

# UNIVERSIDAD DE BURGOS

## ESCUELA POLITÉCNICA SUPERIOR



### Grado en Ingeniería Informática

### **Monitorización del proceso de trabajo con Git**

Conceptos: Medición y Calidad

Alumno      Andrés Pérez Juárez

Tutor      Carlos López Nozal  
DEPARTAMENTO DE INGENIERÍA CIVIL  
Área de Lenguajes y Sistemas Informáticos

Burgos, 17 de febrero de 2017

Este documento está licenciado bajo [Creative Commons Attribution 3.0 License](https://creativecommons.org/licenses/by/3.0/)



## Índice de contenido

1. OBJETIVOS ESPECIFICOS.....	4
2. ENUNCIADO.....	4
3. CUESTIONES.....	5
• ¿Se ha realizado trabajo en equipo?.....	5
• ¿Tiene calidad el conjunto de pruebas disponibles?.....	5
• ¿Cuál es el esfuerzo invertido en realizar la actividad?.....	5
• ¿Cuál es el número de fallos encontrados en el código?.....	6
REFERENCIAS.....	7

**Índice de ilustraciones**

Ilustración 1: Porcentaje tests.....4

## **1. OBJETIVOS ESPECIFICOS**

- Comprender los objetivos de medición relacionados con la caracterización y la evaluación de productos, procesos y recursos software.
- Comprender, aplicar y analizar técnicas de medición sobre entidades de productos software relacionados con conjuntos de pruebas de software.
- Comprender, aplicar y analizar medidas relacionadas sobre entidades de proceso y recursos de prueba del software.

## **2. ENUNCIADO**

En la práctica se va simular un pequeño desarrollo de un producto software para realizar mediciones sobre él. El objetivo es establecer un caso de estudio que sirva para caracterizar y evaluar tanto el producto desarrollado como el proceso seguido.

### 3. CUESTIONES

#### • ¿Se ha realizado trabajo en equipo?

El trabajo no se ha podido llevar a cabo en equipo, debido a que me encuentro solo para la realización de las prácticas.

#### • ¿Tiene calidad el conjunto de pruebas disponibles?

El conjunto de pruebas llevadas a cabo tiene una calidad del 78,6%, quedando el desarrollo de los test de las diferentes clases de la siguiente manera. Lo que es genial y nos permite Git es tener acceso a absolutamente todo lo que hemos programado, tenga mayor o menor calidad. La forma de realización de JUnit nos permite crear unos test que aunque cuesten trabajo, a posteriori nos ahoraran tiempo para encontrar el error.

ReusablePoolTest (16-feb-2017 22:21:34)

















Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
▼ P01_DASS	 78,6 %	187	51	238
▼ src	 78,6 %	187	51	238
▼ ubu.gii.dass.c01	 65,9 %	85	44	129
> Client.java	 8,1 %	3	34	37
> Reusable.java	 23,1 %	3	10	13
> DuplicatedInstanceException.java	 100,0 %	4	0	4
> NotFreeInstanceException.java	 100,0 %	4	0	4
> ReusablePool.java	 100,0 %	71	0	71
▼ ubu.gii.dass.test.c01	 93,6 %	102	7	109
▼ ReusablePoolTest.java	 93,6 %	102	7	109
▼ ReusablePoolTest	 93,6 %	102	7	109
● testReleaseReusable()	 84,6 %	33	6	39
● testAcquireReusable()	 94,1 %	16	1	17
● setUp()	 100,0 %	10	0	10
● tearDown()	 100,0 %	15	0	15
● testGetInstance()	 100,0 %	15	0	15

Ilustración 1: Porcentaje de test

#### • ¿Cuál es el esfuerzo invertido en realizar la actividad?

El esfuerzo ha resultado bastante a la hora de programarlo, debo reconocer que hacia tiempo que no manejaba eclipse y retomarlo me ha llevado más horas de las que supuse en un principio. Los problemas han ido surgiendo según realizaba un test tras otro implementado previamente, comencé sin crear objetos Reusable ni ReusablePool,, a la larga me di cuenta de que debía definir ese tipo de variables. El tema de tratar las excepciones me ha llevado también mucho tiempo, no sé porque no acertaba al principio en cosas relativamente sencillas, y al estar solo en la práctica y sin segunda opinión en algunas ocasiones me he atascado con errores que una segunda opinión me habría ayudado a solventar. Los test en principio los trate de realizar llamando directamente a la función en los assert, despues con variables sin tratar las excepciones en los test, más tarde tratando las excepciones... por ello mi código ha sufrido subidas y bajadas del porcentaje debido a que he tratado de indagar cuales eran los motivos por los que JUnit me subía o bajaba el mismo.

Horas empleadas;

- Viernes 10/02 por la mañana:(1 hora y media),
- Lunes 13/02 por la noche:(2 horas).
- Martes 14/02 por la noche:(3 horas).
- Miércoles 15/02 por la mañana y noche:(4 horas).
- Jueves 16/02 por la mañana y noche(4 horas y media).
- Viernes 17/02 por la mañana:(hora y media)
- Total de 15 horas y media.

### • ¿Cuál es el número de fallos encontrados en el código?

El número de fallos no los he ido contando, en principio el código realice el test `testGetInstance()`, con una implementación sencilla que parecía correcta, lo mismo hice con `testAcquireReusable()`, seguía todo correcto hasta que me propuse realizar el test de `testReleaseReusable()`, donde me di cuenta que me vendría bien la creación de las variables mencionadas en la cuestión anterior, también me di cuenta de que ingenuo de mi, no había tratado la excepción de `testAcquireReusable()`. Trás tratar la excepción de `testAcquireReusable()` tambien empecé a fijarme que el método `testGetInstance()` precisaba de algunas modificaciones y `assert`. Respecto a el porcentaje correcto es de 84,6%, el porcentaje restante creo que es de no tratar la excepción `NotFreeInstanceException` necesaria para la llamada a `acquireReusable()`.

El porcentaje restante para llegar al 100% corresponden a los test de la clase `Client` y `Reusable`, para las cuales he implementado algún `assert` en el método `tearDown()`.

## REFERENCIAS

- [1] Código de repositorio en GitHub.<https://github.com/clopezno/poolobject>.  
Autor: Carlos López Nozal.
- [2] Eclipse IDE for Java Developers <http://www.eclipse.org/downloads/>
- [3] Plantilla de documentación de prácticas. Autor: Carlos Lopez Nozal.
- [4] Plugin Eclipse EcEmma <http://www.eclemma.org/>.
- [5] Repositorio GitHub.<https://github.com/>