Universidad Mayor de San Simón
Facultad de Ciencia y Tecnologia
Carrera Ingenieria de Sistemas
Carrera Licenciatura en Informatica

SISTEMA WEB DE CONTROL DE VERSIONES, PARA PROYECTOS DESARROLLO DE SOFTWARE "SUPAY"

Univ. José Emmanuel Valdivia Ignacio
Univ. Jonathan Daniel Huayta Donaire

23 de julio de 2014

http://scesi.org

Índice

1. Introducción

Un sistema de control de versiones es vital para la eficiencia en cuanto a trabajo en equipo, el no usarlo tiene muchas consecuencias en el proyecto que se desarrolla. Al tener a muchas personas trabajando sobre un mismo proyecto cada una realiza cambios de acuerdo a sus necesidades y/o requerimientos, esto genera que cada persona involucrada en el proyecto tenga una "versión"propia, y al momento de llegar a juntar cada una de estar partes se produce perdida de información o funcionalidad, es peor cuando se trabaja con grandes equipos porque en cierto punto llegarían a tener un rompecabezas que no se pueda armar, porque cada parte será excluyente de otra.

Otro aspecto importante es no saber qué persona realiza cambios y de qué tipo, el no tener un control de los cambios que realiza cada integrante del proyecto, este punto es crítico tanto para equipos centralizados como para equipos no centralizados, ya que el desconocimiento de los cambios que realizan integrantes del equipo, conlleva muchos problemas al momento de desarrollar el proyecto.

Control de versiones es la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas

2. Antecedentes

El proceso de desarrollo de software es una tarea compleja, en la cual participan multiples actores, eso implica que los mismos, realizaran modificaciones en distintas etapas, agregarán y eliminaran elementos, sin que exista una coordinación efectiva y eficiente. Estos cambios tienen diversos orígenes como: nuevas necesidades del cliente, crecimiento o reducción del negocio, multiples factores externos; para afrontar este problema existe un conjunto de actividades para gestionar los cambios a lo largo del ciclo de vida del software de computadora, el GCS (Gestión de Configuración del Software) surge como respuesta a la necesidad de realizar dichas actividades de forma confiable, mismo que tiene como principal función el control de versiones(cambios).

Al realizar GCS surgen varias interrogantes:

- ¿Cómo identifica y gestiona una organización las diferentes versiones

que existe de un programa de forma que se pueda introducir cambios eficientemente?

- ¿Cómo controla la organización los cambios que vayan a realizarse?
- ¿Cómo garantizamos que los cambios se han llevado a cabo de manera exitosa?
- ¿Qué mecanismos se usa para comunicar al resto del equipo de los cambios realizados?

Estas preguntas nos llevan a la definición de tareas que debe cumplir el GCS: Identificación, Control de versiones, Auditorias de configuración, Generación de informes.

3. Justificación

El proyecto tiene como base crear un ambiente donde se pueda gestionar todos los cambios que se van realizando a lo largo del proceso de desarrollo de software, para que varias personas que trabajan sobre un mismo proyecto puedan tener un espacio donde alojar los proyectos que les brinde la información, ademas control total sobre el ciclo de desarrollo, pudiendo disponer de todas las versiones que posee el proyecto, la existencia de multiples versiones en distintas etapas del proyecto es necesaria, ya que de realizar modificaciones al proyecto sin existir un backup(respaldo) existe el riesgo mayor de afectar funciones vitales del mismo, quizas incluso cuando las mismas se encontraban en estado optimo.

A la fecha existen muchas aplicaciones que cubren esta necesidad, pero su uso no es tan simple, ya que para hacer uso de un sistema de cualquier tipo en este caso un sistema de control de versiones requiriere estar familiarizado con su uso, conocer sus comandos, su funcionamiento. Por otro lado es obligatorio tener instalado un software en la máquina del miembro del equipo del cliente, para poder interactuar con la aplicación, lo que provoca una dependencia hacia este.

La implementación de un sistema de este tipo presenta múltiples ventajas, tanto para equipos de desarrollo de software bien constituidos como para personas que participen en proyectos de indole general que atraviesen por procesos combinados, ya que pone a disposición de ellos una herramienta útil al momento de tener control sobre la gestión de los procesos de desarrollo del proyecto haciendo mucho más amigable la aplicación de versionamiento.

4. Planteamiento del problema

A la fecha existen muchas aplicaciones de control de versiones, pero su uso no es tan simple, ya que para hacer uso de un sistema de cualquier tipo en este caso un sistema de control de versiones requiere estar familiarizado con este, conocer sus comandos, su funcionamiento. Por otro lado es obligatorio tener instalado un software en la máquina del miembro del equipo del cliente, para poder interactuar con la aplicación.

En síntesis el uso de aplicaciones de control de versiones presenta dos grandes problemas: la complejidad de su uso y la instalación de software únicamente que realice dicha tarea, lo que ocasiona confusión entre las personas que desean usar versionamiento y optan por no usarlo en el desarrollo de sus proyectos.

5. Formulación del Problema

Confusión al desarrollar un proyecto por no tener un seguimiento adecuado de los cambios realizados, ocasionando retrasos en la consecución del proyecto, ya sea que se desarrolle un proyecto en equipo o de forma individual.

6. Objetivos

6.1. Objetivo General

Fomentar el uso de un sistema de control de versiones, implementando una apliacion web con un conjunto de funciones básicas que presenta un GIT de tal forma que pueda facilitar a los usuarios el versionamiento de sus diversos proyectos, y familiarizarse mas con el concepto de versionar

6.2. Objetivos Específicos

- Desarrollar módulo de almacenamiento de nuevas repositorios de los proyectos creados.
- Desarrollar el módulo de administración de repositorios.
- Desarrollar módulo de control de acceso a los proyectos, para permitir trabajar sobre una misma versión.

7. Hipótesis o Idea a Defender o preguntas científicas y Tareas de investigación

Con la implementacion de un sistema web de control de versiones, se desea fomentar el uso de control de versiones desde etapas tempranas del proceso de aprendizaje de los individuos asi tambien responder las preguntas:

- ¿De que manera se puede incentivar el uso de versionadores en el desarrollo de un proyecto?
- ¿Cómo hacer mas accesible el uso de versionadores?
- ¿Cómo disminuir la desorganización originada por los continuos cambios en las versiones de un proyecto?

8. Novedad y aporte científico

Este proyecto intenta cubrir dos aspectos: eliminar la dependencia de instalar un software en las pc de los miembros del equipo sino más bien que se pueda interactuar con un servicio web, para poder asi realizar cambios a una versión seleccionada, y no obligar al usuario a conocer los comandos necesarios para interactuar con un sistema de control de versiones, planteando una interfaz amigable en la cual cualquier usuario pueda mantener un control de versiones de una manera simple solo subiendo los ficheros a un servidor y llenando un formulario.

9. Diseño metodológico y teórico

El proyecto se basa en el sistema de control de versiones GIT como nucleo central para el desarrollo del sistema . Y básicamente el proyecto se estructura de la siguiente forma:

- **Control de Usuarios** Implementamos nuestro propio modulo de gestion de usuarios para que puedan acceder a los repositorios y crear nuevos repositorios, previamente logueado cada usuario tiene las opciones de crear nuevos proyectos o ser colaborador de uno ya existente.
- **Conexión entre GIT y PHP** Para evitar el uso de GIT directamente se uso librerias especificas para poder usar GIT desde la aplicacion web, tambien se hizo uso directo de comandos del sistema para ciertas tareas y para otras la libreria GLIP

- **Interfaz Gráfica** Diseñar una interfaz amigable que presente todas las opciones básicas de un sistema de control de versiones, y que tambien presente opciones para poder descargar una version de un proyecto y para pudir subir una nueva version.

10. Desarrollo del proyecto

El proyecto hace uso del lenguaje PHP en su versión 5.3 o superior, el proyecto esta vercionada en GitHub

Al ser un programa escrito en PHP se hizo uso del framework Zend 1.12, debido a que cada componente de Zend está construido con una baja dependencia de otros componentes. Esta arquitectura débilmente acoplada permite a los desarrolladores utilizar los componentes por separado. A menudo se refiere a este tipo de diseño como üse-at-will"(uso a voluntad).

Tambien en el diseño se hizo uso del paradigma de PROGRAMACION ORIENTADA A OBJETOS, ya que el mismo presenta multiples ventajas al momento de realizar la arquitectura y posterior implementacion, debido a que todo es un objeto tenemos la posibilidad de manejar cada módulo del sistema de forma independiente del mismo, con lo cual ganamos flexibilidad al momento de modificar o aumentar funcionalidades, un alto nivel de abstracción que nos brinda una vision objetiva de los requerimientos reales del sistema.

Para la persistencia de los datos se hizo uso del ORM Doctrine 2, ya que es el mas popular.

El mapeo objeto-relacional (más conocido por su nombre en inglés, Object-Relational mapping, o sus siglas O/RM, ORM, y O/R mapping) es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y la utilización de una base de datos relacional, utilizando un motor de persistencia. En la práctica esto crea una base de datos orientada a objetos virtual, sobre la base de datos relacional. Esto posibilita el uso de las características propias de la orientación a objetos (básicamente herencia y polimorfismo).

Una característica de Doctrine es el bajo nivel de configuración que necesita para empezar un proyecto. Doctrine puede generar clases a partir de una base de datos existente y después el programador puede especificar relaciones y añadir funcionalidad extra a las clases autogeneradas. No es necesario generar o mantener complejos esquemas XML de base de datos como en otros frameworks.

Otra característica importante de Doctrine es la posibilidad de escribir

consultas de base de datos utilizando un dialecto de SQL denominado DQL (Doctrine Query Language)

Se hizo uso de estas herramientas debido a que integran el paradigma orientado a objetos, y se hizo uso de varios patrones siendo los que más resaltan:

- MVC: El modelo vista controlador (MVC) es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario1 2 . Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

  De manera genérica, los componentes de MVC se podrían definir como sigue:

  - Modelo: Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la vista aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al modelo a través del controlador.
  - Vista: Presenta el modelo (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario) por tanto requiere de dicho modelo la información que debe representar como salida.
  - Controlador: Responde a eventos (usualmente acciones del usuario) e invoca peticiones al modelo cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a su vista asociada si se solicita un cambio en la forma en que se presenta de 'modelo' (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos),

por tanto se podría decir que el 'controlador' hace de intermediario entre la vista y el modelo

- DAO: Un Data Access Object (DAO, Objeto de Acceso a Datos) es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo.

  La mayoría de aplicaciones requieren de interacción con una base de datos, ya sea del tipo relacional u orientada a objetos y en la mayoría de los casos la persistencia de datos está mezclada con la lógica del negocio.

  Lo que supone un problema a la hora de querer cambiar el motor de base de datos.

  CARACTERISTICAS: DAO oculta completamente los detalles de implementacion de la fuente de datos a sus clientes, de esta manera cuando la capa de lógica de negocio necesite interactuar con la base de datos ésta utilizará los métodos ofrecidos por DAO.

11. Conclusiones y recomendaciones

El sistema se encuentra aun en fase de desarrollo, pero ya se cuenta con un prototipo estable, dadas las características, el sistema tiene grandes posibilidades de convertirse en un sistema de control de versiones de uso simple y masivo.

El uso de aplicaciones de control de versiones puede resultar muy complejo y tedioso al momento de usarlo, existiendo además multitud de servicios sin embargo con el desarrollo de esta aplicación web se facilita enormemente su uso y lo más importante al ser una aplicación web no necesita la instalación de software extra y esta accesible en todo momento.

Este aplicación no se limita al desarrollo de proyectos de software, su uso se podría expandir a controlar las versiones todo tipo de proyectos tanto educativo o empresariales.

Referencias

[1] "Ingeniería de Software" Ed. Pearson Addison Wesley 7ma edición. España.

[2] “Ingeniería de Software. Un enfoque práctico ” Ed. McGraw-Hill Interamericana. 6ta edición España.

[3] “sistemas de control de Versiones” licencia CreativeCommonsAttributionLicense.