**Identificar al menos 4 herramientas que se utilicen en el versionamiento a nivel de proyectos o SW**

**TITULACIÓN DE SISTEMAS INFORMÁTICOS Y COMPUTACION**

**PROCESOS DE INGENIERÍA DE SOFTWARE**

**TRABAJO N.1**

**SEGUNDO BIMESTRE**

**Nombre:**

Carlos Caraguay

**Tutores:**

Ing. Daniel Guamán

**Fecha:** 15 – 06 – 2015

**Periodo Académico** Abril – Agosto 2015

UTPL

2015

**UNIVERSIDAD TECNICA PARTICULAR DE LOJA**

**GitHub**



Es una plataforma de desarrollo colaborativo de software para alojar proyectos utilizando el sistema de control de versiones Git.

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o front end como Cogito o StGIT.

**CARACTERÍSTICAS:**

* El código se almacena de forma pública, aunque también se puede hacer de forma privada, creando una cuenta de pago.
* GitHub aloja tu repositorio de código y brinda herramientas muy útiles para el trabajo en equipo, dentro de un proyecto tales como:
* Wiki
* Sistema de seguimiento de Problemas
* Herramienta de revisión de código
* Visor de ramas
* GitHub provee de funcionalidades para hacer un fork (es simplemente clonar un repositorio ajeno, eliminar algún bug o modificar cosas de él) y solicitar pulls (envía los cambios realizados al repositorio).
* Gestión eficiente de proyectos grandes, dada la rapidez de gestión de diferencias entre archivos, entre otras mejoras de optimización de velocidad de ejecución.
* Realmacenamiento periódico en paquetes (ficheros). Esto es relativamente eficiente para escritura de cambios y relativamente ineficiente para lectura si el reempaquetado (con base en diferencias) no ocurre cada cierto tiempo.

**Mercurial**



Es un sistema de control de versiones multiplataforma, para desarrolladores de software, publicada bajo una licencia libre (GPL). Está implementado principalmente haciendo uso del lenguaje de programación Python, pero incluye una implementación binaria de diff escrita en C. Mercurial fue escrito originalmente para funcionar sobre Linux. Ha sido adaptado para Windows, Mac OS X y la mayoría de otros sistemas tipo Unix. Mercurial es, sobre todo, un programa para la línea de comandos. Todas las operaciones de Mercurial se invocan como opciones dadas a su programa motor, hg (cuyo nombre hace referencia al símbolo químico del mercurio)

**CARACTERÍSTICAS:**

* Carácter distribuido
* Gran velocidad de funcionamiento.
* Recuperación de versiones y Visualización de diferencias entre revisiones
* Visualización de las distintas cabezas de un proyecto.

**MONOTONE**

Monotone es una herramienta software de fuente abierta para control distribuido de versiones.

Monotone registra revisiones de ficheros, agrupa conjuntos de revisiones ('changesets') y mantiene el histórico tras cambios de nombre. El principio de diseño es el de operación distribuida haciendo gran uso de primitivas criptográficas para trazar revisiones de ficheros (mediante el hash seguro SHA-1) y para autenticar acciones de usuarios (mediante firmas criptográficas RSA). Cada participante mantiene su propio almacén de revisiones históricas en una base de datos SQLite local.

**CARACTERÍSTICAS:**

* Al igual que GNU arch y al contrario que Subversion, Monotone se basa en un modelo distribuido de control de versiones. Usa hashes SHA-1 en lugar de números de versión a la hora de identificar ficheros o grupos de ficheros específicos, al igual que Git y Mercurial.
* Aunque Monotone solía admitir varios protocolos de red para sincronizar árboles, ahora usa de forma exclusiva un protocolo propio bautizado netsync, que es más robusto y eficiente (sin embargo, desde la versión 0.27, es posible usar el protocolo netsync sobre otros medios, incluyendo conexiones ssh).
* No existe un servidor específico debido a que cualquier cliente Monotone puede actuar como servidor, de acuerdo con el principio end-to-end (comunicación directa entre extremos, que aparece en el diseño de TCP).

**SUBVERSION**

Subversion (SVN) es una herramienta de control de versiones open source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Es software libre bajo una licencia de tipo Apache/BSD.

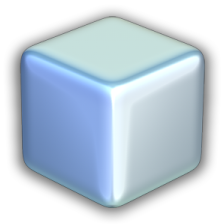
Utiliza el concepto de revisión para guardar los cambios producidos en el repositorio. Entre dos revisiones sólo guarda el conjunto de modificaciones (delta), optimizando así al máximo el uso de espacio en disco. SVN permite al usuario crear, copiar y borrar carpetas con la misma flexibilidad con la que lo haría si estuviese en su disco duro local. Dada su flexibilidad, es necesaria la aplicación de buenas prácticas para llevar a cabo una correcta gestión de las versiones del software generado.

**CARACTERISTICAS:**

* Se sigue la historia de los archivos y directorios a través de copias y renombrados.
* Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
* La creación de ramas y etiquetas es una operación más eficiente. Tiene coste de complejidad constante (O(1)) y no lineal (O(n)) como en CVS.
* Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).
* Puede ser servido mediante Apache, sobre WebDAV/DeltaV. Esto permite que clientes WebDAV utilicen Subversion de forma transparente.
* Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto).
* Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.
* Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor provee a la hora de autentificar archivos (SQL, LDAP, PAM, etc).

**Consulte al menos 4 aplicaciones de software las cuales tengan versionamiento, por lo menos 2 niveles (V1, V1.1, V1.2, V2)**

**NetBeans**



NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE2 es un producto libre y gratuito sin restricciones de uso.

**Autocad**

Autodesk AutoCAD es, como lo indica su nombre, un software CAD utilizado para dibujo 2D y modelado 3D. Actualmente es desarrollado y comercializado por la empresa Autodesk. El nombre AutoCAD surge como creación de la compañía Autodesk, en que Auto hace referencia a la empresa creadora del software y CAD a Diseño Asistido por Computadora (por sus siglas en inglés), teniendo su primera aparición en 1982.1 AutoCAD es un software reconocido a nivel internacional por sus amplias capacidades de edición, que hacen posible el dibujo digital de planos de edificios o la recreación de imágenes en 3D; es uno de los programas más usados por arquitectos, ingenieros, diseñadores industriales y otros.

**OpenRefine**

OpenRefine es una aplicación de escritorio de código abierto independiente para la limpieza y transformación de datos a otros formatos, la actividad conocida como disputas de datos. Es similar a las aplicaciones de hoja de cálculo (y puede trabajar con formatos de archivo de hoja de cálculo), sin embargo, se comporta más como un base de datos.

Las versiones estables se liberan cada 6 meses, y Canonical proporciona ayuda técnica y actualizaciones de seguridad para la mayoría de las versiones de Ubuntu durante 18 meses, excepto para las versiones LTS (Long Term Support), para las que ofrece 3 años para la versión de escritorio y 5 años para la versión de servidor, a partir de la fecha del lanzamiento. A partir de la liberación de su versión 12.04, todas las versiones LTS sea de escritorio o de servidor, tendrán un soporte de 5 años, y para las ediciones No LTS tendrán un soporte de 9 meses, esto a partir del lanzamiento de la versión 13.042

La nomenclatura de las versiones no obedece principalmente a un orden de desarrollo, se compone del dígito del año de emisión y del mes en que esto ocurre, por ejemplo: Ubuntu 10.04 (2010, abril).

**Describir que significa haber pasado del 1 a la 2.**

Para definir el estado en el cual se cambia de una versión a otra definimos primeramente q existe un versionamiento semántico en el cual se basa esto que es un convenio o estándar a la hora de definir la versión de tu código, dependiendo de la naturaleza del cambio que estás introduciendo. De tal forma, se identifican 3 tipos de cambios:

Major: Cambio drástico en el software. No es compatible con código hecho en versiones anteriores.

Minor: Cambio que añade alguna característica nueva al software o modifica alguna ya existente, pero que sigue siendo compatible con código existente. También cuando marcamos algo como obsoleto.

Patch: Cuando arreglamos un bug siendo el cambio retrocompatible.

De esta forma, tenemos un lenguaje común entre desarrolladores y consumidores a la hora de hablar de versiones.

La fórmula para definir esto es:

Major.minor.patch = > 1.0.1

Por lo tanto el pasar de una versión a otra v1 a v2 significa que hubo cambios significativos en el software, donde se actualizo todo tipo de normas y más mecanismos de programación que ya no son compatibles con código anterior.

**Bibliografía**

RVBurke. (08 de 07 de 2012). Recuperado el 13 de 06 de 2015, de Control de versiones con Mercurial: http://blog.rvburke.com/2006/09/08/control-de-versiones-con-mercurial-ii-uso-de-mercurial/

Castillo, L. (31 de 03 de 2014). Introduccion a GitHub. Recuperado el 13 de 06 de 2015, de Github.com: http://conociendogithub.readthedocs.org/en/latest/data/introduccion/

JuntaANDALUCIA. (2012). Juntadeandalucia.es. Recuperado el 13 de 06 de 2015, de Política de versionado de productos software: http://www.juntadeandalucia.es/servicios/madeja/contenido/libro-pautas/148

Universidad de Sevilla. (s.f.). Gestion de la configuracion. Recuperado el 13 de 06 de 2015, de Versionado de Productos de software: http://www.lsi.us.es/docencia/get.php?id=1090