SERVICION NACIONAL DE APRENDIZAJE SENA

APRENDIZ

GIOVANNY EDUARDO PÉREZ TORRES

TEMA

HERRAMIENTA Y TECNOLOGIA DE VERSIONAMIENTO

INSTRUCTORA

MARIETH PERPIÑAN

2024

CUCUTA, NORTE DE SANTANDER

INTRODUCCION

La gestión eficiente del código fuente es un componente fundamental en el desarrollo de software. En este contexto, la integración continua emerge como una práctica primordial que permite a los equipos de desarrollo automatizar la integración de cambios en el código, garantizando una entrega más rápida y confiable de software. Para llevar a cabo la integración continua de manera efectiva, es importante contar con herramientas y tecnologías de control de versiones de manera estable para que faciliten la colaboración entre los miembros del equipo y garanticen la estabilidad y la calidad del código.

En este informe técnico, se analizarán las herramientas y tecnologías de versionamiento que serán implementadas en nuestro entorno tanto actuales como futuros. Se explorarán las opciones disponibles, se justificará la elección de la herramienta seleccionada y se detallarán los objetivos que se esperan alcanzar con su implementación.

OBJETIVO

1. Selección de la herramienta de control de versiones: Investigar y evaluar diferentes herramientas de control de versiones para determinar cual se adapta mejor a las necesidades del equipo de desarrollo y al entorno de integración continua.

2. Configuración e integración de la herramienta seleccionada: Definir los procedimientos y configuraciones necesarios para integrar la herramienta de control de versiones seleccionada en el proceso de integración continua. Esto incluira la configuración de flujos de trabajo, la automatización de pruebas y la gestión de ramas de código.

3. Optimización del flujo de desarrollo: Establecer practicas y directrices para el uso eficiente de la herramienta de control de versiones, con el objetivo de mejorar la colaboración entre los miembros del equipo, reducir los conflictos de fusión y garantizar la calidad del codigo a lo largo del ciclo de vida del desarrollo de software.

HERRAMIENTAS Y TECNOLOGIAS DE VERSIONAMIENTO

El versionamiento es el proceso de asignación de un nombre, código o número único, a un software para indicar su nivel de desarrollo, siendo esto fundamental para su desarrollo, llevando un registro de los cambios realizados en el código fuente y en sus componentes a lo largo del tiempo.

Pero, para elegir las herramientas primero se deben identificar las herramientas y tecnologías de control de versiones más adecuadas para el entorno de desarrollo en cuestión. Algunas de las opciones más comunes incluyen Git, Subversion (SVN), Mercurial, entre otras.

Para esto veremos las herramientas que podríamos llegar a usar y sus características:

Figura 1

Git es un sistema de control de versiones distribuido, diseñado para manejar desde proyectos pequeños hasta grandes con rapidez y eficiencia. Permite a los desarrolladores trabajar en paralelo, mantener un historial completo de cambios y fusionar fácilmente el trabajo realizado por múltiples colaboradores.

Distribuido: Cada usuario tiene una copia local completa del repositorio con historial completo.

Velocidad: Operaciones son rápidas debido al uso eficiente de recursos.

Ramas: Facilita y promueve el uso de ramas para desarrollo paralelo y colaboración.

Integridad de datos: Usa un sistema de checksums para garantizar la integridad de los datos.

Flexibilidad: Adaptable a diferentes flujos de trabajo y estilos de desarrollo.

Figura 2

Subversion (SVN):

Subversion es un sistema de control de versiones centralizado, donde un servidor central almacena el código y los desarrolladores sincronizan su trabajo con este servidor. Aunque menos utilizado en la actualidad, todavía es popular en algunos entornos.

Centralizado: Repositorio central donde los usuarios sincronizan el código.

Historial completo: Almacena el historial completo de archivos, pero solo como snapshots completos.

Operaciones complejas: Algunas operaciones pueden ser más complejas que en sistemas distribuidos como Git.

Conflictos: Puede haber más conflictos al trabajar con ramas debido a la naturaleza centralizada.

Figura 3

Mercurial es otro sistema de control de versiones distribuido, similar a Git. Permite a los usuarios tener copias locales completas de un repositorio y trabajar de forma independiente. Aunque menos común que Git, es valorado por su simplicidad y facilidad de uso.

Distribuido: Similar a Git, permite que cada usuario tenga una copia local completa.

Simplicidad: Diseñado para ser más fácil de usar que Git, con comandos más intuitivos.

Rendimiento: Generalmente más lento que Git en operaciones grandes y complejas.

Extensiones: Ampliable con extensiones para agregar funcionalidades adicionales.

Figura 4

GitHub utiliza Git, un sistema de control de versiones distribuido, como su base. Permite a los desarrolladores colaborar en proyectos, realizar seguimiento de cambios, revertir a versiones anteriores y gestionar ramas de código de manera eficiente. Además del control de versiones, GitHub ofrece funcionalidades adicionales como seguimiento de problemas, integración continua, gestión de proyectos y wikis, lo que lo convierte en una plataforma completa para el desarrollo de software colaborativo.

Esta plataforma de desarrollo colaborativo basada en Git ofrece diversas cualidades y características como:

Control de versiones: Utiliza Git para controlar y gestionar versiones de código, lo que permite a los desarrolladores trabajar en proyectos de forma colaborativa y mantener un historial completo de cambios.

Almacenamiento de proyectos: Permite a los desarrolladores guardar y compartir proyectos de forma gratuita, facilitando la colaboración y el trabajo en equipo.

Seguimiento de problemas: Ofrece herramientas para gestionar problemas, errores y solicitudes de funciones, lo que facilita la comunicación y la resolución de problemas en los proyectos.

Integración continua: Permite configurar integración continua para automatizar el proceso de construcción, pruebas y despliegue de aplicaciones.

Gestión de proyectos: Proporciona herramientas para gestionar tareas, organizar el trabajo y realizar un seguimiento del progreso del proyecto.

Wikis: Permite crear wikis para documentar proyectos y compartir información sobre el desarrollo y la colaboración.

Colaboración: Facilita la colaboración entre desarrolladores mediante la posibilidad de realizar solicitudes de extracción, revisar código y realizar comentarios en el código de otros.

Una vez vistas a cada una de las herramientas y comparando sus ventajas y desventajas, la mejor opción por el momento para usar seria GitHub, y no solo radica a su popularidad en este último tiempo, sino a ciertos factores como es la colaboración, siendo que permite a los equipos de desarrollo trabajar juntos en proyectos, facilitando la colaboración y la comunicación entre miembros del equipo.

Además de que ayuda a registra todos los cambios realizados en un proyecto, lo que facilita la gestión de versiones y la identificación de errores, también ofrece un historial completo de cambios en un proyecto, integración continúa facilitando la automatización y la gestión de problemas, errores y solicitudes de funciones de manera eficiente, facilitando la colaboración y la resolución de problemas.

Todos estos son factores con los que podremos contar a la hora de hacer nuestro proyecto, de manera que todos los integrantes del equipo puedan trabajar a la par, sin necesidad de tener una copia externa, siendo bastante impráctico a la hora demovilizar el proyecto.

Por estas razones se ha decidido utilizar GitHub como herramienta de versionamiento para le proyecto y futuros trabajos grupales.

FUENTES

https://es.wikipedia.org/wiki/Subversion_(software)

https://magmax.org/blog/mercurial-subversion/

https://es.scribd.com/document/630272622/Herramientas-para-el-control-de-versiones

https://openwebinars.net/blog/que-es-github/

https://gist.github.com/erlinis/57a55dfb0337f5cd15cd

https://es.wikipedia.org/wiki/GitHub

FUENTES

https://aulab.es/articulos-guias-avanzadas/56/principales-caracteristicas-de-git

https://es.wikipedia.org/wiki/Subversion_(software)

https://www.kranio.io/blog/descubriendo-git-caracteristicas-y-ventajas