Sistema de control de versiones, repositorios de código fuente

Griselda Medina Avendaño

medinagris7@gmail.com

Universidad de la Sierra Sur

2022/04/11

1 Introducción

A lo largo del desarrollo de software es importante llevar un historial y registro de cada modificación, mejora o cambio al código fuente, para ello es de mucha utilidad hacer uso de herramientas tales como los sistemas de control de versiones a través de los cuales se puede dar un seguimiento al código generado y además proporciona ventajas como el poder crear ramas, para que cada integrante de un equipo la trabaje, etc. En este documento se abordaran las ventajas de estos sistemas de control de versiones y así mismo se hablará de los repositorios de código fuente.

2 Sistema de control de versiones

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.El software de control de versiones realiza un seguimiento de todas las modificaciones en el código en un tipo especial de base de datos. Si se comete un error, los desarrolladores pueden ir hacia atrás en el tiempo y comparar las versiones anteriores del código para

ayudar a resolver el error, al tiempo que se minimizan las interrupciones para todos los miembros del equipo. (Abrutsky,)

2.1 Ventajas

1. ***Un completo historial de cambios a largo plazo de todos los archivos.***Esto quiere decir todos los cambios realizados por muchas personas a lo largo de los años. Los cambios incluyen la creación y la eliminación de los archivos, así como los cambios de sus contenidos. Las diferentes herramientas de VCS difieren en lo bien que gestionan el cambio de nombre y el movimiento de los archivos. Este historial también debería incluir el autor, la fecha y notas escritas sobre el propósito de cada cambio. Tener el historial completo permite volver a las versiones anteriores para ayudar a analizar la causa raíz de los errores y es crucial cuando se tiene que solucionar problemas en las versiones anteriores del software. Si se está trabajando de forma activa en el software, casi todo puede considerarse una ”versión anterior” del software.(Leal, Sosa, & Leal,)

2. ***Creación de ramas y fusiones.*** i se tiene a miembros del equipo trabajando al mismo tiempo, es algo evidente; pero incluso las personas que trabajan solas pueden beneficiarse de la capacidad de trabajar en flujos independientes de cambios. La creación de una ”rama” en las herramientas de VCS mantiene múltiples flujos de trabajo independientes los unos de los otros al tiempo que ofrece la facilidad de volver a fusionar ese trabajo, lo que permite que los desarrolladores verifiquen que los cambios de cada rama no entran en conflicto. Muchos equipos de software adoptan la práctica de crear ramas para cada función o quizás para cada publicación, o ambas. Existen muchos flujos de trabajo diferentes que los equipos pueden elegir cuando deciden cómo utilizar la creación de las ramas y las fusiones en VCS.

3. ***Trazabilidad.***Ser capaz de trazar cada cambio que se hace en el software y conectarlo con un software de gestión de proyectos y seguimiento de errores como Jira, además de ser capaz de anotar cada cambio con un mensaje que describa el

propósito y el objetivo del cambio, no solo te ayuda con el análisis de la causa raíz y la recopilación de información. Tener el historial anotado del código a tu alcance cuando estás leyendo el código, intentando entender lo que hace y por qué se ha diseñado así, puede permitir a los desarrolladores hacer cambios correctos y armoniosos que estén en línea con el diseño previsto a largo plazo del sistema. Esto puede ser especialmente importante para trabajar de manera eficaz con código heredado y es esencial para que los desarrolladores puedan calcular el trabajo futuro con precisión. (Ruiz-Bertol & Zarazaga-Soria,)

3 Repositorio de código

Es el lugar en el que se almacena y se puede realizar la distribución del código de una aplicación o un programa. Este debe ser un servidor seguro que utiliza sistemas de control de versiones. Debe contener las diferentes versiones de la aplicación o programa, disponiendo de un historial con los cambios realizados sobre el original y sobre cada nueva versión. Además, debe permitir poder revertir esos cambios. Y permitir que la aplicación o programa pueda ser utilizado en paralelo por diferentes usuarios al mismo tiempo, en la misma o en sus diferentes versiones.(Doria, del Prado, & Haustein,)

3.1 Ventajas

1. 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.

2. 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.

3. 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.(Texier, De Guisti, & Gordillo,)

4 Conclusiones

En el desarrollo deben hacerse uso de estas herrmientas para agilizar el trabajo y tener referentes sobre lo que se esta realizando en el código, así mismo es útil en caso de que se necesite hacer un revert ya que se cuenta con un respaldo en caso de que se necesite echar mano de el.

References

Abrutsky, A. (2016). Implantación de un sistema de control de versiones.

Doria, M. V., del Prado, A. M., Haustein, M. C. (2015). Repositorios digitales y software open source. *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*(15), 73–81.

Leal, E. T., Sosa, C. M., Leal, D. A. T. (2012). Revisión de los sistemas de control de versiones utilizados en el desarrollo de software. *Ingenierías USBMed*, *3*(1), 74–81.

Ruiz-Bertol, F. J., Zarazaga-Soria, F. J. (2007). El control de versiones en el aprendizaje de la ingeniería informática: Un enfoque práctico. *Actas de las XIII Jornadas de Enseñanza Universitaria de Informática*.

Texier, J., De Guisti, M., Gordillo, S. (2014). El desarrollo de software dirigido por modelos en los repositorios institucionales. *Dyna*, *81*(184), 186–192.