

**NOMBRE:** Diego Eduardo Celi Alviar

**TUTOR:** Ing. Daniel Guamán.

Información del Documento

|  |  |
| --- | --- |
| Título: | Versionamiento |
| Subtítulo: | Identificación de procesos en Cada Operación. |
| Versión: | 1.0.0 |
| Archivo: | Entregable |
| Autor: | Diego Celi |
| Estado: | Entregable |

Lista de Cambios

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Fecha | Autor | Descripción |
| V 1.0.0 | 03-06-2014 | Diego Celi | Investigación de 4 herramientas que utilicen versiona miento |
|  |  |  |  |

Firmas y Aprobaciones

|  |  |  |  |
| --- | --- | --- | --- |
| Elaborado Por: | Diego Celi  Autor del trabajo | |  |
| Fecha: | [03-06-2014] | Firma: |
| Revisado Por: | Ing. Daniel Guamán | |  |
|  | *TUTOR* |  |  |
|  | [03-06-2014] | Firma: |  |

1. **Identificar al menos 4 herramientas que se utilicen en el versionamiento a nivel de proyectos o SW.**
2. **CVS:** (**Concurrent Versioning System**), es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros, principalmente en el código fuente, que forman un proyecto y permite que distintos desarrolladores se encuentren a gran distancia también colaboren.

**Características:**

* CVS utiliza una arquitectura cliente-servidor: un servidor guarda la(s) versión(es) actual(es) del proyecto y su historial.
* Los clientes se conectan al servidor para sacar una copia completa del proyecto.
* El cliente y servidor se conectan utilizando Internet, pero con el sistema CVS el cliente y servidor pueden estar en la misma máquina.
* El sistema CVS tiene la tarea de mantener el registro de la historia de las versiones del programa de un proyecto solamente con desarrolladores locales.
* Los clientes pueden también comparar diferentes versiones de archivos, solicitar una historia completa de los cambios, o sacar una "foto" histórica del proyecto tal como se encontraba en una fecha determinada o en un número de revisión determinado.
* Los clientes también pueden utilizar la orden de actualización con el fin de tener sus copias al día con la última versión que se encuentra en el servidor.
* CVS también puede mantener distintas "ramas" de un proyecto. Por ejemplo, una versión difundida de un proyecto de programa puede formar una rama y ser utilizada para corregir errores.

1. **Git**: Es un software de control de versiones, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena. Hay algunos proyectos de mucha relevancia que ya usan Git, en particular, el grupo de programación del núcleo Linux.

**Características:**

* Fuerte apoyo al desarrollo no lineal, por ende rapidez en la gestión de ramas y mezclado de diferentes versiones. Git incluye herramientas específicas para navegar y visualizar un historial de desarrollo no lineal.
* Git le da a cada programador una copia local del historial del desarrollo entero, y los cambios se propagan entre los repositorios locales.
* Los almacenes de información pueden publicarse por [HTTP](http://es.wikipedia.org/wiki/HTTP), [FTP](http://es.wikipedia.org/wiki/File_Transfer_Protocol), [rsync](http://es.wikipedia.org/wiki/Rsync) o mediante un protocolo nativo, ya sea a través de una conexión TCP/IP simple o a través de cifrado [SSH](http://es.wikipedia.org/wiki/SSH). Git también puede emular servidores [CVS](http://es.wikipedia.org/wiki/CVS), lo que habilita el uso de clientes CVS pre-existentes y módulos IDE para CVS pre-existentes en el acceso de repositorios Git.
* Los repositorios Subversion y svk se pueden usar directamente con git-svn.
* Gestión eficiente de proyectos grandes, dada la rapidez de gestión de diferencias entre archivos, entre otras mejoras de optimización de velocidad de ejecución.
* Todas las versiones previas a un cambio determinado, implican la notificación de un cambio posterior en cualquiera de ellas a ese cambio (denominado autenticación criptográfica de historial).

1. **Mercurial:** Es un sistema de control de versiones multiplataforma, para desarrolladores de software.

**Características:**

* 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).
* Las principales metas de desarrollo de Mercurial incluyen un gran rendimiento y escalabilidad; desarrollo completamente distribuido, sin necesidad de un servidor; gestión robusta de archivos tanto de texto como binario; y capacidades avanzadas de ramificación e integración, todo ello manteniendo sencillez conceptual.
* Incluye una interfaz web integrada.
* El código fuente se encuentra disponible bajo los términos de la licencia GNU GPL versión 2, lo que clasifica a Mercurial como software libre.

1. **Microsoft Visual SourceSafe:** Es un sistema de control de versiones en el nivel de archivos, que permite a muchos tipos de organizaciones trabajar en distintas versiones de un proyecto al mismo tiempo. Esta funcionalidad es especialmente ventajosa en un entorno de desarrollo de software, donde se usa para mantener versiones de código paralelas. Sin embargo, el producto también se puede utilizar para mantener archivos en cualquier otro tipo de equipo.

**Funciones:**

* Ayuda al equipo a evitar la pérdida accidental de archivos.
* Permite realizar un seguimiento de las versiones anteriores de un archivo.
* Admite la bifurcación, el uso compartido, la combinación y la administración de versiones de archivos.
* Realiza el seguimiento de las versiones de proyectos completos.
* Realiza el seguimiento del código modular (un archivo que se reutiliza, o se comparte, en varios proyectos).

1. **Diseñar una solución software utilizando cualquier lenguaje de programación que permita registrar datos en archivos de una clase persona.**
2. **De las 4 herramientas identificadas, seleccione una para versionar:**

**Documentación en formato Word. (Al menos 4 historias)**

**Documentar código de la solución de software. (Al menos 4 historias)**

**En la codificación documentar o versionar el código.**

1. **Consulte al menos 4 aplicaciones de software las cuales tengan versionamiento, por lo menos 2 niveles (V1, V1.1, V1.2, V2), describir que significa haber pasado del 1 a la 2.**