UNIVERSIDAD DE GRANADA

Tema 10:
Ejercicios de teoría

Autor
Juan Manuel Castillo Nievas

Máster Profesional en Ingeniería Informática

—

Granada, 11 de diciembre de 2020

1. Escoge dos sistemas de control de versiones y realiza un pequeño trabajo de una cara indicando cómo funcionan, cuáles son sus principales características y qué ventajas tiene uno frente al otro.

Se ha elegido **Git** ya que es un sistema que controlo bastante y por otro lado se ha elegido **AccuRev**.

1. AccuRev

Es un sistema de control de versiones que está centralizado en un modelo cliente-servidor y cuya comunicación se realiza mediante TCP/IP. Los servidores pueden funcionar como servidores de equipo, de integración continua o de compilación.

Está basado en una arquitectura de flujos (**streams**). Es un repositorio ordenado cronológicamente que mantiene la historia de ficheros que cambian en cada stream y permiten desarrollar diferentes versiones del código en cada stream, mientras que cada stream se actualiza de manera transparente con el código principal de los streams de su padre e hijos. Un stream sólo se actualiza por un hijo cuando el hijo provoca cambios en el stream del padre.

En **AccuRev** se pueden hacer cambios mediante la línea de comandos, mediante una interfaz gráfica de Java, una interfaz web o entornos de desarrollo como Eclipse.

2. Git

Es otro sistema de control de versiones que ofrece la posibilidad de modificar código en conjunto con otros desarrolladores y mantener un historial de lo que ha cambiado en cada versión.

Git cuenta con un repositorio remoto que está almacenado en un servidor y de forma local en cada ordenador de cada desarrollador. Esto implica que cada desarrollador tiene una copia local en su ordenador de todo el repositorio. Esto tiene numerosas ventajas como el poder consensuar los cambios con los demás compañeros de equipo antes de subir al servidor central. De esta manera un desarrollador puede cambiar lo que crea oportuno en el código en su repositorio local y, una vez ha hablado con sus compañeros para ver si están de acuerdo con sus cambios, puede subir al repositorio central que va a permitir que sus compañeros puedan bajarse esos cambios.

También ofrece lo que son las **branch** o ramas. Una rama es un puntero al último *commit* del repositorio. Una vez que se ha creado la rama, todo *commit* que se haga a esa rama no afectará a la rama principal. Esto puede ser útil para crear distintas versiones del código. Imaginemos que se quieren implementar dos nuevas funcionalidades independientes. Se pueden crear dos ramas para cada funcionalidad y, una vez se tengan las dos funcionalidades hechas, se mergean para combinarlas en la rama principal. También podría darse el caso de que una versión al final no la queramos, pues podria borrarse esa rama sin que afecte al código principal.

3. Comparación

En la Tabla 1 se muestra una comparación entre estas dos herramientas.

	AccuRev	Git
CARACTERÍSTICAS	Integra las mejores prácticas de los gestores de configuración de software.	Permite tener múltiples ramas localmente que pueden ser totalmente independientes. La creación, combinación y eliminación de estas ramas se hacen de forma muy rápida.
	Permite vincular el código a una serie de defectos, requisitos, o historias de usuario. De esta manera se permite compartir sólo los issues que se han completado, y el equipo ve todo el código asociado a ese issue.	Casi todas las operaciones se hacen de forma local, que hace que se hagan de forma mucho más rápida que si tuvieran que hacerse sobre un servidor central.
	Arquitectura basada en flujos. Permite crear ramas para el aislamiento y desarrollo en paralelo sin el coste de crear y manejar ramas.	Es un gestor de configuración de software distribuido, lo cual significa que se tiene el repositorio de forma local mediante la operación **clone**
	Se permite manejar de forma fácil el proceso de desarrollo de código, desde el área privada del desarrollador hasta el flujo de producción.	Se asegura la integridad encriptográfica. Es imposible poder obtener algo de git que no sea el número de bits de lo que se pone.
		Tiene un área de staging en el que se permite ver los ficheros que se han modificado, borrado, o aquellos ficheros que git aún no conoce o que conoce pero no se han tocado antes de hacer un commit.
LENGUAJES	C y otros	Otros
TIPO DE CÓDIGO	Cerrado	Abierto
LICENCIA	Propietario	GPL
TIPOS DE S.O	Windows, Mac, Linux, Unix	Windows, Mac, Linux, Unix, iOS
PRECIO	Prueba gratuita	Totalmente gratuito

Figura 1: Tabla de comparación