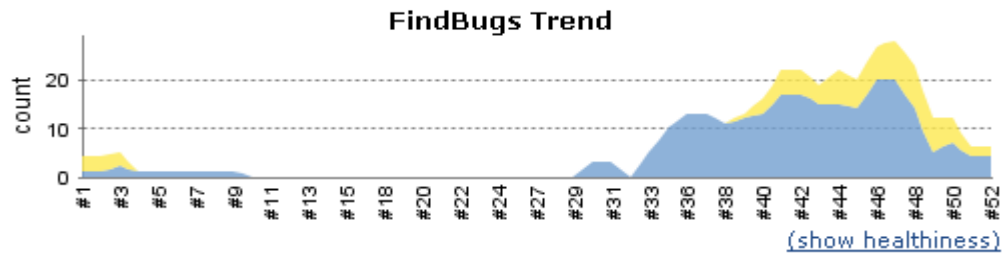
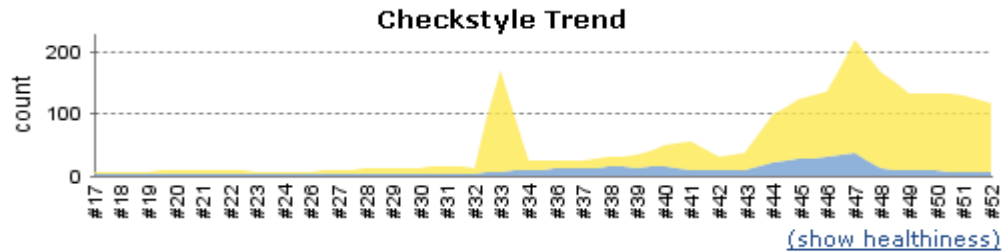
Stacktrace

```
groovy.lang.MissingMethodException: No signature of method: com.jobsket.services.BillingService.generateInvoice() is applicable for argument types:
(com.jobsket.core.model.User, java.util.Date, java.lang.Boolean) values: {com.jobsket.core.model.User@1c226, Fri Nov 13 18:52:52 GMT 2009, true}
    at org.codehaus.groovy.runtime.ScriptBytecodeAdapter.unwrap(ScriptBytecodeAdapter.java:55)
    at org.codehaus.groovy.runtime.ScriptBytecodeAdapter.invokeMethodN(ScriptBytecodeAdapter.java:172)
    at AdminControllerTests.testAffiliatePayments(AdminControllerTests.groovy:101)
    at org.codehaus.groovy.grails.support.GrailsTestSuite.runTest(GrailsTestSuite.java:72)
    at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:86)
    at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:230)
```

Hudson

Hudson » 1 Main Line » 31-Ana

-  [Back to Project](#)
-  [Status](#)
-  [Changes](#)
-  [Console Output](#)
-  [Tag this build](#)
-  [Checkstyle Warnings](#)
-  [Duplicate Code](#)
-  [FindBugs Warnings](#)
-  [PMD Warnings](#)
-  [Open Tasks](#)
-  [Compiler Warnings](#)



 **Build #15 (Sep 10, 2009**
12:02:44 AM)

Started 18 min ago
Took [10 sec](#)

 [add description](#)



No changes.



Started by user [anonymous](#)



Chuck Norris's beard can type 140 wpm.

Permalinks

- [Build number](#)



[Hudson ver. 1.317](#)

ject wadisadi

 [add description](#)

 Chuck Norris can instantiate an abstract class.



[Workspace](#)



[Recent Changes](#)

Permalinks

- [last build \(#34\), 17 min ago](#)
- [last stable build \(#34\), 17 min ago](#)
- [last successful build \(#34\), 17 min ago](#)



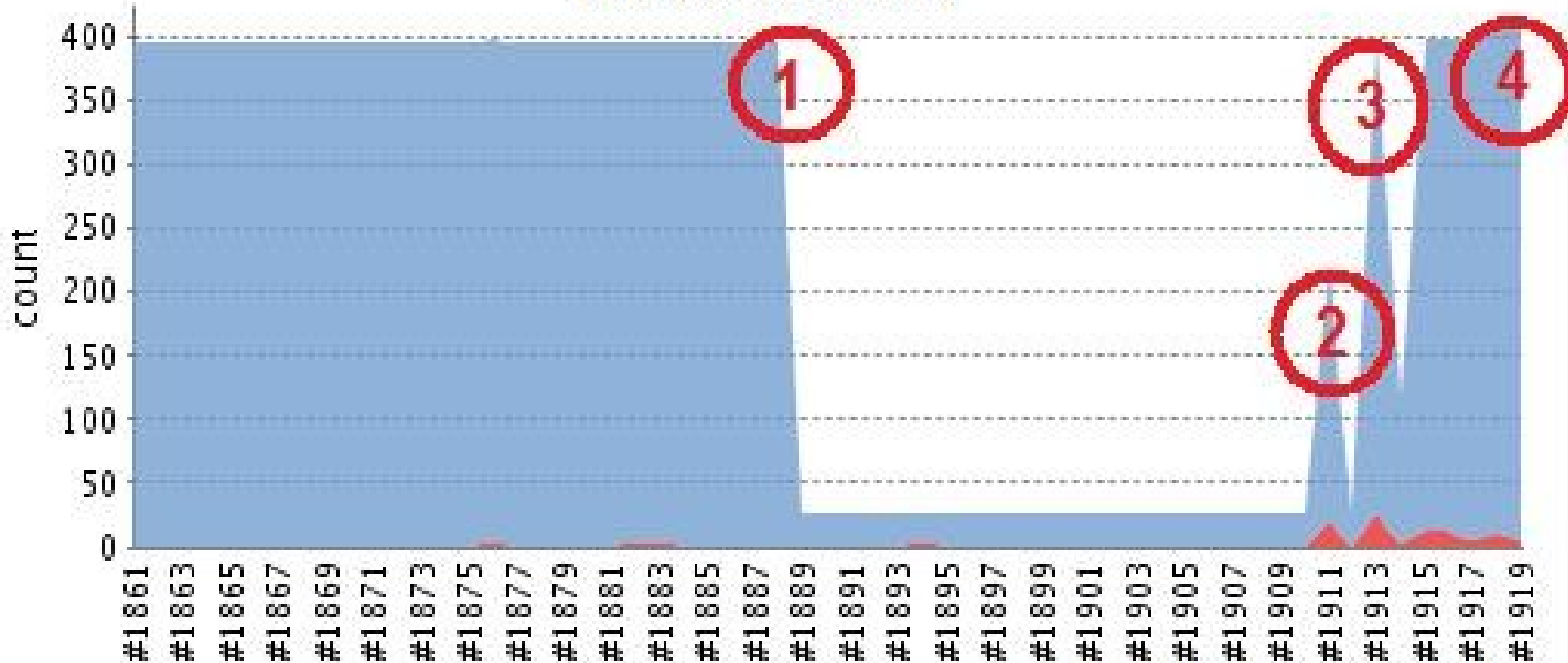
Chuck Norris

[Hudson ver. 1.317](#)

Respetar la build

 [edit description](#)

Test Result Trend



[\(just show failures\)](#) [enlarge](#)

Integración Continua

- Algunas buenas prácticas:
 - La build debe ser muy rápida. Intentar siempre acortar el tiempo de build.
 - No tener una única build. Separarlas para hacer el proceso más rápido.
 - Si la build se rompe, hay que parar y arreglarla.
 - Si hay QAs creando tests quizás convenga que tengan otra máquina.
 - Los tests funcionales suelen tomar bastante tiempo. Build propia o idealmente otra máquina.

Referencias

- <http://www.agile-spain.com>
- <http://groups.google.es/group/agile-spain/files?pli=1>
- <http://sites.google.com/site/axilmente/Home/recursos>
- <http://pragprog.com>
- <http://blog.objectmentor.com>
- <http://www.crisp.se/henrik.kniberg/ScrumAndXpFromTheTrenches.pdf>
- <http://www.infoq.com>
- <http://www.presionblogosferica.com>
- <http://www.javahispano.org>

Gracias