**Sistemas de Control de versiones.**

Carlos Daniel Franco Pardo, Dairo Andrés Rincón Hernández, Julián Betancourth Hernández y Oscar Andrés Camacho Aguilar.

Centro de Electricidad, Electrónica y Telecomunicaciones, SENA.

2338322: Tecnología en Análisis y Desarrollo de Sistemas de Información.

Instructor: Hernando Enrique Moreno Moreno.

22 de agosto de 2021.

Sistemas de Control de Versiones.

Resumen.

Este trabajo se presenta con el fin de demostrar lo aprendido en clase sobre los sistemas de control de versiones, para su realización nos apoyaremos con los videos de las respectivas clases, material bibliográfico buscado en la internet y del debate entre los integrantes del grupo.

Palabras claves: sistemas de control de versiones, versión control system.

Sistemas de Control de versiones.

Sistemas de Control de Versiones.

La investigación que se muestran en el presente trabajo tiene como objetivo dar a conocer los conocimientos adquiridos en el tema de los sistemas de control de versiones, teniendo en cuenta que una sociedad que desde hace varios años se viene desenvolviendo mas de manera digital y la creación de software ha venido en aumento se hizo necesario la creación de sistemas de control de versiones para controlar los códigos que se vayan escribiendo. Por lo tanto, nosotros como aprendices debemos estar al tanto de este tipo de tecnología conocer su funcionamiento y diferentes opciones que el mercado actual ofrece, para de este modo poder tener un desempeño eficiente en el ámbito cooperativo y laboral.

Actividades de Apropiación.

1. Definir lo siguientes conceptos.

* Definir que son sistemas de control de versiones.

Un control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo de tal manera que sea posible recuperar versiones especificas más adelante.

* Definir e identificar los diferentes tipos de sistemas de control de versiones. (locales, centralizados y distribuidos).

**Sistema de control de versiones locales:** Sistema donde se utiliza una simple base de datos, en la que se lleva el registro de todos los cambios realizados a los archivos.

**Sistema de control de versiones centralizados:** Este sistema tiene un único servidor que contiene todos los archivos versionados y varios clientes que descargan los archivos desde ese lugar, ha sido el estándar para el control de versiones por muchos años. Esta configuración ofrece varias ventajas especialmente a las locales, por ejemplo, todas las personas saben hasta que cierto punto está trabajando los otros colaboradores del proyecto, los administradores tienen control sobre que puede hacer cada usuario y es mucho más fácil administrar que estar lidiando con bases de datos locales en cada cliente. La desventaja más obvia seria que si el servidor se cae durante una hora durante esa hora nadie podrá colaborar ni guardar cambios.

**Sistema de control de versiones distribuidos:** Con este tipo de sistema las personas 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 puede ser copiado al servidor con el fin de restaurarlo, cada clon es realmente una copia completa de los datos.

* Identificar los sistemas de control de versiones más usados en el mercado.

CVS.

SourceSafe.

ClearCase.

Darcs.

Bazaar.

Plastic.

SCM.

Git

Mercurial.

Perforce.

Fossil SCM.

* Identificar las semánticas para el manejo de versiones mas usados en el mercado.

**Número Mayor:** el número principal o primer número, indica la versión actual de la interfaz pública del paquete. Esto debe incrementarse cada vez que realice un cambio que requiera que los usuarios existentes de su paquete actualicen su trabajo.

**Numero menor:** El numero menor o segundo número describe la versión funcional actual del software. Esto se incrementa cada vez que se agrega una nueva funcionalidad, pero no altera la interfaz de su paquete. Notifique a los usuarios que sea realizado un cambio significativo, pero el paquete sigue siendo totalmente compatible con el número menor anterior.

**Numero de Parche:** El numero de parche o tercer numero se incrementa cada vez que se realiza un pequeño cambio que no afecta ni la interfaz publica ni a la funcionalidad general del paquete. Esto se usa más comúnmente para corregir errores. Los usuarios siempre deberían poder actualizar la última versión del parche son dudarlo. La mayoría de los proyectos deben usar 1.0.0 como su versión inicial.

* Identificar buenas prácticas de calidad en manejo de un sistema de control de versiones.

Suele ser muy útil la generación de informes con los cambios introducidos entre dos versiones, informes de estado, marcado con nombre identificativo de la versión.

Conclusiones.

El sistema de control de versiones más que muy útil es casi indispensable para que los proyectos que lleven principalmente programación de código fuente se lleve un control del flujo del trabajo que se realiza, como para saber en qué punto de desarrollo va el proyecto evitar que lleguen en lo más mínimo errores a la rama principal y de esta manera el proyecto o repositorio se termine de una manera rápida y cumpliendo con las expectativas y requerimientos del cliente.

Referencias.

<https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones>

. <https://es.wikipedia.org/wiki/Control_de_versiones>

<https://diarioinforme.com/que-es-el-control-de-versiones-semantico/>