Sistema de control de versiones

Cristian Adair Ramirez Rodriguez

crisadaramrod@gmail.com

Universidad de la Sierra Sur

28/05/2022

1 Introducción

El control de versiones, también conocido como "control de código fuente", es la práctica de rastrear y gestionar los cambios en el código de software. Los sistemas de control de versiones son herramientas de software que ayudan a los equipos de software a gestionar los cambios en el código fuente a lo largo del tiempo. A medida que los entornos de desarrollo se aceleran, los sistemas de control de versiones ayudan a los equipos de software a trabajar de forma más rápida e inteligente.

2 Caracteristicas

Un sistema de control de versiones debe proporcionar:

1.-Mecanismo de almacenamiento de los elementos que deba gestionar (ej. archivos de texto, imágenes, documentación...).

2.-Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos).

3.-Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto).

3 tipos de sistemas de control de versiones

Podemos clasificar los sistemas de control de versiones atendiendo a la arquitectura utilizada para el almacenamiento del código: locales, centralizados y distribuidos.

* Locales. Los cambios son guardados localmente y no se comparten con nadie. ... * Centralizados. ...

* Distribuidos.

4 Beneficios

Permite el trabajo en paralelo de dos o más usuarios de una aplicación o programa sin que se den por válidas versiones con elementos que entren en conflicto entre sí. La seguridad de un repositorio de código en un servidor es máxima ya que este garantiza la misma mediante diferentes métodos avanzados de ciberseguridad y la creación constante de copias de seguridad. Permite disponer y acceder a un historial de cambios. Cada programador tiene la obligación de señalar qué cambios ha realizado, quién los ha llevado a cabo y también cuando se hicieron. Esto permite un mayor control sobre cada versión, permite corregir cambios y facilita que cada cambio realizado se vea como un nuevo estado. Facilita el entendimiento del proyecto, sus avances y el estado actual de la última versión. Algo posible gracias a que cada pequeño cambio realizado por un programador debe ir acompañado por un mensaje explicativo de la tarea realizada. Así evita a los demás programadores perder tiempo cuestionándose el por qué de los cambios realizados. Y facilita la corrección de errores si los hubiera y son detectados por otro programador.

5 tipos de sistema control de versiones

Sistemas de Control de Versiones Locales Un método de control de versiones, usado por muchas personas, es copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron, si son ingeniosos). Este método es muy común porque es muy sencillo, pero también es tremendamente propenso a errores. Es fácil olvidar en qué directorio te encuentras y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no querías.

Sistemas de Control de Versiones Centralizados El siguiente gran problema con el que se encuentran las personas es que necesitan colaborar con desarrolladores

en otros sistemas. Los sistemas de Control de Versiones Centralizados (CVCS por sus siglas en inglés) fueron desarrollados para solucionar este problema. Estos sistemas, como CVS, Subversion y Perforce, tienen un único servidor que contiene todos los archivos versionados y varios clientes que descargan los archivos desde ese lugar central. Este ha sido el estándar para el control de versiones por muchos años.

Sistemas de Control de Versiones Distribuidos Los sistemas de Control de Versiones Distribuidos (DVCS por sus siglas en inglés) ofrecen soluciones para los problemas que han sido mencionados. En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no solo descargan la última copia instantánea de los archivos, sino que se replica completamente el repositorio. De esta manera, si un servidor deja de funcionar y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios disponibles en los clientes puede ser copiado al servidor con el fin de restaurarlo. Cada clon es realmente una copia completa de todos los datos.

6 Conclusión

En conclusion se puede tener muy en cuenta el uso de un controlador de versiones por un orden y mantener un respaldo guardado. Las actualizaciones van acompañadas de errores y bugs y si alguna version anterior no lo tenia puede ser util para volver a iniciar desde ahí.

References

[1] Qué son los repositorios de código y cuáles son sus beneficios - The Black Box Lab. (2022). Retrieved 7 May 2022, from

https://theblackboxlab.com/2021/02/22/que-son-los-repositorios-de-codigo-y-cuales-son-sus-beneficios/

[2] Git - Acerca del Control de Versiones. (2022). Retrieved 7 May 2022, from https://git-scm.com/book/es/v2/Inicio—Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones

[3] (2022). Retrieved 7 May 2022, from https://closermarketing.es/pros-y-contras-de-la-estrategia-de-repositorio-de-codigo/