

Práctica

Configuración de un pipeline de integración continua

Objetivo

El objetivo de la práctica es que el alumno configure un pipeline de integración continua de un proyecto software poniendo en práctica los conceptos vistos en el módulo de Integración Continua del Curso de Experto en Aseguramiento de la Calidad Software.

Proyecto Software

El pipeline de integración continua se utilizará para la gestión del ciclo de vida de un proyecto software existente. El software que deberá usarse es el juego de las Tres en Raya para el que se han implementado diferentes tipos de tests en la Práctica 2 del Módulo de Pruebas de Software.

A aquellos alumnos que no han realizado la práctica del módulo anterior, se les proporcionará una versión del software con los tests ya implementados.


Pipeline de Integración Continua

El pipeline de Integración Continua deberá tener las siguientes características:

- ✧ Se utilizará la forja de desarrollo de codeurjc¹ que incluye la mayoría de los servicios necesarios para configurar el pipeline. El único servicio necesario que no está disponible en la forja es el servidor SonarQube, que habrá que ejecutar por separado (preferiblemente dockerizado).
- ✧ El proyecto software se alojará en un repositorio del servidor Gerrit incluido en la forja.
- ✧ Se crearán los siguientes jobs en Jenkins:
 - **Job de commit:** Se creará un job de Jenkins, que será ejecutado cada vez que se suba un parche a Gerrit. El job ejecutará únicamente los tests unitarios del proyecto software y se utilizará para verificar el parche.
 - **Job de merge:** Se creará un job de Jenkins que será ejecutado cada vez que se haga merge de un parche en master. El job realizará las siguientes operaciones:
 - Ejecutará los tests unitarios y de sistema.
 - Al ejecutar los tests de sistema implementados usando browsers se deberá grabar el contenido de dichos browsers y esa grabación se archivará en la ejecución del Job. Es

1 <https://github.com/codeurjc/codeurjc-forge>

posible que esto requiera modificar el código de los tests para usar TestContainers o algún servicio de SeleniumGrid como Zalenium o Selenoid.

- Analizará la calidad del código con SonarQube, publicando los resultados del análisis en este servicio.
- Para facilitar la trazabilidad, se incluirá información del commit en el proyecto Spring que se hará disponible usando una API REST. 
- Se generará una imagen Docker. Se incluirá como label el identificador del commit para facilitar la trazabilidad.
- Se publicará la imagen en DockerHub con el tag dev.
- **Job de nightly:** Se creará un job que será ejecutado periódicamente por las noches (por ejemplo a las 1:00 am). Este job realizará las siguientes tareas:
 - Se ejecutarán los tests unitarios y de sistema.
 - Se creará una imagen Docker
 - Se publicará la imagen con el tag siguiendo el formato X.Y.Z.nightly.YYYYMMDD
 - Se ejecutará el software usando la imagen creada y se ejecutarán contra ese software los tests de sistema. De esta forma se comprobará la calidad de la imagen.
 - Si los tests pasan, esa imagen se subirá de nuevo a DockerHub con el tag “nightly”.
- **Job de release:** Se creará un job de release que será ejecutado de forma manual.
 - Recibirá como parámetro la nueva versión de desarrollo del software. Esa versión se usará para actualizar el fichero pom.xml una vez que se haya generado la release.
 - Se modificará el pom.xml quitando el SNAPSHOT de la versión.
 - Se ejecutarán los tests unitarios y de sistema.
 - Se generará la imagen Docker y se publicará en DockerHub con el tag igual a la versión del pom.xml
 - Se publicará de nuevo con el tag “latest”.
 - Se creará un tag en el repositorio Git en Gerrit con el nombre igual a la versión del pom.xml
 - Se actualizará el pom.xml con la versión indicada como parámetro. Si no acaba en “-SNAPSHOT”, se añadirá de forma automática ese valor.



Entrega

Al entregar la práctica, se deberán entregar los siguientes elementos:

- ⌘ Memoria: Un documento PDF explicando cada una de las tareas realizadas durante el desarrollo de la práctica:
 - Instalación y configuración de SonarQube.
 - Creación y configuración de los diferentes tipos de Jobs: Pipeline, triggers, etc. El pipeline

de cada Job deberá incluirse en ficheros “Jenkinsfile” en el repositorio git. Además deberán incluirse en la memoria a modo de documentación.

- Configuraciones adicionales que sean necesarias en Jenkins o en cualquiera de los servicios de la forja.

⌘ Código fuente del proyecto.

Procedimiento y fecha de entrega

La práctica puede realizarse de forma individual o por parejas.

La entrega se realizará el día **27 de Mayo de 2019** por el campus virtual de la UPM (<https://moodle.upm.es>). En caso de que algún alumno no tenga acceso, lo enviará por correo electrónico a **patxi.gortazar@gmail.com**. Se deberá entregar un .zip cuyo contenido será el proyecto eclipse en el que se ha realizado la práctica y la memoria en PDF.

