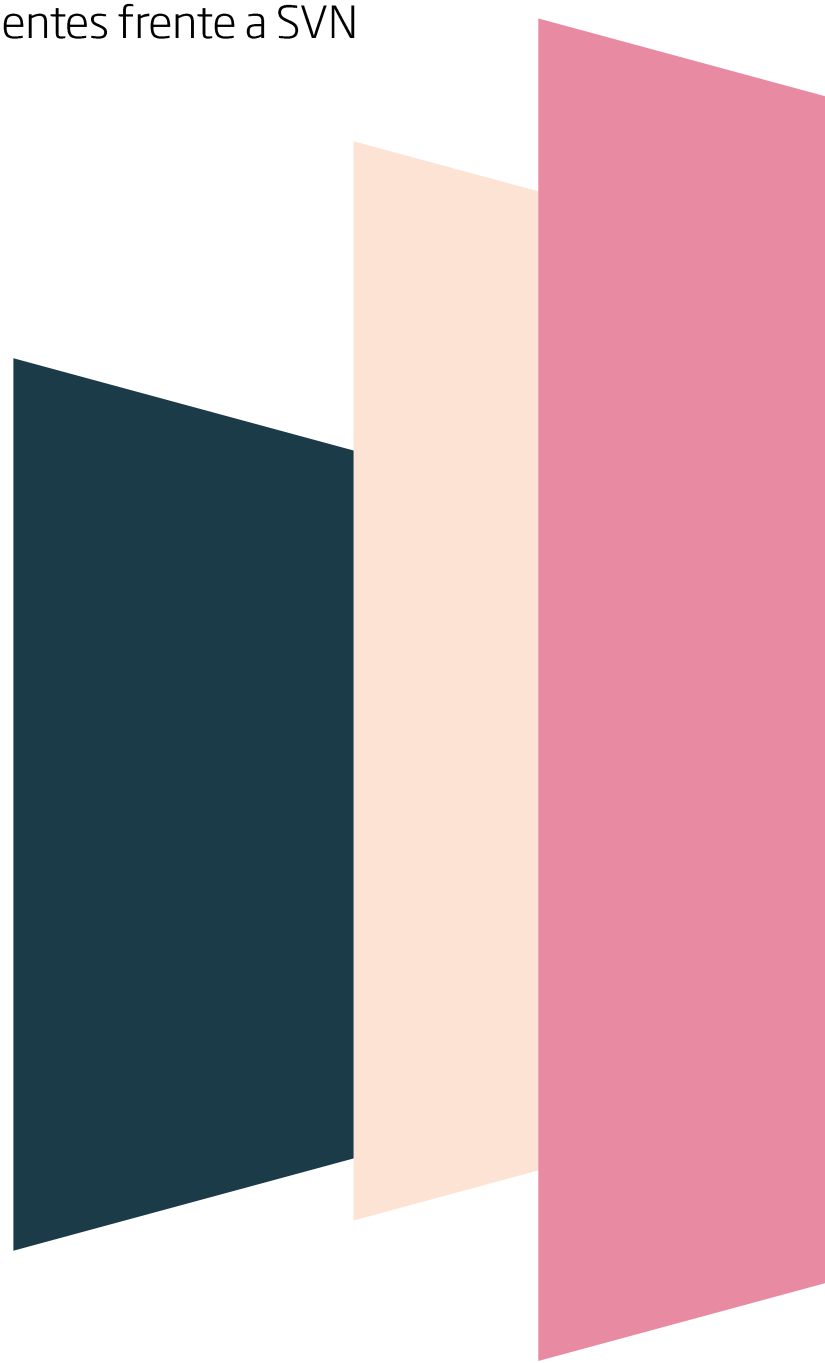
minsait

An Indra company

Git vs SVN

¿Qué es?, Ventajas e Inconvenientes frente a SVN

Octubre 2019



Índice

1 ¿Qué es GitHub?

GitHub es una plataforma que permite alojar proyectos para el desarrollo colaborativo utilizando para ello el sistema de control de versiones Git. Su principal uso es la creación de código fuente para software. El contenido de los proyectos que se encuentran en GitHub de forma general se almacena típicamente de forma pública, aunque utilizando una cuenta de pago, también permite hospedar repositorios privados.

2 Sistema de control de versiones de código.

Un sistema de control de versiones, permite a los usuarios llevar un control de los cambios realizados en proyectos y además colaborar en ellos.

Al utilizar este sistema, los desarrolladores pueden trabajar juntos en un mismo proyecto, con tareas separadas, las cuales luego pueden combinar.

Algo también muy importante, es que pueden ver el historial de cambios realizados, saber quién los realizó, cuándo y por qué.

Además, también es posible volver a un cierto momento en la historia de un proyecto o utilizar/cambiar código que fue creado en cualquier punto de éste.

SVN y GIT, si bien son ambos sistemas de control de versiones, tienen una diferencia bastante significante. SVN es un sistema de control de versiones centralizado (CVCS, Centralized Version Control System), mientras que Git es un sistema de control de versiones distribuido (DVCS, Distributed Version Control System).

Esto es algo muy importante a tener en cuenta, ya que cual elijas determinará el modo en el que trabajes en tu proyecto.

3 ¿Cuál es la diferencia entre DVCS y CVCS?

Un sistema de control de versiones centralizado opera con la idea de que hay una sola copia del proyecto a la cual los desarrolladores realizar el commit de los cambios, y un solo lugar en el cual todas las versiones de un proyecto se encuentran guardadas.

Un sistema de control de versiones distribuido, por otro lado, trabaja con el principio de que cada desarrollador «clona» el repositorio del proyecto al disco duro de su dispositivo.

Una copia del proyecto es guardada en cada máquina local, y los cambios deben ser subidos (pushed) y bajados (pulled) hacia y desde el repositorio online para actualizar la versión que cada desarrollador tiene en su máquina local.

Como vemos, si bien funcionan de forma similar, ambos cuentan con una gran diferencia, que según lo que necesites o la forma en la que trabajes, puede que necesites un sistema distribuido o centralizado.

Veamos ahora cuáles son las diferencias entre ambos para que puedas determinar cual es el mejor a utilizar en tu caso.

Ahora que sabemos qué es cada uno y cómo funciona, veamos algunas ventajas y desventajas de trabajar con ellos.

4 ¿Qué es SVN?

SVN o también conocido como Subversion es uno de los CVCS más populares y utilizados en el mercado.

SVN como ya hemos mencionado, utiliza un sistema centralizado, todos los archivos, así como también el historial de éstos son guardados en un servidor central.

Es decir que los desarrolladores deben realizar el «commit» de los cambios directamente al repositorio en el servidor central.

4.1 ¿Cómo funciona SVN?

SVN fue diseñado originalmente como una interfaz de línea de comandos, es decir que para trabajar con él debes abrir una terminal y tipear los comandos.

Para poder trabajar con SVN, se necesitan dos elementos importantes

El servidor, el cual tiene todas las versiones de los archivos

Una copia local de los archivos, la cual será tu computadora

Con SVN realizarás los cambios en tu ambiente de trabajo y luego, una vez que se encuentre todo funcional, se realiza el commit de los cambios al servidor SVN, el cual es llamado repositorio.

Cada vez que se realiza un commit, SVN se encarga de realizar una nueva versión, para que sea posible regresar a viejos commits.

Normalmente siempre se trabaja en las últimas versiones, pero siempre que lo necesites podrás regresar a una vieja.

5 Entonces ¿Qué es Git?

Git como ya hemos mencionado anteriormente, es un DVCS, es decir que utiliza múltiples repositorios: un repositorio central y una serie de repositorios locales.

Los repositorios locales son una copia exacta del repositorio completo, incluyendo el historial de cambios.

5.1 ¿Cómo funciona Git?

La forma de trabajo en Git es bastante similar a SVN, con la diferencia de que contiene un paso extra: para crear una nueva funcionalidad, debes crear una copia del repositorio en tu maquina local.

Luego de eso, trabajarás en ella de la misma forma, y al finalizar realizarás el push de los cambios desde el repositorio local al central.

5.1.1 ¿Por qué GIT es superior a sus competidores como controlador de versiones de código?

- Trabajo en equipo. Permite que varios desarrolladores trabajen al mismo tiempo y en paralelo en un proyecto con un acceso compartido, pero sin estar físicamente cerca, así como identificar qué usuario y cuándo ha realizado cada modificación.
- Mayor autonomía. Cada desarrollador cuenta con una copia local de todo el proyecto y de los cambios generados, lo que le permite trabajar de forma individual y a su propio ritmo, en cualquier momento y lugar.
- La velocidad es otro de los puntos fuertes de Git frente a otros sistemas controladores de versiones, ya que necesita menos capacidad de procesamiento y gestión al poder realizar las operaciones en local.
- Sin conexión a la red. El desarrollador puede trabajar o enviar cambios de código al sistema sin necesidad de estar conectado a Internet o a cualquier otra red, al disponer de un repositorio local.
- Estructura en árbol. Esta característica hace posible que los desarrolladores puedan trabajar en diferentes ramas de un proyecto, pero sin modificar en el código base principal, facilitando así que puedan probar nuevas funcionalidades sin miedo a cometer equivocaciones, ya que siempre pueden dar marcha atrás y volver a versiones anteriores.
- Es un sistema escalable.
- Colaboración. Permite, a través de plataformas como Gitlab, GitHub o Bitbucket, que funcionan como repositorios remotos, colaborar en proyectos de otros desarrolladores.
- Permite comparar, fusionar o restaurar versiones de una aplicación y contar con una copia del código fuente para volver atrás ante cualquier imprevisto.
- Es software libre y open source.

- Es multiplataforma por lo que se puede usar para crear repositorios locales en todos los sistemas operativos más comunes, como Windows, Linux o Mac.
- Lo utilizan empresas tecnológicas de referencia como Google, Facebook o Nefflix para controlar las versiones de código fuente sus proyectos.

5.2 Fortalezas de GITHUB

- Seguimiento de errores.
- Búsqueda rápida.
- Cuenta con una potente comunidad de desarrolladores en todo el mundo.
- Permite descargar como archivo el código fuente.
- Posibilita la importación en Git, SVN o TFS.
- Puedes personalizar cualquier servicio host en la nube.

6 GIT vs SVN

6.1 Ventajas de GIT frente a SVN

6.1.1 Sistema Distribuido

El hecho de ser un sistema distribuido significa que múltiples repositorios redundantes y ramificaciones son conceptos de primera clase de la herramienta.

En un sistema distribuido como Git, cada usuario tiene una copia completa guardada del proyecto, haciendo que el acceso a la historia de cada uno sea extremadamente rápido.

Algo muy importante es que debido a ello permite que puedas utilizarlo con poca o sin conexión a Internet.

También significa que cada usuario tiene una copia completa del repositorio, es decir que, si por algún caso alguno queda corrupto, solo los cambios que hayan sido únicos para ese repositorio serán perdidos.

En SVN por otro lado, solamente el repositorio central contiene el historial completo. Por lo cual los usuarios deben comunicarse a través de la red al repositorio central para obtener el historial de los archivos.

Por otro lado, si el repositorio central es perdido por algún fallo del sistema, éste debe ser restablecido desde un backup y esto puede provocar que cambios sean perdidos.

Dependiendo de las políticas de backup que hayan, podrían incluso perderse semanas de trabajo.

6.1.2 Control de Acceso

Al ser un sistema distribuido, no hay que otorgar acceso a otras personas para que puedan utilizar las funciones de control de versiones. En vez de eso, es el dueño del repositorio el que decide a qué cambios realizar el merge y de quién.

Subversión por otro lado sí controla el acceso, por ejemplo, el usuario requiere acceso para realizar commits.

En git, los usuarios pueden tener control de versión de su propio proyecto, mientras que el proyecto principal está controlado por el propietario del repositorio.

6.1.3 Manejo de Ramas (Branches)

En Git utilizar branches es muy común y fácil, mientras que se puede considerar que en SVN es un proceso un poco más engorroso y no tan habitual.

De hecho, la queja más común sobre SVN es lo tan tedioso que es trabajar con ramas. En SVN, las branches se crean como directorios, algo que a muchos desarrolladores no les gusta.

Además, en SVN 1.6, se introdujo un concepto llamado conflicto de árboles, los cuales son causados por cambios en la estructura de los directorios y ocurren con frecuencia.

Y debido a ellos las complicaciones para realizar commits entre branches se hace más compleja.

6.1.4 Performance

Dado que git trabaja con un repositorio local, no hay latencia en la mayoría de las operaciones, exceptuando push y fetch, en las cuales sí necesitarás conectarte al repositorio central.

6.2 Ventajas de SVN vs GIT

6.2.1 Historial Permanente

Con Subversion siempre obtendrás exactamente la misma información de tu repositorio, tal y como estaba en el pasado.

Así como también puedes rastrear todos los cambios de un archivo o carpeta, debido a que el historial en Subversion es permanente.

En Git por otro lado, puede que el historial de un archivo/directorio sea perdido. Git no se preocupa del rastro o la historia precisa de cada archivo en los repositorios.

Por ejemplo, el renombrar un archivo o el comando «git rebase» hacen difícil el encontrar el historial «verdadero» de nuestros repositorios.

Ya que, en caso de renombrar un archivo o realizar un git rebase, la historia de los archivos involucrados puede ser perdida.

6.2.2 Un único repositorio

Dado que SVN solamente permite tener un repositorio, no debemos preocuparnos por dónde algo está guardado.

En caso de necesitar un backup o querer buscar algo, no nos quedará duda de que todo lo que necesitemos se encuentra en el repositorio central.

Dado que Git trabaja con repositorios distribuidos, puede que sea más difícil saber qué cosas están ubicadas dónde.

6.2.3 Control de acceso

Dado que Subversion tiene un repositorio central, es posible especificar allí el control de lectura y escritura y será forzado en todo el proyecto.

6.2.4 Archivos Binarios

Los sistemas de control de versiones tienen como idea que la mayoría de los archivos que serán versionados son fusionables.

Es decir, que debería de ser posible fusionar dos cambios simultáneos realizados en un archivo. Este modelo es llamado Copy-Modify-Merge y tanto Git como SVN lo utilizan.

El único problema, es que esto generalmente no es aplicable a archivos binarios, y es por ello que Subversion brinda soporte para el modelo Bloquear-Modificar-Desbloquear para estos casos.

Por otra parte, Git no admite bloqueos de archivos exclusivos, lo cual hace que sea más difícil para empresas con proyectos donde existen muchos archivos binarios no fusionables.

En caso de querer utilizar git con archivos binarios, deberás especificarle cuáles de ellos lo son.

minsait

An Indra company

Persona de contacto
arquitecturaDSP@minsait.com

Avda. de Bruselas 35
28108 Alcobendas,
Madrid, España
T +34 91 480 50 00
F +34 91 480 50 80

www.minsait.com