PROTOTIPO WEB PARA CONTROL DE VERSIONES SOFTWARE Y DOCUMENTACIÓN EN LÍNEA, APLICADO AL SERVIDOR DE LA ESCUELA DE INGENIERÍA DE SISTEMAS – UIS, CORMORÁN

ANGELICA MARÍA NIÑO DÍAZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2008

PROTOTIPO WEB PARA CONTROL DE VERSIONES SOFTWARE Y DOCUMENTACIÓN EN LÍNEA, APLICADO AL SERVIDOR DE LA ESCUELA DE INGENIERÍA DE SISTEMAS – UIS, CORMORÁN

Trabajo de Grado para optar al titulo de
Ingeniera de sistemas

ANGELICA MARÍA NIÑO DÍAZ

Director
Ing. MANUEL GUILLERMO FLÓREZ BECERRA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2008

DEDICATORIA

A Dios, por estar siempre conmigo.

A mi Mamá, por comprenderme, ayudarme, aguantarme y escucharme aún cuando no entiende lo que le cuento. Gracias por su infinito amor.

A mi Papá, por su inmenso cariño, dedicación y su esperanza en mí.

A mi Hermano, por ser amigo, guía y ejemplo.

A mis tios que me han apoyado siempre.

A Andrés por ser mi cómplice incondicional.

A mis amigos, por hacer de mi paso por la universidad la mejor experiencia personal.

ANGÉLICA MARÍA NIÑO

AGRADECIMIENTOS

A la Universidad Industrial de Santander.

A la Escuela de Ingeniería de Sistemas e Informática.

Al profesor Manuel Guillermo Flórez Becerra, Director de este proyecto.

Al profesor Fernando Ruiz Diaz, Director de la EISI.

A los encargados del servidor Cormorán, Monica, Laura y Juan Carlos.

Contenido

Lista de Figuras

Lista de tablas

Glosario

Añadir:	Comando de Subversion que se utiliza para añadir un fichero o un directorio al repositorio.
Apache:	Apache Web Server: Software libre para el manejo de servidores Web.
BDB:	Base de datos Berkeley: Un soporte de base de datos muy probado para los repositorios, que no puede utilizarse en unidades compartidas de red. Por defecto para repositorios anteriores a la versión 1.2.
Conflicto:	Es la situación en la que los cambios realizados por uno o mas usuarios se solapan con los cambios realizados por otro usuario al mezclar las copias de trabajo dentro de un repositorio.
Copia de trabajo:	Una replica local de los archivos y directorios almacenados en el repositorio.
FSFS:	Sistema de ficheros FS: Un sistema de ficheros propietario de Subversion que se utiliza como soporte de los repositorios. Se puede utilizar en unidades compartidas de red. Por defecto para los repositorios a partir de la versión 1.2.
GPL:	GNU GPL: General Public License o licencia pública general. Es una licencia creada por la Free Software Foundation a mediados de los 80, para la distribución, modificación y uso del software libre.
GUI:	Graphical User Interface: Interfaz gráfica del usuario.
HTTP:	Hypertext Transfer Protocol: Protocolo de transferencia de hipertexto.
HTTPS:	Hypertext Transfer Protocol Secure: Protocolo seguro de transferencia de hipertexto (versión segura del http).

Rama:	Línea de desarrollo de un proyecto.
Repositorio:	Un repositorio es un lugar central donde se almacenan y mantienen los datos. Un repositorio puede ser un lugar donde se encuentran múltiples bases de datos o ficheros para distribuirlos en una red, o un repositorio puede ser un lugar directamente accesible por el usuario sin tener que viajar por una red.
Revisión BASE:	La revisión base actual de un fichero o una carpeta en su *copia de trabajo*. Esta es la revisión en la que estaba el fichero o la carpeta cuando se hizo la última operación de obtener, actualizar o confirmar. La revisión BASE normalmente no equivale a la revisión HEAD.
Revisión HEAD:	La última revisión de un fichero o una carpeta en el *repositorio*.
Shell:	Parte fundamental de un sistema operativo encargada de ejecutar las órdenes básicas para el manejo del sistema. También se denomina núcleo o kernel.
SSL:	Secure Socket Layer: Capa de conexión segura.
Tags:	Raíz de las etiquetas del proyecto.
Trunk:	Rama de desarrollo principal.
Versión:	Estado asociado a una clave, normalmente un número.

RESUMEN

TÍTULO [1]

PROTOTIPO WEB PARA CONTROL DE VERSIONES SOFTWARE Y DOCUMENTACIÓN EN LÍNEA, APLICADO AL SERVIDOR DE LA ESCUELA DE INGENIERÍA DE SISTEMAS – UIS, CORMORÁN

AUTORA

ANGÉLICA MARÍA NIÑO DÍAZ [2]

PALABRAS CLAVE

Sistema de control de versiones, Documentación en línea, prototipo, Programación Extrema

DESCRIPCIÓN

Este proyecto desarrollado para el servidor Cormorán de la Escuela de Ingeniería de Sistemas (EISI) de la Universidad Industrial de Santander, tiene como propósito administrar proyectos y documentación alojados en él. Permite centralizar proyectos e información en general, facilitar el desarrollo distribuido de aplicaciones y mantener la documentación del servidor actualizada y en línea.

Para llevar a cabo estas operaciones se utiliza la metodología de control de versiones. El control de versiones es la administración de múltiples revisiones de una misma unidad de información. Es comúnmente utilizado en ingeniería y desarrollo de software para manejar el historial de documentos digitales tales como código fuente de aplicaciones y otra información crítica que puede ser trabajada en un grupo de personas.

Durante el desarrollo, se realizó el análisis de una herramienta de control de versiones adecuada para el servidor, la instalación y configuración de la misma y se implementó un portal Web bajo la metodología de desarrollo denominada programación extrema; En este portal se administra la creación de repositorios, el acceso a los mismos por parte de los usuarios y la catalogación de documentos. A través de él se ofrecen los servicios de búsqueda de documentación, actualización y visualización de proyectos, y acceso a los manuales de los clientes gráficos del sistema de control de versiones. Finalmente, mediante su implantación se presta un servicio importante y novedoso a los miembros de la EISI y se brinda un mayor nivel de seguridad al servidor Cormorán.

[1] Proyecto de Grado en la Modalidad de Investigación

[2] Facultad de Ingenierías Físico – Mecánicas, Escuela de ingeniería de sistemas e informática, Director: Manuel Guillermo Flórez Becerra.

ABSTRACT

TITLE [3]

WEB PROTOTYPE TO VERSION CONTROL SOFTWARE AND ONLINE DOCUMENTATION, APPLIED TO THE SERVER OF SYSTEM ENGINEERING SCHOOL OF THE INDUSTRIAL UNIVERSITY OF SANTANDER UIS - CORMORÁN.

AUTHOR

ANGÉLICA MARÍA NIÑO DÍAZ [4]

KEY WORDS

Version Control system, Online Documentation, Prototype, Extreme Programming.

DESCRIPTION

This project developed for the *Cormorán* server at the School of Systems Engineering (EISI) of the Industrial University of Santander, has as purpose, to manage projects and documentation, content on it. Allows centralizing projects and general information, makes easier the development of distributed applications and maintain the server documentation updated and online.

To accomplish these operations the version control methodology is used. Version control is the management of multiple revisions of the same unit of information. It is most commonly used in engineering and software development to manage history data of digital documents like applications source code and other critical information that may be worked on by a team of people.

During development, it was performed the analysis of the version control tool suitable for the server, installation and configuration of the same and implemented a web portal under the development methodology called extreme programming; In this website is administrated the repositories creation, user access to those repositories and documents cataloguing. Services such as documentation research, projects updating and visualization, and access to manuals of graphic user interface of version control system are provided through this. Finally, after this implementation is given to EISI members an important and new service that gives a higher level of security in the *Cormorán* Server.

[3] Graduation work in the investigation modality

[4] Physics and Mechanics Sciences Faculty, System Engineering School, Director: Manuel Guillermo Flórez Becerra

1 Introducción

Durante el desarrollo de software normalmente se encuentra la necesidad de trabajar en un grupo, en donde varios desarrolladores deben acceder al mismo código fuente. La coordinación de las modificaciones a éste, de forma manual es una tarea tediosa que exige paciencia y una cantidad considerable de tiempo. En la búsqueda de una solución a este problema se encuentra la metodología de control de versiones, que además de ser una excelente alternativa, provee otras características para apoyar el desarrollo de todo tipo de proyectos y documentación en general.

Los sistemas de control de versiones son un conjunto de herramientas para gestionar carpetas y archivos con el fin de mantener una sola versión actualizada, evitando conflictos entre las versiones de varios desarrolladores.

Estos sistemas proveen la posibilidad de hacer un historial de cambios, volver a una versión anterior de un proyecto con un clic, hacer pruebas con la seguridad que da poder deshacer los cambios, y muchas otras características que ayudan en el desarrollo de las tareas en un grupo o en un proyecto individual. Esto influye además en el mejoramiento de la calidad del producto final y facilita tareas como la reingeniería e ingeniería inversa en la retoma de un proyecto.

Durante los últimos años, las comunidades virtuales de desarrollo han utilizado y difundido ampliamente varias herramientas de control de versiones por los excelentes resultados que han obtenido. En estas comunidades, personas que no se conocen, geográficamente dispersas, y que hablan distintos idiomas, pueden llevar a cabo proyectos exitosos. Donde si bien estas aplicaciones no reemplazan la comunicación que debe existir, si se convierte en una técnica crítica, sin la cual, seria prácticamente imposible integrar los cambios de tantos desarrolladores. Uno de los aspectos característicos de estos proyectos consiste en la liberación rápida de resultados, para lo cual se utiliza la metodología de programación extrema, por los principios en los que esta se basa.

Durante el transcurso de este proyecto se desarrollara un portal Web para apoyar la labor de administración del sistema de control de versiones y realizar las búsquedas sobre la documentación almacenada en el servidor Cormorán gestionada por el mismo sistema.

El objetivo del portal es convertirse en una herramienta de trabajo para los desarrolladores de aplicaciones de nuestra escuela y en una fuente de información importante para toda la comunidad.

2 Aspectos Generales

2.1 Justificación

Actualmente en la escuela de ingeniería de sistemas e informática de la universidad se realizan gran cantidad de proyectos de grado enfocados a la creación de aplicaciones software de utilidad para la comunidad, algunas de estas aplicaciones se ponen al servicio de la escuela a través del servidor Cormorán. Sin embargo aunque cada proyecto tiene una documentación complementaria a él, esta, no se encuentra disponible para los usuarios, y muchas veces no esta debidamente actualizada, ya sea por los cambios realizados por los creadores o por las correcciones de configuración que son necesarias en el servidor.

Se cuenta con una documentación estática del funcionamiento y configuración del servidor, pero este documento no puede ser fácilmente actualizado y las búsquedas de información en él no resultan tan sencillas como deberían ser, por lo que no se utiliza de forma constante para la resolución de dudas e inquietudes.

Respecto al uso de tecnologías para el desarrollo de las aplicaciones y su respectiva documentación, no se utilizan técnicas efectivas para desarrollar actividades dentro de grupos interdisciplinarios. En el mercado, existen diferentes soluciones para hacer más fácil la comunicación de los cambios en el código fuente y la documentación de los proyectos, pero a pesar de esto, no se utilizan en nuestro medio dichas herramientas ni siquiera entre los grupos de estudiantes que trabajan sobre un mismo proyecto y no se tiene un control sobre los versiones de las aplicaciones.

Cuando un estudiante o un grupo desea realizar un trabajo de grado, sobre un proyecto desarrollado anteriormente, generalmente empieza de cero, desperdiciando gran cantidad de tiempo y recursos, ya sea porque se dificulta entender el código fuente del proyecto por una indebida documentación, o porque esta documentación no se encuentra disponible para su uso.

El presente proyecto pretende crear un prototipo, que sirva como base para la solución de los problemas anteriormente planteados, con el fin de ayudar a la comunidad en el desarrollo de los futuros proyectos y en la actualización de las aplicaciones presentes dentro del servidor de la escuela, así como contribuir a una cultura donde se resalte la importancia de la documentación y no se trate como un simple requisito para la creación de los proyectos.

2.2 Objetivo General

Mediante el uso de una herramienta para control de versiones:

- Facilitar el desarrollo de aplicaciones software mediante la metodología de control de versiones en un ambiente de trabajo colaborativo e interdisciplinario.

- Diseñar e implementar un portal Web para la gestión y mantenimiento en línea de los manuales de documentación desarrollados para el servidor Cormorán.

2.3 Objetivos Específicos

- Analizar diferentes herramientas de libre distribución para el control de versiones de software e identificar cual es la más apropiada para las necesidades de la escuela.

- Implantar la herramienta para el servicio de control de versiones software para desarrollo de proyectos que administre de forma segura, código, documentación y elementos multimedia.

- Implementar repositorios de versiones software y diseñar una base de datos que soporte indexamientos, búsquedas, y control de acceso de usuarios al repositorio de documentación en línea en el servidor cormorán.

- Organizar la documentación en línea para la administración del sistema del servidor cormorán fundamentándose en la documentación existente.

- Mostrar a través de una página web los elementos que permitan al administrador conocer los detalles de la configuración interna del servidor y a los desarrolladores de proyectos software la verificación de manuales, código y otros documentos parte del desarrollo de sus proyectos.

- Realizar pruebas de validación y verificación correspondientes para garantizar el correcto funcionamiento de las herramientas y software desarrollado.

2.4 Impacto

Impacto social

El impacto de este proyecto consiste en ser conciente de las necesidades de documentación no solo para los usuarios del servidor, sino también para los desarrolladores de aplicaciones software, haciendo más cómodo manipular los cambios y la evolución del código y la documentación, y poder revisar lo que hizo otro miembro de un grupo, mientras que se concentran solo en los cambios y no en toda la información preexistente. Esto ayuda a la dirección en un grupo jerárquico, o a la colaboración entre pares y grupos interdisciplinarios.

Impacto Económico

El impacto económico de este proyecto se verá reflejado en la mejora de la productividad en la creación de nuevos proyectos y en el manejo dado a los existentes. Los estudiantes tanto de pregrado, como de postgrado, tendrán más facilidades para crear nuevas versiones del software que ha sido desarrollado en otros proyectos, para que se adapten a las nuevas necesidades que cada día surgen dentro de nuestro ámbito tanto académico como social.

Impacto Técnico

Tecnológicamente se dará paso al conocimiento de herramientas de control sobre dichas aplicaciones, así como de la documentación de los proyectos, aprendiendo a tomar experiencia de las soluciones dadas por otros a problemas comunes.

2.5 Viabilidad

Viabilidad financiera

Este proyecto es viable económicamente puesto que se cuenta con el personal necesario, el hardware adecuado, y las herramientas de software son de libre distribución lo que hace que su costo no sea un obstáculo, además el beneficio que se obtiene hace que la relación costo/beneficio sea muy buena.

Viabilidad Técnica

De acuerdo a los conocimientos impartidos en nuestra escuela, un estudiante de ingeniería de sistemas esta en la capacidad técnica de realizar las actividades programadas en el transcurso de este proyecto, tanto en la parte de análisis, como en la programación y realización de pruebas. Dado que se han realizado muchos estudios en esta materia la documentación existente es

abundante en diversos idiomas, lo cual facilita la búsqueda de la información necesaria para su realización.

Viabilidad Social

Con este proyecto no se afecta el medio ambiente, ni las condiciones de vida de las personas que tienen contacto con el.
En el ámbito educativo, este proyecto brinda la posibilidad de poner en práctica los conocimientos adquiridos durante la carrera y conocer más a fondo temas de informática como el desarrollo de aplicaciones Web, diseño y gestión de base de datos.

2.6. Alcances del proyecto

Para el desarrollo de este proyecto se deben tener en cuenta las ventajas y beneficios de su implantación, que son entre otras:

- Proporcionar a la comunidad de desarrolladores de software de nuestra escuela un espacio adecuado para el almacenamiento centralizado de sus aplicaciones en proceso de producción, brindando una herramienta para facilitar la acción de los grupos, la organización del proyecto y el mejoramiento de la calidad del producto final.

- Brindar una herramienta estable para el control de las versiones tanto de código fuente como de documentación y archivos en general, manteniendo un registro de los cambios realizados en el proceso de desarrollo, según la metodología empleada.

- Promover la utilización de estándares de programación, comunicación, e interacción en el desarrollo de los proyectos, a fin de tener como resultado productos finales de alta calidad.

- La promoción de políticas de seguridad sobre los proyectos y documentos en general, la encriptación de contraseñas y el uso de sesiones, que permite a los usuarios tener más confianza sobre el uso de los servicios ofrecidos.

- La implementación de un lugar unificado para mantener la documentación referente al servidor cormorán, su estructura, configuración, los servicios prestados a la comunidad y los proyectos que sobre él son alojados.

- La disminución de la papelería al utilizar medios seguros de almacenamiento electrónico tanto como para consultas internas y externas como para el respaldo de seguridad.

3 Marco Teórico

Las siguientes son las herramientas utilizadas para el desarrollo del portal web:

3.1 PHP

PHP: Hypertext Preprocessor. Es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Es de código abierto que funciona casi en cualquier sistema operativo y tiene funciones nativas para la mayoría de Bases de datos sin utilizar un mediador para la conexión con lo cual se obtendrá mayor velocidad de respuesta.

Después de comparar los tres lenguajes más utilizados para el desarrollo de este tipo de proyectos y con ayuda de la siguiente información se estableció PHP como el lenguaje adecuado para presentar el proyecto.

Tabla 1. Comparación entre PHP, ASP y JSP

Característica	PHP	ASP	JSP
Velocidad de la conexión a base de datos	Alta (Conexión directa por contar con controladores de bases de datos)	Media (A través de ADO, ActiveX Data Objects y usando fuentes ODBC)	Baja (Usando JDBC)
Disponibilidad de recursos	Alta	Alta	Media
Costo	Bajo	Alto	Bajo
Curva de aprendizaje	Baja	Baja	Media

Gómez, Gutiérrez Juddy Alexandra y Miranda, Mercado Oscar Aníbal. Prototipo software soportado en Internet para la creación administración y mantenimiento de sitios web dinámicos, orientado a grupos y centros de investigación de la Universidad Industrial de Santander. Tesis de grado. Pág. 29

PHP fue concebido a finales de 1994 por un programador independiente (Rasmus Lerdorf) para uso personal. La primera versión estuvo disponible al público a principios de 1995. Posteriormente fue reescrito y renombrado como PHP/FI version 2, y un tiempo después (1997) el proyecto se convirtió en grupal y llamado PHP version 3. Las versiones utilizadas actualmente son las 4.X y 5.

Es de fácil uso y posee gran similitud con los más comunes lenguajes de programación estructurada, como el C y el Perl.

Con PHP es posible procesar información de formularios, generar páginas Web con contenidos dinámicos, o mandar o recibir cookies.

Algunas de las bases de datos soportadas por este lenguaje son:

- Adabas
- InteBase
- PostgreSQL
- FilePro
- FrontBase
- Solid
- IBM DB2
- MSQL
- Sybase
- Informix
- MySQL
- Velocis
- Empress
- Dbase
- Oracle

Las páginas web son documentos que se pueden visualizar a través de Internet y que pueden incluir texto, imágenes, enlaces de hipertexto y otros medios. Para poder visualizar estas páginas, es necesario tener un programa que sea capaz de comunicarse con el servidor web, a través de un protocolo de comunicación llamado http.

HTML es un lenguaje de composición de documentos y especificaciones de enlaces de hipertexto que define la sintaxis y coloca instrucciones especiales que no muestra el navegador, aunque si le indica como desplegar el contenido del documento, incluyendo texto, imágenes y otros medios soportados. HTML también le indica como hacer un documento interactivo a través de ligas especiales de hipertexto, las cuales conectan diferentes documentos.

Las páginas de HTML únicamente tienen contenido estático, pero cuando se emplean otros lenguajes como PHP, se puede incluir contenido dinámico, es decir contenido que varia dependiendo de la información suministrada por el usuario. La mayoría de las veces, la información que se genera de manera dinámica, proviene de una Base de Datos.

3.2 PostgreSQL

La mayoría de aplicaciones Web requieren un manejador de base de datos para obtener la información necesaria que estas manejan de una forma más eficiente.

PostgreSQL es un sistema de gestión de base de datos de tipo objeto-relacional basada en POSTGRES, desarrollada en el departamento de informática de la

Universidad de California (Berkeley). POSTGRES fue un proyecto dirigido por el Profesor Michael Stonebraker y contó con el apoyo de diversos organismos y empresas americanas.

PostgreSQL es de código abierto muy extendido, especialmente en los desarrollos basados en Unix/Linux (algunas distribuciones como Red Hat y Fedora lo instalan por defecto), aunque existen versiones para plataformas Windows.

Este sistema incorporó gran parte de los conceptos de objetos relacionales que hoy en día incluyen algunos gestores de base de datos comerciales. Los gestores tradicionales permiten la existencia de una colección de relaciones identificadas por un nombre, las cuales poseen atributos a los que se le asocia un determinado tipo de datos. Los sistemas de base de datos actuales permiten tipos de datos tales como números en coma flotante, enteros, cadenas de caracteres, monedas y fechas; pero como es bien sabido, esto no basta para satisfacer las necesidades de las aplicaciones del futuro.

Por esta razón, PostgreSQL proporciona un conjunto de nuevos conceptos que permitirán a los usuarios ampliar de forma cómoda sus aplicaciones.

Entre las ventajas y características propias de postgreSQL se pueden destacar las siguientes:

- Soporta casi todas las construcciones SQL, incluso extensiones orientadas a objetos, en este sentido es el motor de gestión de base de datos de código abierto más avanzado. Soporta construcciones como Transacciones, Vistas, y Triggers.
- Amplia conectividad a través de lenguajes como C, C++, Perl, Python, TCL, PHP, JDBC y ODBC
- Diversidad de herramientas disponibles.
- Herramientas de administración: copias de seguridad y restauración.
- Conexión segura de acceso a datos sobe SSL y SSH
- Clientes, como por ejemplo pgaccess, y pgadmin, interfaces graficas para la administración.
- Nuevos tipos de datos como:
    - Tipos geométricos.
    - Tipos de dirección de red.
    - Tipo de datos Bolean.
    - Manejo de tipos de datos sobre fechas.
    - Tipo serial.
    - Tipo text.
- Funciones y operadores para los tipos de datos soportados.

Para el caso del servidor Cormorán esta instalado el paquete Postgresql-8.0.0.

Para el desarrollo de este proyecto se escogió este gestor de base de datos especialmente por los tipos de datos que soporta y la conveniencia de un modelo relacional.

3.3 Apache Web Server

Un servidor Web es un programa que implementa el protocolo http (hypertext transfer protocol). Este protocolo esta diseñado para transferir lo que llamamos hipertextos, páginas Web o páginas HTML (Hypertext markup language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonido.

El servidor Web se encarga de mantenerse a la espera de peticiones http llevada a cabo por un cliente http que solemos conocer como navegador. El navegador realiza una petición al servidor y este le responde con el contenido que el cliente solicita. El cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página; el servidor tan solo se limita a transferir el código de la página sin llevar a cabo ninguna interpretación de la misma.

Apache es el servidor Web más extendido en el mercado. Nació como un conjunto de parches del servidor NCSA. Es un servidor http de código abierto para plataformas Unix/Linux, Windows y otras, que implementan la noción de sitio virtual.

Apache proporciona multitud de características:

- Respeto a los estándares (HTTP/1.1, etc.)
- Multiplataforma. Aunque tradicionalmente es utilizado en Unix/Linux, ha sido portado a otros sistemas como Windows, AS/400 y OS/2.
- Estabilidad, mediante técnicas como el virtual hosting.
- Soporte a lenguajes de programación en el lado servidor (PHP, JSP, Perl, etc.)
- Seguridad mediante SSL, control de acceso y autentificación (no incluido en la configuración inicial)
- Precio, confiabilidad, rendimiento, construcción modular.

Además de las anteriores características con la elección del servidor Web se busca dar la mayor de funcionalidades al sistema de control de versiones que se utilizara en el servidor.

3.4 Modelo del proceso del software

Para el desarrollo de un proyecto es necesario tener una metodología acorde con la naturaleza del mismo. El modelo del proceso de desarrollo es una forma de ordenar de una forma sistemática todas las etapas que deben ser tenidas en cuenta para que el proyecto tenga un alto nivel de calidad.

La metodología o planificación del trabajo es normalmente conocida como el ciclo de vida del software. Es posible decir que existen tantas metodologías como desarrolladores, pero todas ellas están basadas en modelos ya definidos, y estos a su vez han ido evolucionando a través del tiempo a medida que proyectos desarrollados con estos modelos son exitosos.

En general estos métodos pueden ser clasificados en lineales o secuenciales, circulares y evolutivos. La metodología clásica se refiere a los modelos lineales en los que el trabajo dividido por etapas debe ser realizado de una forma secuencial en que para pasar a la siguiente fase es necesario tener totalmente terminada la anterior. Modificaciones a estos modelos implican que esa secuencia no debe ser tan estricta y que algunos de los pasos deben ser realizados en varias etapas. En los modelos circulares se aplica la misma secuencia de pasos pero al finalizar esta, implica que los resultados deben ser la situación inicial de una nueva secuencia. Los modelos evolutivos son semejantes a los circulares, sin embargo estos permiten cambios en las actividades, implican que el proyecto vaya creciendo en el tiempo.

Dentro de los modelos Secuenciales se destacan el de Cascada Pura y el Desarrollo rápido de aplicaciones DRA. Dentro de los evolutivos están el Prototipado Simple, Prototipado Evolutivo, Entrega por etapas, Modelo en espiral, Programación Extrema, y Proceso Unificado de desarrollo.

Para el desarrollo del presente proyecto el modelo de desarrollo escogido fue el de Programación Extrema. A continuación se presentan sus características, las mismas por el cual es el más adecuado para esta aplicación en particular.

3.4.1 Programación Extrema

La metodología XP ó Xtreme Programing es una modificación de otras metodologías de desarrollo iterativas, que se está haciendo muy popular gracias a sus prácticas, sus características generales son las siguientes:

- Realimentación entre el cliente y el equipo de desarrollo. Esto se realiza a través de historias de usuario en las cuales el usuario establece los requerimientos a través de descripciones breves de las características que debe tener el sistema.
- Comunicación fluida entre todos los participantes.

- Simplicidad en las soluciones implementadas.
- Coraje para enfrentar los cambios.

3.4.2 Practicas XP

La metodología recomienda unas prácticas que deben ser utilizadas en lo posible mientras el proyecto y los participantes lo permitan.

- El juego de la planificación.

Es un espacio frecuente de comunicación entre el cliente y el programador. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración. Esta práctica se puede ilustrar como un juego, donde existen dos tipos de jugadores: Cliente y Programador. El cliente establece la prioridad de cada historia de usuario, de acuerdo con el valor que aporta para el negocio. Los programadores estiman el esfuerzo asociado a cada historia de usuario. Se ordenan las historias de usuario según prioridad y esfuerzo, y se define el contenido de la entrega y/o iteración, apostando por enfrentar lo de más valor y riesgo cuanto antes. Este juego se realiza durante la planificación de la entrega, en la planificación de cada iteración y cuando sea necesario reconducir el proyecto.

- Entregas pequeñas.

La idea es producir rápidamente versiones del sistema que sean operativas aunque obviamente no cuenten con toda la funcionalidad pretendida para el sistema pero si constituyan un resultado de valor para el negocio.

- Metáfora.

En XP no se enfatiza en la definición temprana de una arquitectura estable para el sistema. Dicha arquitectura se asume evolutiva y los posibles inconvenientes que se generarían por no contar con ella explícitamente en el comienzo del proyecto se solventan con la existencia de una metáfora. El sistema es definido mediante una metáfora o conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema.

- Diseño Simple.

Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto. La complejidad innecesaria y el código extra debe ser removido inmediatamente. Kent Beck, el creador y máximo exponente de la metodología, afirma que en cualquier momento el diseño adecuado para el software es aquel que: supera con éxito todas las pruebas, no tiene lógica duplicada, refleja claramente la

intención de implementación de los programadores y tiene el menor número posible de clases y métodos.

- Pruebas.

La producción de código está dirigida por las pruebas unitarias. Las pruebas unitarias son establecidas antes de escribir el código y son ejecutadas constantemente ante cada modificación del sistema. Los clientes escriben las pruebas funcionales para cada historia de usuario que deba validarse. En este contexto de desarrollo evolutivo y de énfasis en pruebas constantes, la automatización para apoyar esta actividad es crucial.

- Refactorizacion.

La refactorización es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. La refactorización mejora la estructura interna del código sin alterar su comportamiento externo. No se puede imponer todo en un inicio, pero en el transcurso del tiempo este diseño evoluciona conforme cambia la funcionalidad del sistema. Para mantener un diseño apropiado, es necesario realizar actividades de cuidado continuo durante el ciclo de vida del proyecto.

- Programación en parejas.

De ser posible, toda la producción de código debe realizarse con trabajo en parejas de programadores. Las principales ventajas de introducir este estilo de programación son: muchos errores son detectados conforme son introducidos en el código, por consiguiente la tasa de errores del producto final es la más baja, los diseños son mejores y el tamaño del código menor, los problemas de programación se resuelven más rápido, se posibilita la transferencias de conocimientos de programación entre los miembros del equipo, varias personas entienden las diferentes partes del sistema, los programadores conversan mejorando así el flujo de información y la dinámica del equipo, y finalmente los programadores disfrutan más su trabajo.

- Propiedad colectiva del código.

Cualquier programador puede cambiar cualquier parte del código en cualquier momento. Esta práctica motiva a todos a contribuir con nuevas ideas en todos los segmentos del sistema, evitando a la vez que algún programador sea imprescindible para realizar cambios en alguna porción de código.

- Integración continúa.

Cada pieza de código es integrada en el sistema una vez que este lista. Así el sistema puede llegar a ser integrado y construido varias veces en un mismo día. Todas las pruebas son ejecutadas y tienen que ser aprobadas

para que el nuevo código sea incorporado definitivamente. La integración continua a menudo reduce la fragmentación de los esfuerzos de los desarrolladores por falta de comunicación sobre lo que puede ser reutilizado o compartido.

- Cliente in-situ.

El cliente tiene que estar presente y disponible todo el tiempo para el equipo. Gran parte del éxito del proyecto XP se debe a que es el cliente quien conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada. La comunicación oral es más efectiva que la escrita, ya que esta última toma mucho tiempo en generarse y puede tener más riesgo de ser mal interpretada.

- Estándares de programación.

XP enfatiza la comunicación de los programadores a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación. Los estándares de programación mantienen el código legible para los miembros del equipo, facilitando los cambios.

3.5 Control de Versiones de software

Un sistema de control de versiones es un software que administra el acceso a un conjunto de ficheros, y mantiene un historial de cambios realizados. El control de versiones es útil para guardar cualquier documento que cambie con frecuencia, como un manual, o el código fuente de un programa.

Normalmente consiste en una copia maestra en un repositorio central, y un programa cliente con el que cada usuario sincroniza su copia local. Esto permite compartir los cambios sobre un mismo conjunto de ficheros. Además, el repositorio guarda registro de los cambios realizados por cada usuario, y permite volver a un estado anterior en caso de necesidad.

Un sistema de control de versiones debe proporcionar:

- Un mecanismo de almacenaje de cada uno de los ítems que deba gestionarse (archivos de texto, imágenes, documentación...)
- Posibilidad de modificar, mover, o borrar cada uno de los elementos.
- Histórico de las acciones realizadas con cada elemento pudiendo volver a un estado anterior dentro de ese historial.

Una posible clasificación de los sistemas de control de versiones es la siguiente:

3.5.1 Sistemas de Control Centralizados

Los sistemas de control centralizados se basan en un repositorio único central, al que todos los desarrolladores se conectan para reportar cambios. Allí aparece el concepto de copia de trabajo, que es sobre la cual los desarrolladores aplican las modificaciones, que después se sincronizaran con el repositorio. Los cambios en este sistema están ligados directamente al tiempo y están organizados en orden cronológico.

Estos sistemas tienen un modelo Cliente-Servidor, preferiblemente el servidor debe estar encendido todo el tiempo y debe ser configurado con un sistema de autenticación y permisos, que algunos sistemas de control de versiones traen integrados.

Generalmente hay dos tipos de Sistemas de control centralizados según el modelo que utilizan para la entrega de los cambios al servidor. El modelo Lock-Modify-Unlock, que consiste como su nombre lo indica en bloquear el archivo que se va a modificar antes de empezar a hacer los cambios, y cuando estos son terminados nuevamente desbloquear el archivo. El segundo modelo Copy-Modify-Merge propone que las modificaciones se hagan sobre una copia del archivo que una vez son terminados los cambios son mezclados con el original guardado en el repositorio.

3.5.2 Sistemas de control Distribuidos

La característica más importante de un sistema de control distribuido, como su nombre lo indica, es que no existe un punto central de desarrollo sino que los repositorios están distribuidos y descentralizados en diversos equipos que pueden o no ser independientes entre si, y técnicamente no hay alguno más importante que otro. Esto trae bastantes beneficios al momento de desarrollar, dado que el modo de trabajo suele ser que cada desarrollador tenga su repositorio propio sobre el cual trabaje de forma independiente, y periódicamente se pongan en común los trabajos de todos en algún repositorio convenido a tal efecto.

Los cambios que forman parte un repositorio no necesariamente deben estar dispuestos de forma cronológica, si bien muchos sistemas trabajan de esta forma, existen algunos que lo hacen de manera distinta, basándose en la relación entre los cambios para este fin.

4 Metodología y Fases de la Elaboración del Proyecto

La metodología XP (extreme programing)[5] es una de las llamadas metodologías ágiles, puesto que se basan en el concepto de que cualquier cambio debe ser fácilmente adaptable al proyecto, para así garantizar su éxito. En estas metodologías es muy importante la interacción entre las partes y la colaboración que el cliente ofrece en lugar de la negociación, a su vez indica que es más importante el código que funciona y no la documentación de las aplicaciones. Un plan cerrado no debe ser seguido si en el proyecto se presentan cambios, estos deben ser tomados en cuenta sobre la planeación.

El movimiento XP es relativamente nuevo (a partir de febrero de 2001). En una reunión en la cual estaban los representantes de diferentes metodologías ágiles se expusieron importantes ideas sobre lo que es ideal para un proyecto de desarrollo de software, dándole paso a esta metodología.

Los principios sobre los cuales esta fundamentada no son nuevos, simplemente son el realce de aquellos elementos que se tienen en cuenta en el desarrollo de los proyectos pero que tienen una mayor importancia respecto a otros. Podría decirse que no es una nueva metodología sino una forma diferente de ver el desarrollo del software. Recalca la importancia de la simplicidad, el sentido común, y de los resultados, sobre los procesos ineficientes que demoran tiempo y no tienen gran importancia.

El cliente siempre se encuentra cercano al desarrollo del proyecto, tanto que casi hace parte del equipo desarrollador, pues esta siempre disponible a resolver las dudas. El es el encargado de realimentar el proceso después de cada iteración y de llevar a cabo las pruebas de aceptación de las que se hablaran más adelante en este documento.

Al inicio del proceso el cliente define en sus palabras las características que desea del proyecto, y a medida que el desarrollador define las posibilidades tanto técnicas como económicas, se ponen de acuerdo para aclarar las funciones a realizar y la prioridad de estas. A medida que las iteraciones se van realizando el cliente revisa los resultados y se crean nuevas sugerencias para el desarrollo del proyecto por parte de ambos lados.

Este proceso es muy diferente en otras metodologías, especialmente en las tradicionales, aunque se guarda cierta relación con las listas de requerimientos que son generalmente utilizadas.

[5] Tomado de http://www.xprogramming.com

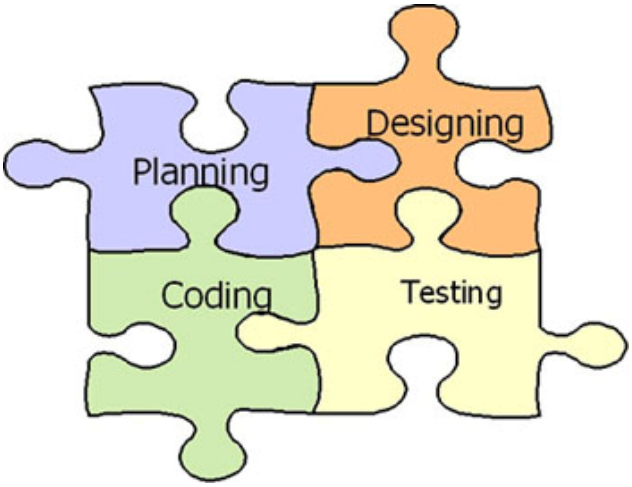


Figura 1. Fases de la programación extrema

En este caso el diseño es un proceso continuo, ya que los cambios a los que se está acostumbrado en la metodología presentan como opción redefinir partes del proyecto para agregar funcionalidad al sistema. Debido a que no se conocen con anticipación toda la información sobre los elementos e inconvenientes que se presenten durante el proyecto, los autores de la programación extrema rechazan la fórmula de las metodologías tradicionales para hacer un diseño completo antes de empezar a codificar.

El desarrollo en esta metodología debe ser tan rápido como sea posible, pero bajo los principios de “Hacerlo funcionar, hacerlo bien y entonces hacerlo rápido”.

Uno de los términos más innovadores de la programación extrema es el de metáfora. Es una herramienta para mejorar la comunicación entre los desarrolladores para crear una visión global y común del sistema que se va a desarrollar. Se expresa en términos sencillos y conocidos para todos, y la forma más sencilla es comparando lo que se va a desarrollar con algo que se encuentra en la vida real. Los conceptos en los que incurre la metáfora son claridad, reusabilidad y simplicidad.

La programación extrema persigue que exista una integración continua a través del desarrollo del proyecto, evitando al máximo la integración al final. Entre las ventajas de este proceso se encuentra la posibilidad de hacer siempre las pruebas a la totalidad del sistema, y no solo a módulos o partes del mismo.

Por lo tanto para todo desarrollo realizado con esta metodología se deber tener siempre una versión integrada del proyecto en un repositorio central accesible para todo el equipo. Se recomienda que la sincronización con las copias locales de los desarrolladores se haga al menos una vez al día. De esta manera

siempre se trabajará sobre la última versión del proyecto y no sobre una obsoleta.

Como se busca que cualquier persona tenga derecho a modificar lo que otro hizo se concluye que el código desarrollado no tiene “dueño”, y cualquier fracción del sistema puede ser modificada sin problemas. De esta manera también se garantiza que el retiro de un miembro del equipo del proyecto no sea catastrófico, sino que existe la posibilidad de que otros miembros que han trabajado con su código conozca tanto como el su trabajo.

Otro de los elementos muy importantes de esta metodología es la refactorización. Este proceso consiste en modificar constantemente el código a fin de que este sea lo más sencillo y legible como sea posible. Es importante que el código reemplazado tenga la misma funcionalidad, aunque también se pretende no aumentar los tiempos de ejecución de los programas.

Este proceso ayuda a que otros desarrolladores puedan entender de una manera simple todos los elementos del código y hacer menos personalizada la aplicación, así no se depende de la presencia de una persona específica para que el proyecto siga el curso establecido. Además, cuando se vea la necesidad de migrar módulos a otros lenguajes o plataformas, no será un proceso tedioso por falta de información sobre las clases o métodos utilizados.

El código debe ser comentado de la misma manera: sencilla. Pero además tratando de explicar lo máximo posible, se recomienda el uso de estándares al momento de escoger los nombres de variables, funciones y demás elementos.

Una de las prácticas más novedosas y a su vez criticadas de esta metodología es la de programación en parejas. Esta dice que un desarrollador que tenga la posibilidad de trabajar con otro, lo haga. Esto incluye la misma pantalla, teclado y ratón, y además que es preferible si cada cierto tiempo este compañero rota en el grupo. Esta innovadora idea dice que una persona que trabaja en compañía es más creativa y feliz en su trabajo, por lo tanto es más productiva puesto que el proceso es continuo y se visualizan los errores con más rapidez y facilidad. Además el hecho de hacer rotación influye en el aprendizaje de nuevos métodos y técnicas por parte de los compañeros.

Cuando se habla de metodologías ágiles se tienen muy en cuenta las pruebas del sistema. Las primeras pruebas que se ponen a consideración son las de unidad, después generalmente se hacen las funcionales, de carga del sistema y finalmente las de aceptación por parte del cliente. A continuación se describen un poco cada uno de estos tipos:

Pruebas de unidad o funcionamiento: Estas pruebas se realizan con el fin de asegurar que los módulos del sistema tiene una funcionalidad adecuada. Este tipo de pruebas se escriben antes de la realización de cada uno de los módulos,

y se desarrolla el código pensando en que supere cada una de las pruebas asignadas. Así, los programadores saben hacia donde deben ir enfocados y no tienen que pasar tiempo extra corrigiendo los problemas que aparecen después cuando se hacen pruebas posteriores.

Pruebas de funcionalidad: son las llamadas pruebas de Caja Negra en las que se examinan las salidas que presenta el sistema dependiendo de las entradas, es muy importante documentar este tipo de pruebas para no reincidir en los errores que se han presentado con anterioridad.

Pruebas de carga: En estas se presentan en el sistema situaciones extremas o no previstas y se examina el comportamiento y la respuesta del mismo. Esto ayuda a detectar los puntos débiles y los que son necesarios optimizar. Por ejemplo se examinan las entradas a la base de datos de forma concurrente.

Pruebas de aceptación: Estas pruebas son generalmente realizadas por el cliente final o el jefe del proyecto en su defecto. Es conveniente tener listo un documento que explique de la manera más sencilla la funcionalidad del sistema a fin de que el cliente examine si es correcto el funcionamiento e inclusive los detalles por pulir antes de la entrega final. Con este paso se da fin al proceso de desarrollo, una vez el cliente esta satisfecho ya no queda más sino poner a funcionar el sistema y si es necesario capacitar clientes secundarios en su uso.

4.1 Valores de la programación extrema

Los valores representan aquellos aspectos que los autores de XP han considerado como fundamentales para garantizar el éxito de un proyecto de desarrollo de software. Estos son:

- Comunicación
- Simplicidad
- Realimentación
- Coraje

Los partidarios de la programación extrema afirman que estos son necesarios para hacer diseños y códigos simples y de alta calidad, y métodos eficientes de desarrollo. Todo por la necesidad de obtener clientes contentos.

Como ya se ha resaltado la comunicación es un pilar fundamental en esta metodología. Y no solo se refiere a la comunicación dentro del grupo sino también con el cliente final.

La simplicidad es parte de cualquier metodología ágil de desarrollo, y busca entre otras ventajas la de poder entender con facilidad todo lo que se hace de forma rápida.

La realimentación además de paciencia y humildad a la hora de compartir el código con otros desarrolladores requiere tiempo, que al final se ve recompensado gratamente con la alta calidad y la satisfacción de los jefes y del cliente.

De estos valores se resalta el coraje y de este el lema “si funciona, mejóralo”, que difiere del pensamiento tradicional de dejar lo que funciona quieto para que siga en ese mismo estado.

4.2 Principios y prácticas de la programación extrema

Los principios fundamentales se apoyen en los valores y también son cuatro. A través de esta metodología se busca:

- Realimentación Veloz
- Modificaciones Incrementales
- Trabajo de calidad
- Asunción de simplicidad

Estos principios a su vez suponen un puente con las practicas y estas están ligadas a las técnicas que se van a seguir. A continuación se retoman las prácticas presentadas en el capitulo anterior:

- El juego de la planificación (The planning game)
- Entregas pequeñas (Small releases)
- Metáfora (Metaphor)
- Diseño simple (Simple design)
- Pruebas (Testing)
- Refactorizacion (Refactoring)
- Programación en parejas (Pair programming)
- Propiedad colectiva del código (Collective ownership)
- Integración continua (Continuous integration)
- Cliente in-situ o cliente en casa (on-site costumer)
- Estándares de programación (Coding standards)
- 40 horas semanales (40-hour week)

En palabras de Kent Beck, “la programación extrema es un forma ligera, eficiente, flexible, predecible, científica y divertida de generar software”. Aunque no es fácil aplicar una nueva metodología, puesto que esto implica aprender una nueva forma de trabajar, en los últimos años se ha empezado a cambiar la forma de pensar de muchos desarrolladores sobre la forma como una metodología debe influenciar los grupos de trabajo.

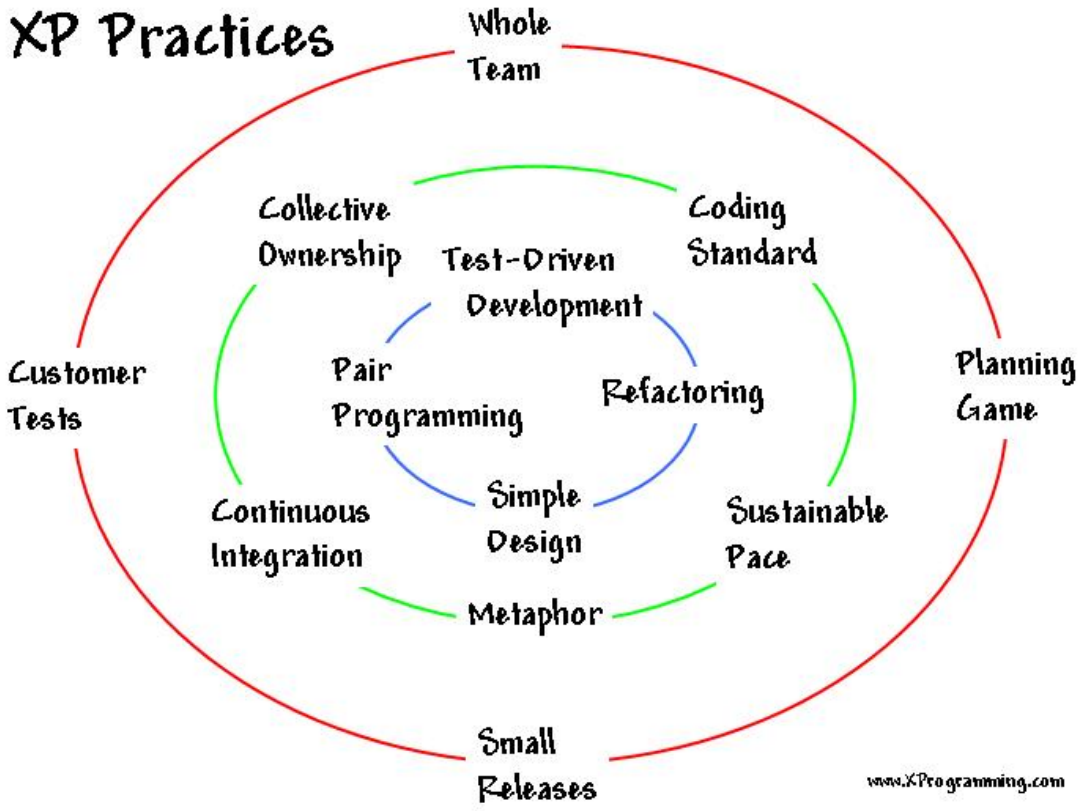


Figura 2. Practicas de la programación extrema, imagen extraída de la web de Ron Jeffries.

De los trabajos desarrollados en estos años con esta metodología la conclusión más frecuente que se encuentra es que la implementación de esta no debe ser inmediata sino gradual, para no generar un rechazo automático debido a la naturaleza de no aceptar el cambio con facilidad.

4.3 Fases en la elaboración del proyecto

A continuación se describen las actividades que se realizan en cada una de las iteraciones del proyecto a medida que estas son necesarias para continuar con el desarrollo del mismo.

Fase I. Lectura y Análisis de Información

Esta fase es realizada durante la mayor parte del tiempo durante el cual el proyecto se desarrolla. El objetivo de esta fase en la etapa inicial es principalmente la contextualización del desarrollador en el proyecto específico. Después de esto se realiza la capacitación sobre cada una de las herramientas y técnicas disponibles que serán utilizadas.

Las actividades realizadas en esta fase son:

- Conocimiento del problema al cual se intenta dar solución con el desarrollo del proyecto.

- Análisis de las posibles soluciones al problema planteado.

- Capacitación y entrenamiento sobre las herramientas de programación y bases de datos a utilizar.

- Instalación de las herramientas de programación, bases de datos, seguridad y servidor web.

- Capacitación y entrenamiento sobre el sistema de control de versiones que se requiere implementar para dar solución al problema.

- Instalación del sistema de control de versiones y clientes gráficos según sistemas operativos.

- Elaboración del documento de especificaciones del sistema.

Fase II – Diseño

En esta fase se definen el modelo de datos, la estructura de los archivos y se diseñan las interfases de interacción con los usuarios.

Las actividades a desarrollar en esta fase son:

- Diseño de la base de datos

- Diseño de la interfaz gráfica del portal

- Diseño de las interfases con la base de datos

- Diseño de las pruebas sobre la aplicación

- Diseño de las pruebas integrales

- Elaboración del documento de especificaciones de diseño

Fase III – Implementación

Durante el transcurso de esta fase se lleva a cabo la codificación del módulo Web del sistema de documentación.

Las actividades a realizar en esta fase son:

- Carga y descarga de la base de datos con registros de prueba.

- Codificación de todos los módulos del portal.

- Desarrollo de las interfases con la base de datos

Fase IV – Pruebas

En esta fase se llevan a cabo las pruebas que fueron diseñadas anteriormente y otras pruebas tanto individuales como integradas para comprobar el correcto funcionamiento de las aplicaciones.

Las actividades a desarrollar durante esta fase son:

- Pruebas de desempeño

- Pruebas de carga de los datos al sistema

- Pruebas de almacenamiento

- Pruebas de las páginas en forma individual

- Pruebas de la aplicación en forma integral

Durante el transcurso del proyecto se realizaran los manuales y demás documentación respecto a la aplicación.

5 Sistemas De Control De Versiones

Cuando se realiza un proyecto, especialmente uno grupal, pueden aparecer distintos problemas o inconvenientes sobre el desarrollo del mismo. Uno de ellos es la aparición de conflictos entre distintas versiones de un documento, especialmente cuando hablamos de código fuente.

En proyectos de duración prolongada también se ve el caso en el que se observa la aparición de un problema en la aplicación debido a la codificación o a un método específico utilizado. En este caso es necesario hacer una revisión para saber qué está causando el problema, si este se ocasionó en un tiempo pasado muy corto, desde el cual no se hicieron muchos cambios, lo más recomendable es recurrir a un estado anterior. Pero si por el contrario, no se sabe donde ocurrió el error se debe hacer una comparación entre los estados del código para arreglar el problema.

Para estos casos, y algunos otros inconvenientes que se presentan en el desarrollo de un proyecto, especialmente software, se utilizan herramientas de control de versiones. Estas herramientas no son para reducir el tiempo de desarrollo, en realidad en algunas hacen lo contrario. Lo que se busca son aplicaciones con excelente calidad y con un historial completo que ayude en próximos desarrollos relacionados.

Como se mencionó en un capítulo anterior, los sistemas de control de versiones son en general los encargados de manejar información en un almacén de datos, guardando información sobre los cambios que han sido realizados a través del tiempo sobre esta.

En la actualidad hay una gran cantidad de herramientas que realizan entre otras funciones la de control de versiones, tanto en el sector propietario, como en el de software libre. Aunque tienen características similares cada una de ellas tiene diferentes componentes y utilidades, así como su forma de utilización es variada.

El procedimiento habitual de un sistema de control de versiones se puede asimilar con el siguiente:

- El cliente descarga la copia de trabajo inicial (Checkout)
- Ciclo de trabajo habitual:
    - Modificación de los ficheros
    - Actualización de los ficheros en la copia local (Update)
    - Resolución de conflictos (En el caso que existan)

- Actualización de ficheros en el repositorio central (Commit)

Al realizar la investigación sobre las posibilidades existentes y las ventajas que estos paquetes pueden ofrecer a la EISI, se destacó la utilización de CVS (concurrent version system). Muchos proyectos de gran importancia han sido, y son en la actualidad, controlados a través de este sistema. Fue uno de los pioneros en su tipo y quizás el más popular también. Muchas otras herramientas han surgido gracias a la importancia que este proyecto tiene.

Aunque actualmente CVS es una solución viable porque su licencia es GPL, lo que significa que no tiene costo, y por su gran popularidad existe mucha documentación sobre su instalación y manejo; este sistema de control presenta algunos problemas que no fueron planteados en su diseño inicial.

Tratando de corregir estos problemas han surgido con los años diversas soluciones que ha su vez han incorporado nuevas utilidades y facilitado la interacción con los usuarios.

A continuación se presenta la descripción de las soluciones más populares en el mercado, junto con sus características más importantes.

5.1 CVS

Concurrent version system, mejor conocido como CVS es un sistema de control de versiones que mantiene el registro de todo el trabajo y los cambios realizados durante la implementación de un proyecto y permite que los desarrolladores colaboren entre sí, sin importar la distancia física entre ellos. CVS es muy popular en el mundo del software libre especialmente porque fue uno de los primeros y más completos sistemas de su tipo.

CVS al igual que otros sistemas de control utiliza la arquitectura cliente servidor; esto quiere decir que las versiones de un proyecto y su historia se guardan en el servidor, y los clientes se conectan a el para descargar la versión actual y hacer los cambios que consideren necesarios; una vez el proceso termina vuelven a conectarse para actualizar su trabajo. El sistema generalmente se conecta a través de Internet, pero también es posible utilizarlo localmente como cliente y servidor en un solo equipo. El servidor normalmente utiliza un sistema operativo de la familia UNIX, mientras que los clientes pueden estar en cualquier sistema operativo bien conocido.

Varios clientes pueden sacar copias del proyecto al mismo tiempo. Cuando se ingresan las modificaciones el servidor trata de acoplarlas de la forma más conveniente, pero si existen inconsistencias es tarea del equipo desarrollador resolverlas y enviar la versión correcta. A cada envío CVS incrementa automáticamente el número de revisión de todos los archivos que son

guardados en el repositorio. Con cada revisión también se tiene una historia sobre lo que se hizo y quíen fue el responsable, por ello se puede volver a un estado anterior he inclusive hasta el inicio del proyecto. Sin embargo, a pesar de sus interesantes características, este sistema tiene varias debilidades importantes, por lo cual han surgido otros sistemas que las corrigen.

Características Positivas:

- Estable.
- Bien documentado.
- Muy conocido (es fácil encontrar ayuda).
- Soporta múltiples sistemas operativos.
- Fuente abierta Licencia GNU GPL.
- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Es posible trabajar en un directorio específico del repositorio.
- Tiene excelente documentación técnica sobre su uso.
- Se administra a través de un juego de comandos muy simple.
- Buen soporte a redes.
- Permite replicación remota del repositorio usando CVSup de John Polstra.
- El cliente funciona en UNIX, Windows y Mac OS. El servidor en sistemas UNIX y Windows con emuladores.
- Existen interfaces graficas como CVSweb, ViewVC, Chora y wwCVS.
- Tiene disponibles clientes gráficos como WinCVS, Cervisia, y TortoiseCVS.

Características Negativas:

- Las operaciones con el repositorio no son atómicas.
- No soporta archivos binarios.
- No soporta renombrar y mover archivos.
- No soporta copiar directorios y archivos.
- No gestiona permisos en el repositorio.

5.2 Aegis

Aegis es un sistema de control de Peter Miller. Trata de garantizar la integridad del código por varios medios como registrando pruebas automáticas y revisiones de código. Es distribuido y tiene otras características interesantes pero es un sistema de archivos primario y puede ser usado solo por redes que utilizan NFS o un protocolo similar.

Características Positivas:

- Licencia GNU GPL.
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.
- Gestiona permisos en el repositorio.
- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Existen interfaces Web compatibles.
- El cliente gráfico es tkaegis.

Características Negativas:

- No soporta copiar directorios y archivos.
- No es posible trabajar en un directorio específico del repositorio.
- Tiene documentación técnica sobre su uso de calidad regular.
- Se administra a través de un juego de comandos bastante complejo.
- Pobre soporte a redes.
- La fuente funciona en todos los sistemas UNIX, pero no en Windows.

5.3 Arch

Arch fue creado por Tom Lord para satisfacer la necesidad de control de revisiones suyas y de otras personas. Maneja renombrar, ramas, grupos de cambios y otras acciones que no puede realizar CVS. Es muy sencillo de configurar y poner en servicio. Utiliza el modelo distribuido.

Características Positivas:

- Licencia GNU GPL.
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.
- Gestiona permisos en el repositorio.
- Tiene documentación técnica sobre su uso.
- Excelente soporte a redes.
- La fuente funciona en los sistemas UNIX y en Windows con emuladores.
- Como interfaz grafica tiene ViewARCH y ArchZoom.
- Tiene disponibles clientes gráficos como tlator, Octopy y ArchWay que está en desarrollo.

Características Negativas:

- No soporta copiar directorios y archivos.
- No muestra la historia por línea.
- Es posible trabajar en un directorio específico del repositorio, pero el checkout se debe hacer del repositorio entero.
- Se administra a través de un juego de comandos complejo y mixto.

5.4 BitKeeper

BitKeeper es un sistema de control de versiones comercial creado por BitMover Inc. Es costoso pero muy bueno, soporta casi todas las características que se pueden tener en cuenta para un sistema de su tipo. Es de tipo distribuido y funciona en la mayoría de sistemas operativos.

Características Positivas:

- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.
- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Tiene muy buena documentación técnica sobre su uso.
- Se administra a través de un juego de comandos sencillo.
- Buen soporte a redes.
- Funciona en los sistemas UNIX y Windows 98.

Características Negativas:

- Software propietario. Pago por Licencia de uso.
- No es posible trabajar en un directorio específico del repositorio.
- Una interfaz Web para este sistema esta en proceso de producción.

5.5 ClearCase

Este sistema se convierte en parte del sistema operativo para obtener control completos sobre todos los accesos a los archivos que el administra. Puede además adaptarse al modelo centralizado o distribuido. Esto es necesario tanto para seguridad como para ofrecer un servicio transparente.

Características Positivas:

- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos, aunque usarlo es difícil.
- Propagación de cambios entre repositorios usando Multisite.
- Gestiona permisos en el repositorio.
- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Es posible trabajar en un directorio específico del repositorio.
- Tiene abundante documentación técnica sobre su uso.
- Se administra a través de un juego de comandos excelente.
- Disponible en Windows y algunos UNIX, incluyendo MacOSX y Linux como Red Hat.
- Tiene una interfaz Web compatible.

Características Negativas:

- Licencia propietaria, pago de sostenimiento anual.
- No permite replicación remota del repositorio.
- Pobre soporte a redes.

5.6 CMSynergy

Es un sistema de control basado en tareas, ayuda al equipo desarrollador a trabajar más rápido y fácil incrementando la comunicación y la colaboración entre sí. Ofrece un repositorio distribuido para la creación de software de alta calidad.

Características Positivas:

- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos.
- Permite replicación remota del repositorio si se tiene la distribución oficial.
- Propagación de cambios entre repositorios.
- Es posible trabajar en un directorio específico del repositorio.
- Se administra a través de un juego de comandos poderoso.
- Buen soporte a redes.
- Funciona en Windows familia NT, VMS, y algunos sistemas UNIX.

Características Negativas:

- Licencia propietaria, el precio varia según el vendedor.

- No gestiona permisos en el repositorio.

5.7 Co-Op

Co-Op es un sistema de control de versiones de colaboración distribuida para desarrolladores que necesitan manejar y proteger su código fuente. No es necesario un servidor central para su uso. Su interfaz grafica es muy intuitiva y tiene una funcionalidad flexible, aunque tan bien se puede obtener una interfaz de línea de comandos.

Caracteriasticas Positivas:

- Transacciones atómicas.
- Tiene muy buena documentación técnica sobre su uso.
- Se administra a través de un juego de comandos básico.
- Soporte a redes a través de LAN.
- Funciona siempre de manera local, no requiere interfaz Web.

Características Negativas:

- Licencia propietaria, $160 aproximadamente por computador.
- Para renombrar archivos es necesario crear un nuevo directorio y copiar los archivos. La historia no se pierde.
- Copiar archivos o directorios borra la historia, mientras moverlos no lo hace.
- El primer acceso al repositorio requiere aprobación del administrador. Las siguientes entradas no son controladas.
- No es posible trabajar en un directorio específico del repositorio.
- No posee clientes gráficos.
- No posee interfaz Web.
- Solo funciona en Windows a partir de Windows 95.

5.8 Darcs

Es un sistema de control de versiones distribuido fácil de configurar, basado en la teoría de los parches y con características importantes.

Características Positiva:

- Licencia GNU GPL.
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.

- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Tiene buena documentación técnica sobre su uso.
- Buen soporte a redes.
- Es soportado por muchos sistemas UNIX, Mac OSX, y Windows.

Características Negativas:

- No soporta copiar directorios y archivos.
- No gestiona permisos en el repositorio.
- Es posible trabajar en un directorio específico del repositorio, pero se debe obtener todo el repositorio.
- Se administra a través de un juego de comandos mixto.

5.9 Mercurial

Mercurial es un programa de control de versiones que inició como respuesta a la falta de disponibilidad de BitKeeper con licencia de código abierto. Fue diseñado intencionalmente para ser pequeño, fácil de usar y altamente escalable. Las principales fortalezas de este sistema son su simplicidad y rapidez. Aunque son conceptualmente diferentes a CVS y SubVersion puesto que es distribuido, su interfaz de línea de comandos es similar aunque el inicio toma unos minutos.

Características Positivas:

- Licencia GNU GPL.
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.
- Gestiona permisos en el repositorio.
- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Es posible trabajar en un directorio específico del repositorio, se obtiene una copia parcial.
- Tiene muy buena documentación técnica sobre su uso.
- Se administra a través de un juego de comandos parecido al de CVS.
- Excelente soporte a redes.
- Corre en todas las plataformas suportadas por Python.
- Tiene una interfaz Web incluida como un componente.

Características Negativas:

- El inicio de la interfaz de comandos toma algunos minutos.

5.10 Monotone

Monotone es un sistema de control de versiones distribuido con una filosofía diferente. Generalmente los grupos de cambios se fijan al repositorio, que se obtienen de varias fuentes. Luego cada desarrollador descarga los que quiera tener en su repositorio privado. Sin embargo, Monotone es lento, las bases de datos y los historiales no son escalables.

Características Positivas:

- Licencia GNU GPL.
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.
- Gestiona permisos en el repositorio.
- Muestra línea a línea la historia de la revisión y el autor en un documento, a partir de la versión 0.19.
- Tiene buena documentación técnica sobre su uso.
- Se administra a través de un juego de comandos parecido al de CVS.
- Buen soporte a redes.
- El ejecutable funciona en todos los sistemas UNIX y Win32.
- Sistema de almacenamiento sencillo (SQL)
- Basado en transacciones.
- Eficiente protocolo de sincronización (netsync).
- Versiones basadas en firmar criptográficas.

Características Negativas:

- La internacionalización no es completa (Los nombres deben ser ASCII).
- Se encuentra en pleno desarrollo.
- No puede eliminar directorios del repositorio (solo archivos).
- No tiene cliente gráfico.
- No tiene interfaz Web.
- Es posible trabajar en un directorio específico del repositorio, pero se debe obtener todo el árbol exacto.

5.11 OpenCM

OpenCM es diseñado como un reemplazo para CVS, seguro, de alta integración y centralizado. Soporta varias cosas de las que carece CVS. Provee soporte de

primera clase para renombrado y configuración, autenticación y control de acceso por criptografía, y control de ramas.

Características Positivas:

- Licencia GNU GPL.
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Tiene buena documentación técnica sobre su uso.
- Se administra a través de un juego de comandos similar al de CVS.
- Buen soporte a redes.

Características Negativas:

- No soporta copiar directorios y archivos.
- No permite replicación remota del repositorio.
- No es posible trabajar en un directorio específico del repositorio.
- Funciona en todos los sistemas UNIX.
- No posee interfaz web.

5.12 Perforce

Esta es una solución para el control de versiones centralizada y comercial. Es muy rápida, estable y robusta, tiene buena escalabilidad y buena reputación. Requiere una licencia anual por usuario, pero también está disponible para desarrolladores de software libre de manera gratuita. Es prácticamente equivalente a Subversion, sin embargo este último es de código abierto.

Características Positivas:

- Transacciones atómicas.
- Soporta copiar directorios y archivos.
- Permite replicación remota del repositorio vía Perforce Proxy tool.
- Gestiona permisos en el repositorio.
- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Es posible trabajar en un directorio específico del repositorio.
- Tiene muy buena documentación técnica sobre su uso.
- Se administra a través de un juego de comandos muy completo pero no compatible con el de CVS.
- Buen soporte a redes.
- Corre en sistemas UNIX, Mac OS, BeOS y Windows.
- La interfaz Web es P4Web.

Características Negativas:

- Licencia propietaria comercial.
- No soporta renombrar y mover archivos directamente, se manejan ramas del archivo.
- No posee clientes gráficos.
- No posee interfaz Web.

5.13 PureCM

PureCM es un sistema de control similar a Microsoft Visual SoruceSafe y CCS, que provee varios servios a desarrolladores de software y proveedores de contenido. De modelo centralizado. El papel principal de este sistema es el de capturar configuraciones y permitir cambios auditando las mismas.

Características Positivas:

- Licencia propietaria.
- Transacciones atómicas.
- Soporta renombrar y mover archivos, pero es necesario crear nuevos archivos y carpetas y borrar las antiguas
- Soporta copiar directorios y archivos.
- Gestiona permisos en el repositorio.
- Es posible trabajar en un directorio específico del repositorio.
- Tiene muy buena documentación técnica sobre su uso.
- Se administra a través de un juego de comandos sencillo.
- Buen soporte a redes.
- El servidor funciona en forma grafica en Windows y en linea de comandos en sistemas UNIX y Mac OS.

Características Negativas:

- No permite replicación remota del repositorio.
- No tiene interfaz Web.

5.14 Subversion[6]

Este es un proyecto creado por Tigris.org, para ser el mejor reemplazo posible de CVS, resolviendo los principales inconvenientes de este sistema y ofreciendo otras características. Subversion tiene un diseño modular y usa tecnologías conocidas como Berkeley DB 4.0, Apache 2.0, mod_dav, la librería Neon y Apache Portable Run-Time. Su código tiene alta calidad y tiene convenciones

[6] Tomado de http://subversion.tigris.org

muy estrictas. Es un producto muy profesional y de modelo centralizado. Finalmente es totalmente de código abierto y distribuido bajo la licencia Apache. Dentro de sus figuras extra están el soportar el renombrado y las copias, grupos de cambios implícitos, habilidad de comunicarse sobre HTTP y HTTPS, versionado de directorios, y un eficiente manejo de archivos binarios.

Características Positivas:

- Licencia tipo Apache/BSD (open-source)
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos de una manera muy sencilla.
- Permite replicación remota del repositorio a través de utilidades extras.
- Propagación de cambios entre repositorios.
- Gestiona permisos en el repositorio.
- Muestra línea a línea la historia de la revisión y el autor en un documento, a través de *svn blame*.
- Es posible trabajar en un directorio específico del repositorio.
- Tiene muy buena documentación técnica sobre su uso.
- Muy buen soporte a redes.
- Clientes y servidores funcionan en sistemas UNIX, Windows y Mac OS X.
- Tiene multiples interfaces Web, por ejemplo: ViewVC, SVN::Web, WebSVN, ViewSVN, mod_svn_view, chora, trac, SVN::RaWeb::Light, SVN Browser, Insurrection y Perl svn. Ademas Apache provee la interfaz por defecto.
- Tiene disponibles clientes gráficos como; RapidSVN, TortoiseSVN, Jsvn, etc.

Características Negativas:

- Hay que memorizar comandos.

5.15 Superversion

Este es un sistema de control de versiones multiusuario y distribuido. Es de carácter industrial y una alternativa de código abierto similar a las soluciones comerciales e inclusive más fácil de usar. En el equipo de desarrollo de este sistema las prioridades son eficiencia y uso intuitivo.

Características positivas:

- Licencia GNU GPL.
- Transacciones atómicas.
- Permite replicación remota del repositorio.
- Tiene pobre documentación técnica sobre su uso.

- No es necesario memorizar comandos.
- Buen soporte a redes.
- Cliente y servidor funcionan en cualquier plataforma compatible con Java 1.4.

Características Negativas:

- No soporta renombrar y mover archivos.
- No soporta copiar directorios y archivos.
- No gestiona permisos en el repositorio.
- No es posible trabajar en un directorio específico del repositorio.
- No tiene interfaz Web.

5.16 Svk

Es un sistema de control de versiones distribuido que fue construido con el sistema de archivos de Subversion. Soporta operaciones sin conexión, mezclado sensitivo y se integra con otros sistemas de control de versiones, como también con herramientas de mezclado visuales.

Características Positiva:

- Licencia Perl (open source)
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos igual que subversion.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.
- Gestiona permisos en el repositorio.
- Muestra línea a línea la historia de la revisión y el autor en un documento.
- Es posible trabajar en un directorio específico del repositorio.
- Se administra a través de un juego de comandos sencillo.
- Muy buen soporte a redes.
- Las interfaces Web de Subversion son compatibles para svk.

Características Negativas:

- Tiene pobre documentación técnica sobre su uso.
- El cliente requiere subversion y perl.

5.17 Vesta

Vesta es un software maduro para el control de versiones centralizado que fue originalmente creado para el uso interno de Digital Equipment Corporation

(ahora Compaq/HP). Se uso como reemplazo de CVS y provee más herramientas que otros controladores de versiones simples. Es usado para construirse el mismo, pero también se utilizo otro paquete que se encuentra disponible.

Características Positivas:

- Licencia GNU GPL.
- Transacciones atómicas.
- Soporta renombrar y mover archivos.
- Soporta copiar directorios y archivos.
- Permite replicación remota del repositorio.
- Propagación de cambios entre repositorios.
- Gestiona permisos en el repositorio.
- El soporte a redes es inherente al sistema.
- Debe funcionar en cualquier sistema UNIX. En Solaris y FreeBSD aun no funciona.
- La interfaz web creada para este sistema es Vestaweb.

Características Negativas:

- Tiene pobre documentación técnica sobre su uso.
- Se administra a través de un juego de comandos diferente al estándar, pero los usuarios utilizan generalmente solo 5 de ellos.

5.18 Microsoft Visual SourceSafe (VSS)

Visual SourceSafe es una solución fácil de usar para desarrolladores que quieren un camino simple para manejar los cambios de su código fuente. Es un sistema centralizado.

Características Positivas:

- Soporta copiar directorios y archivos.
- Gestiona permisos en el repositorio.
- Es posible trabajar en un directorio específico del repositorio.

Características Negativas:

- Licencia propietaria, el producto es una agregación de MSDN.
- Transacciones no atómicas.
- No soporta renombrar y mover archivos directamente.
- No Permite replicación remota del repositorio.
- Es un producto Microsoft y por lo tanto funciona solo para Windows.

A continuación se muestra un cuadro comparativo con las principales características, pero se debe tener en cuenta que la mayoría de estas aplicaciones se encuentran en desarrollo y los datos aquí presentados pueden cambiar.

Tabla 2. Comparación entre las herramientas de control de versiones analizadas.

Característica	CVS	AEGIS	ARCH	BITKEEPER	CLEARCASE	CMSYNERGY	CO-OP	DARCS	MERCURIAL	MONOTONE	OPEN CM	PERFORCE	PURE CM	SUBVERSION	SUPERVERSION	SVK	VESTA	VSS
Licencia de Software libre	SI	SI	SI	NO	NO	NO	NO	SI	SI	SI	SI	NO	NO	SI	SI	SI	SI	NO
Repositorio Centralizado	SI	NO	NO	NO	SI NO	NO	NO	NO	NO	NO	SI	SI	SI	SI	NO	NO	SI	SI
Desarrollo Continuado	NO	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	NO
Buen Soporte a Redes	SI	NO	SI	SI	NO	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI		
Transacciones Atómicas	NO	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	NO
Copiado de Archivos	NO	NO	NO	SI	SI	SI		NO	SI	SI	NO	SI	SI	SI	NO	SI	SI	SI
Mover y Renombrar	NO	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI	NO	SI	SI	NO	SI	SI	
Replicación Remota	NO	SI	SI	SI	NO	SI		SI	SI	SI	NO	SI	NO	SI	SI	SI	SI	NO
Propagación de Cambios	NO	SI	SI	SI	SI	SI		SI	SI	SI	NO	NO	NO	SI	NO	SI	SI	NO
Gestión de Permisos	NO	SI	SI	SI	SI	NO	SI NO	NO	SI	SI	SI NO	SI	SI	SI	NO	SI	SI	SI
Trabajo en Directorio Especifico	SI	NO	SI		SI	SI	NO		SI		NO	SI	SI	SI	NO	SI	NO	
Historia de Cada Revisión	SI	SI	NO	SI	SI	NO		SI	SI	SI	NO	SI	NO	SI	NO	SI	NO	NO
Cliente Gráfico Disponible	SI	SI	SI	SI	SI		NO	NO		NO	NO	SI	NO	SI	SI	NO		SI
Interfaz Web Disponible	SI	SI	SI	NO	SI	SI	NO	SI	SI	NO	NO	SI	NO	SI	NO	SI	SI	SI
Documentación Adecuada	SI	NO	SI	SI	SI	SI	SI	SI	SI		NO	SI	SI	SI	NO	NO	NO	SI
Sencillo Juego de Comandos	SI	NO	NO	SI		SI	SI	SI	SI		SI	NO	SI	SI	SI	SI		

6 Herramienta seleccionada

Para escoger la herramienta indicada para el manejo de control de versiones en la EISI se tuvieron en cuenta, además de otros, los siguientes aspectos:

- Tipo de licencia: Como existe una buena cantidad de opciones tanto de la parte comercial como de fuente abierta, se decidió que la licencia de la herramienta debe ser gratuita, para que la EISI no tenga que incurrir en costos innecesarios teniendo la posibilidad de evitarlos.

- Tipo de arquitectura: El principal inconveniente de un sistema centralizado es la necesidad de tener un servidor disponible. Sin embargo, la EISI tiene los recursos necesarios y el servidor Cormorán se encuentra activo. Por lo tanto no se justifica que la herramienta tenga una arquitectura distribuida donde el control no es tan estricto y depende mucho del grupo de trabajo.

- Autenticación: Es necesario mantener la seguridad de los proyectos que se encuentran en desarrollo, así como proteger los códigos fuente y el acceso remoto al servidor, el sistema de control de versiones debe ser seguro en este sentido.

- Soporte a redes: Ya que requiere un servicio que se pueda obtener de forma remota, es necesario que el sistema tenga uno o varios tipos de soluciones para soportar las comunicaciones entre los desarrolladores y el servidor Cormorán.

- Documentación existente: Es importante que la documentación acerca de la herramienta sea fácil de entender y obtener, no solo para hacer más sencilla su instalación y configuración sino también para solucionar posibles problemas que se puedan presentar en el futuro.

- Modelo de entrega de cambios: Se busca un sistema con el modelo copy-modify-merge en lugar de uno con modelo lock-modify-unlock. Esto, porque el primero permite que un archivo pueda ser modificado por varios usuarios al mismo tiempo y no tengan inconvenientes si no

trabajan sobre las mismas líneas, y dado este caso se pregunta sobre la opción correcta, y no se limita a que un archivo solo pueda ser modificado por un solo usuario en una fracción de tiempo indefinido.

- Envíos atómicos: Para la seguridad de las transacciones es necesario que el sistema obligue a realizar una operación completa o no realizarla en lo absoluto con el fin de evitar que por un fallo en la red u otra parte del protocolo los archivos se modifiquen a medias o queden errores en las actualizaciones.

- Copia y Renombrado de archivos y directorios: Para no perder el historial de un archivo que necesita cambiar de ubicación o de nombre el sistema debe permitir estas operaciones.

- Compatibilidad con otros programas: Se necesita que la instalación del controlador de versiones seleccionado no cause problemas de configuración con otros programas previamente instalados y en lo posible de futura utilización.

- Interfaz: Se busca una herramienta con la posibilidad de tener una interfaz Web, además de una interfaz grafica amigable con los usuarios, para el cliente final.

- Portabilidad: Es necesario que la herramienta sea compatible diferentes versiones de Linux y además con Windows, sistemas operativos utilizados más comúnmente utilizados en nuestro medio.

- Tiempo mínimo de carga: Los sistemas de control de versiones al inicio de cada sesión verifican cuáles archivos han sido modificados en una copia de trabajo. Se busca que el sistema se demore lo menos posible realizando esa operación con el fin de optimizar la utilización del mismo.

- Estabilididad: En la actualidad muchas de estas aplicaciones se encuentran en pleno desarrollo y no cuentan con una versión estable, sino que hacen actualizaciones con frecuencia, una herramienta de este tipo haría más difícil la administración de la aplicación puesto que sería necesario cambiar la versión con frecuencia para tomar el mayor provecho. Por otro lado una herramienta con una o varias versiones estables no necesitará tantas actualizaciones como las demás.

Luego de estudiar estas características y analizar también comentarios de usuarios de diferentes herramientas, se llegó a la conclusión que la más apropiada para la EISI es "SubVersion", de Tigris.org. Esta herramienta es la que mejor cumple con los requisitos anteriormente descritos. Aunque también se estudiaron más profundamente opciones como Mercurial que se ajusta de manera similar pero tiene la desventaja del manejo del tiempo y trabaja con el modelo distribuido, o Vesta que parece ser al igual una opción viable, pero no se encontró información de la calidad que se requiere y no tiene la opción de trabajar individualmente sobre un directorio especifico. Más información sobre estos sistemas es posible encontrarla en las direcciones http://www.selenic.com/mercurial/wiki/ y http://www.vestasys.org/ respectivamente.

El siguiente capítulo está enfocado en describir las características de Subversion y los programas con los que el trabajo será realizado.

7 Control de versiones con la herramienta SubVersion

7.1 Introducción a la herramienta

¿Que es SubVesion?

Es un sistema de control de versiones de código fuente abierto y libre desarrollado por Tigris.org. En palabras sencillas, maneja ficheros y directorios a través del tiempo. Para esto utiliza un repositorio central, en el que se mantiene un árbol de ficheros. Este repositorio es como un servidor de ficheros ordinario, pero recuerda todos los cambios hechos a sus ficheros y directorios.

Figura 3. Logo de SubVersion.

La herramienta puede acceder al repositorio a través de redes, lo que permite ser usado por personas en diferentes ubicaciones geográficas. La capacidad para que varias personas puedan modificar y administrar el mismo conjunto de datos desde sus respectivos computadores fomenta la colaboración. Bajo el control de versiones, no hay razones para temer por la calidad del trabajo, si se ha hecho un cambio incorrecto a los datos, simplemente se deshace ese cambio.

7.1.1 Conceptos Básicos

- Repositorio

  Subversion es un sistema centralizado para compartir información. La parte principal de Subversion es el repositorio, el cual es un almacén central de datos. El repositorio guarda información en forma de árbol de archivos, una típica jerarquía de archivos y directorios. Cualquier número de clientes puede conectarse al repositorio y luego leer o escribir en esos archivos. Al escribir datos, un cliente pone a disposición de otros la información; al leer datos, el cliente recibe información de otros.

  Lo que hace especial al repositorio de Subversion es que recuerda todos los cambios hechos sobre él, cada cambio a cada archivo, e inclusive cambios al propio árbol de directorios, tales como la adición borrado y reubicación de archivos y directorios.

Cuando un cliente lee datos del repositorio, normalmente sólo ve la ultima versión del árbol de archivos, pero el cliente también tiene la posibilidad de ver estados previos del sistema. Esto es, un cliente puede hacer consultas históricas, puesto que estos sistemas están diseñados para registrar y seguir los cambios en los datos a través del tiempo.

- Modelo de versionamiento Bloqueo-Modificación-Bloqueo

La misión principal de un sistema de control de versiones es permitir la edición colaborativa y la compartición de datos. El problema es como permitir a los usuarios compartir información pero a la vez evitar que otros puedan sobrescribir o dañar el trabajo realizado cuando están trabajando sobre la misma porción de un proyecto. Muchos sistemas de control de versiones utilizan el modelo de bloqueo-modificación-desbloqueo para evitar este problema. Solo se permite que una persona modifique el archivo en un lapso de tiempo. El primer usuario bloquea el archivo antes de hacerle cambios, mientras tanto otro usuario debe esperar hasta que el primero desbloquee este archivo. Este método es efectivo, sin embargo, no es eficiente. Pueden ocurrir problemas si un usuario olvida que tiene bloqueado un archivo y otros lo necesitan, puesto que no podrán trabajar en el. Tampoco es bueno en el caso de que un usuario deba modificar la primera parte de un documento y otro la parte final, en este caso no debería existir el bloqueo porque se pierde tiempo.

- Modelo de versionamiento Copiar-Modificar-Mezclar

Subversion, CVS y otros sistemas de control de versiones utilizan este modelo como alternativa al bloqueo. Aquí, el cliente de cada usuario se conecta al repositorio, hace una copia del proyecto, trabaja sobre esta copia, después retorna esta copia con los cambios realizados, y el repositorio se encarga de mezclar todas las copias privadas en una nueva versión final. En el caso óptimo el sistema de control de versiones hace totalmente la mezcla, pero en caso de inconsistencias (conflicto) corresponde a un ser humano la responsabilidad de que esto suceda correctamente. Estos conflictos generalmente suceden por falta de comunicación entre los miembros de un grupo de trabajo.

7.1.2 Características

- Versionado de directorios

Subversion implementa un sistema de ficheros versionado “virtual” que sigue los cambios sobre los árboles de directorios completos a través del tiempo.

- Verdadero historial de versiones

Con subversion se puede añadir, borrar, copiar y renombrar ficheros y directorios, a diferencia de CVS, que no soporta operaciones como estas. Además se puede reemplazar un fichero versionado con uno nuevo que lleve el mismo nombre y el nuevo elemento no heredará el historial del fichero antiguo.

- Envíos atómicos

Una colección cualquiera de modificaciones o entra por completo al repositorio, o no lo hace en absoluto, permitiendo a los desarrolladores enviar los cambios como fragmentos lógicos e impide que ocurran problemas cuando sólo una parte de los cambios enviados entran con éxito.

- Versionado de metadatos

Cada fichero y directorio tiene un conjunto de propiedades asociado a él. Se pueden crear y almacenar cualquier par arbitrario de clave/valor que se desee. Las propiedades son versionadas a través del tiempo al igual que el contenido de los ficheros.

- Elección de las capas de red

La elección de las capas, se refiere a la habilidad que SubVersion tiene para adoptar diferentes formas de acceso al repositorio, facilitando a las personas implementar nuevos mecanismos de red. Subversion puede conectarse al servidor http Apache como un módulo de extensión. Esto proporciona le proporciona estabilidad e interoperabilidad, y acceso instantáneo a las características existentes que ofrece este servidor, autenticación, autorización, compresión de la conexión, etcétera. También tiene disponible un servidor de Subversion independiente, y más ligero. Este servidor habla un protocolo propio, el cual puede ser encaminado fácilmente a través de un túnel SSH.

- Manipulación consistente de datos

Subversion expresa las diferencias del fichero usando un algoritmo de diferenciación binario, que funciona idénticamente con ficheros de texto (legibles para humanos) y ficheros binarios (ilegibles para humanos). Ambos tipos de ficheros son almacenados igualmente comprimidos en el repositorio, y las diferencias son transmitidas en ambas direcciones a través de la red.

- Ramificación y etiquetado eficientes

Es posible crear ramas y etiquetas simplemente copiando el proyecto, usando un mecanismo similar al enlace duro. De este modo estas operaciones toman solamente una cantidad de tiempo pequeña y constante.

- Facilidad de desarrollo

La herramienta *SubVersion* no tiene un equipaje histórico; está implementado como una colección de bibliotecas compartidas en C con APIs bien definidas. Esto la hace extremadamente fácil de mantener y reutilizable por otras aplicaciones y lenguajes.

7.1.3 Ciclo básico de trabajo

Inicialmente el repositorio vacío debe obtener la información con la cual todos los usuarios trabajarán; esta operación se llama **import** y normalmente es realizada una sola vez por una persona del grupo, aunque no existen restricciones y se pueden hacer todas las importaciones necesarias.

Cada uno de los usuarios debe obtener en su computador local una copia de esta información, este proceso se conoce como **checkout** y se realiza una vez para cada copia que el usuario quiera obtener.

Finalmente el ciclo regular de trabajo consiste en actualizar (**update**) los cambios de otros, modificar, y confirmar los cambios de cada uno (**commit**).

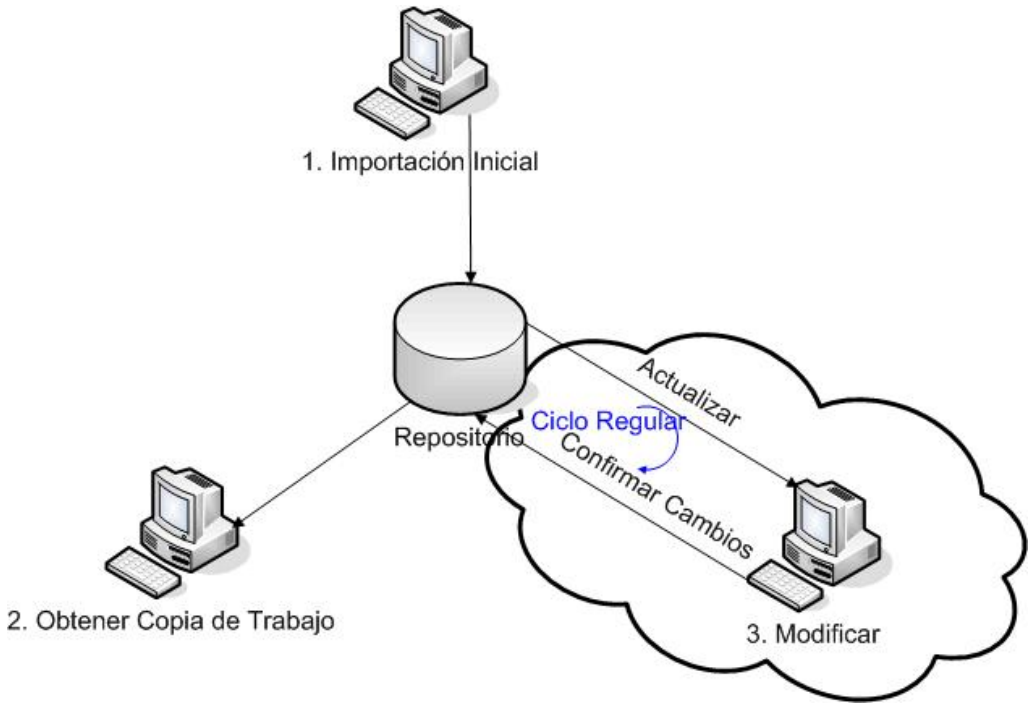


Figura 4. Ciclo básico de trabajo

7.1.4 Operaciones básicas

Import

Cuando se empieza a hacer control de versiones es generalmente porque ya se tiene algo que controlar, por ejemplo un archivo o un conjunto de ellos. El comando “import” es la forma de introducir estos documentos al repositorio para empezar a trabajar con ellos o para compartirlos con los compañeros de un grupo de trabajo.

El comando tiene el siguiente formato cuando se utiliza de forma local:

svn import [opciones] /proyecto/local file:///parentpath/nombre_del_repositorio

Ejemplo:

svn import –m “Mensaje de inicio” /root/mis_proyectos file:///svn/repositorio

Las opciones que puede llevar este comando son varias pero las más importantes son:
--message (-m) “Texto”
--username Usuario
--password Contraseña

Cuando se utiliza de forma remota debe ser configurado como se ve más adelante, y tiene la siguiente forma:

svn import [opciones] /proyecto/local
http://server.domain/parentpath/nombre_repositorio

Ejemplo:

svn import –m “Mensaje de inicio” /root/mis_proyectos
http://200.21.228.144/svn/repositorio

En los dos casos el ejemplo sube los datos iniciales al servidor y éstos se ubican en el repositorio de *SubVersion*.
Después de agregados todos los subdirectorios y documentos que incluya la información original, muestra la siguiente línea.

Commit de la revisión 1.

Esto quiere decir que se tiene la información dentro del repositorio.

Checkout

Cuando ya se tiene un repositorio no es posible trabajar directamente sobre los datos contenidos en él. Para esto son utilizadas copias de trabajo. Una copia de trabajo es la misma información que se tiene en el repositorio, pero que puede ser fácilmente manipulada. El comando Checkout se utiliza para obtener estas copias de trabajo, esto se hace normalmente solo la primera vez que se desea obtener este información, sin embargo se puede utilizar cada vez que se necesite una nueva copia de trabajo.

La estructura es la siguiente:

svn checkout [opciones] file:///parentpath/nombre_repositorio
/camino/a/copia/de/trabajo

Ejemplo:

En este ejemplo se obtendrá la copia de trabajo del contenido de “repositorio” en una carpeta del escritorio.

svn checkout file:///svn/repositorio /root/Desktop/mi_repositorio

Una de las opciones más importantes del comando checkout es:
--revision (-r) Rev
Puesto que nos permite cargar una copia de trabajo de una revisión anterior. Para saber que numero de revisión que necesitamos se utilizan los mensajes de registro de cada commit.

Cuando se utiliza de forma remota:

svn checkout [opciones] http://server.domain.parentpath/nombre_repositorio
/camino/a/copia/de/trabajo

Ejemplo:

svn checkout http://192.168.65.15/svn/repositorio /root/Desktop/mi_repositorio

En este punto es posible utilizar los archivos que contiene “mi_repositorio” para trabajar de forma local sin la preocupación de estar o no en comunicación con el repositorio del servidor.

Update

Este comando permite actualizar la copia de trabajo y así saber si los cambios que hicieron otros usuarios interfieren en los cambios realizados el usuario de la copia de trabajo y así evitar problemas al copiar nuevamente los archivos al repositorio.

Estando ubicado en el subdirectorio que contiene la copia de trabajo ejecutar:

```
cd mi_repositorio
svn update
```

Luego de hacer esta operación ya se han sincronizado los cambios del repositorio a la copia de trabajo local.

Diff

Este comando es muy similar a Update, sin embargo compara la copia de trabajo local y la información contenida en el repositorio y muestra las diferencias.

En el subdirectorio que contiene la copia de trabajo ejecutar:

```
cd mi_repositorio
Svn diff
```

Commit

Este comando es quizá uno de los más importantes. En el momento de guardar en el repositorio el trabajo realizado sobre la copia de trabajo local se ejecuta este comando.

svn commit [opciones]

Ejemplo:

svn commit –m “Mensaje de cambios” mi_repositorio

Ahora los cambios realizados en la copia local están en el repositorio a disposición de los demás usuarios autorizados.

Log

Este comando muestra los registros almacenados cuando hay un commit, es decir, cuando se envían cambios al repositorio.

svn log -v -r 0:N --limit 100 [--stop-on-copy] /Camino/al/archivo

Delete

Este comando sirve para borrar los archivos del repositorio de forma correcta.

svn delete /Camino/al/archivo

Otras operaciones

add:	Añade los ficheros en la ruta /Camino/al/archivo
revert:	Revierte las modificaciones locales de una copia de trabajo. El camino dice que ítems deben revertirse.
cleanup:	Hace limpieza tras operaciones interrumpidas o abortadas.
resolve:	Marca un fichero en conflicto especificado como resuelto.
repocreate:	Crea un repositorio en la ruta /Camino/al/directorio.
switch:	Cambia la ruta especificada para el directorio de destino.
export:	Hace una copia de trabajo sin los ficheros .svn, que son creados en cada copia de trabajo.
merge:	Hace la fusión de la ruta especificada.
copy:	Hace una copia a una rama o una etiqueta.
move:	Mueve los archivos de una ubicación a otra dentro del repositorio.
remove:	Elimina los ficheros especificados del control de versiones
rename:	Renombra el fichero
ignore:	Añade a la lista de ignorados: svn:ignored
lock:	Bloquea el fichero.

- **Herramienta svnlook**

Es una herramienta para examinar las revisiones. Tiene parámetros --revision y --transaction.

autor:	autor del árbol.
cat:	vuelca los contenidos de un fichero en el árbol.
changet:	Da lista de ficheros y directorios modificados.
date:	fecha del árbol.
diff:	Vuelca de ficheros modificados.
dirs_changet:	lista de directorios modificados.
history:	historia de una ruta versionada.
info:	autor, fecha, nº de caracteres, mensaje de registro.
log:	mensaje de registro.
propget:	valor de una propiedad de una ruta en el árbol.
proplist:	nombres y valores de las propiedades de rutas.
tree:	listado del árbol.
uuid:	identificador único de usuario de árbol.
youngest:	último número de revisión.

- **Herramienta svnadmin**

Permite crear los repositorios y realizar operaciones de mantenimiento. Usa los comandos Create, dump y help.

create:	crear un nuevo repositorio.
dump:	vuelca los contenidos del repositorio agrupados por un conjunto dado de revisiones.
hotcopy:	copia en caliente del repositorio (backup).
list-dblogs:	lista las rutas a los repositorio.
list-unused-dblogs:	lista las rutas que han dejado de usar.
load:	carga un grupo de revisiones en un repositorio.
lstxns:	lista nombres de transacciones no confirmadas.
recover:	realiza tareas de recuperación.
rmtxns:	borra limpiamente del repositorio transacciones.
setlog:	sustituye el log actual.
verify:	verifica los contenidos del repositorio

- svndumpfilter:	include exclude Para seleccionar solo una parte de los contenidos del repositorios
- svn list –verbose	Para verificar las importaciones

En la dirección http://svnbook.red-bean.com/index.es.html es posible encontrar varias versiones del manual de referencia en español, en el se encuentra una explicación más detallada de cada uno de estos comandos.

7.2 TortoiseSVN

El mayor enfoque del control de versiones ha sido siempre orientado a los programadores, quienes necesitan hacer pequeños cambios al software y eventualmente deshacer algunos de estos, por lo tanto los desarrolladores orientan su proceso creativo a la programación. Se espera tener estos programadores trabajando concurrentemente y simultáneamente en los mismos ficheros. Para esto se necesita un buen sistema.

Figura 5. Logo de TortoiseSVN.

¿Qué es TortoiseSVN?

TortoiseSVN[7] es un cliente gratuito de código abierto para el sistema de control de versiones *Subversion*, creado por los mismos autores. Está desarrollado para integrarse con el shell de Windows, por lo tanto solo puede ser utilizado en los sistemas operativos Windows a partir de XP. TortoiseSVN maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un repositorio central, prácticamente lo mismo que un servidor de ficheros ordinario, salvo que recuerda todos los cambios hechos en el tiempo. Esto permite recuperar versiones anteriores y examinar la historia de cuando y como cambiaron los datos.

Características de TorotoiseSVN

- Integración con el shell de Windows

TorotoiseSVN se integra perfectamente con el núcleo de Windows, ésto quiere decir que el cliente puede seguir trabajando con las herramientas con las que ya está familiarizado y no tiene que cambiarse a una aplicación diferente cada vez que necesite las funciones del control de versiones. Este cliente utiliza menús contextuales que funcionan en varios administradores de archivos incluyendo el Explorador de Windows.

- Sobreimpresión de iconos

El estado de cada carpeta y fichero versionado se indica por pequeños iconos sobre-impresionados. De esta forma, se puede ver fácilmente el estado en el que se encuentra su copia de trabajo.

- Fácil acceso a los comandos de *Subversion*.

Todos los comandos de SVN están disponibles desde el menú contextual del explorador. TortoiseSVN añade su propio submenú allí.

Dado que TortoiseSVN es un cliente de *Subversion*, las características propias de éste también son heredadas.

La recomendación general cuando se maneja un proyecto grande es que se creen 3 carpetas para el manejo de los datos, estas son:

/trunk
/branches
/tags

[7] Tomado de http://tortoisesvn.tigris.org

La carpeta trunk (tronco) se utiliza como línea principal de los datos y es sobre la cual se basan todos los proyectos, la carpeta branches (ramas), para las copias del proyecto con pequeños cambios, en las cuales se pueden llevar de manera paralela los cambios de algunos archivos y de otros no; y la carpeta tags (etiquetas), para contener las copias/etiquetas del mismo. Esta forma de organización es casi un estándar de los proyectos bajo el control de *SubVersion*.

Cuando hay proyectos relacionados sobre el mismo repositorio se maneja esta estructura, o cada una de las carpetas para cada uno de los proyectos.

Si los proyectos no están relacionados o son diferentes es mejor si son separados en varios repositorios, así, también es más fácil la autenticación de usuarios.

Además, el número de revisión cambia cada vez que se envían los cambios al servidor, no importa si hay varios proyectos sobre los que se realizan cambios en el mismo repositorio, por esta razón es que los usuarios de un proyecto determinado puedan pensar que se están "perdiendo" algunos números de las revisiones. Por supuesto cada desarrollador es libre de organizar su información de la forma que más le convenga sin tener en cuenta la descrita anteriormente.

En la dirección http://tortoisesvn.tigris.org/ es posible encontrar más información sobre el proyecto.

7.3 RapidSVN

RapidSVN [8] es una interfaz grafica multiplataforma del sistema de control de versiones *SubVersion, creado por Tigris.org*. Funciona sobre Windows y la mayoría de distribuciones de Linux, aunque se planea que pronto este disponible para Mac OS y otros sistemas operativos. Es distribuido con una licencia GPL de GNU. Fue escrito en el lenguaje C++, utilizando wxWidgets como plataforma grafica.

Figura 6. Logo de RapidSVN.

Sus principales atributos son los siguientes:

[8] Tomado de http://rapidsvn.tigris.org

- Simple: Provee una interfaz fácil de usar para las características de *SubVersion*.

- Eficiente: Simple para principiantes y lo suficientemente flexible para incrementar la productividad de los usuarios experimentados de *Subversion*.

- Portable: Corre en cualquier plataforma en la que *Subversion* y wxWidgets puedan funcionar.

- Rápido: Enteramente escrito en C++.

Para el correcto funcionamiento de esta herramienta es necesario tener instalado en la maquina *Subversion* y wxWidgets, se recomiendan las ultimas versiones.

Este cliente gráfico a través de las opciones que posee reproduce la mayor parte de las características más utilizadas de Subversion de una manera fácil y eficiente.

En la dirección http://rapidsvn.tigris.org/ es posible obtener información más actualizada sobre el proyecto.

8 Desarrollo del Portal

8.1 Análisis del sistema

8.1.1 Requisitos de información

El portal Web desarrollado debe contener lo siguiente:

- Manejo de la información de usuarios

  El sistema debe almacenar la información dada por los usuarios que son registrados:
  Identificador de usuario (código)
  Identificador del rol
  Estado del usuario
  Nombres
  Apellidos
  Nombre de usuario
  Contraseña
  Correo electrónico
  Genero
  Fecha de creación de usuario
  Fecha de actualización de datos

- Información de roles de usuario

  El sistema debe almacenar la información sobre los roles de usuario:
  Identificador del rol
  Nombre del rol

- Información sobre la documentación

  El sistema debe almacenar la información relacionada a cada uno de los documentos contenidos en el repositorio respectivo:
  Identificador del documento
  Nombre del documento
  Identificador del Tipo de documento
  Autor (Si corresponde)
  Fecha de importación
  Palabras clave
  Otra información referente

- Tipo de documento

  Se debe almacenar la información sobre los tipos de documentos que serán registrados en el sistema:
  Identificador del tipo de documento
  Nombre del tipo de documento

- Información sobre los permisos de documentación

  Se debe almacenar la información referente a los permisos que cada rol tiene sobre los documentos
  Identificador del rol
  Identificador del documento
  Permiso

- Información sobre los proyectos

  Se debe almacenar la información referente a los proyectos que son desarrollados bajo el control de versiones de manera centralizada:
  Identificador del proyecto
  Nombre del proyecto
  Fecha de creación del proyecto
  Descripción del proyecto

- Información sobre los permisos de proyectos

  Se debe almacenar la información referente a los permisos que los usuarios registrados tienen sobre los proyectos en desarrollo:
  Código del usuario
  Identificador del proyecto
  Permiso de Lectura
  Permiso de Escritura
  Permiso de Ejecución

8.1.2 Requisitos funcionales del sistema

- Realizar búsqueda simple

  El sistema debe permitir al usuario realizar una búsqueda sobre parámetros simples.

- Realizar búsqueda avanzada

  El sistema debe permitir al usuario realizar una búsqueda sobre parámetros definidos tales como titulo, autor, palabras claves, etc.

- Mostrar resultados de las búsquedas

  El sistema debe mostrar los resultados de las búsquedas de acuerdo con los parámetros dados por el usuario, los permisos para su rol y la documentación existente.

- Listar proyectos bajo control de versiones

  Se debe mostrar una lista de los repositorios activos en el servidor para que los usuarios autorizados puedan accederlos.

- Mostrar las carpetas y archivos de los proyectos

  El sistema debe mostrar las carpetas y archivos de la última versión de un proyecto bajo el control de *SubVersion* a los usuarios autorizados.

- Realizar la copia de los archivos de los proyectos dentro del servidor

  El sistema debe permitir a un usuario dependiendo de su rol, hacer una copia de trabajo de los archivos contenidos en un proyecto a un directorio de prueba o de producción dentro del servidor que lo contiene.

- Listar usuarios

  El administrador debe poder listar los usuarios registrados en el sistema.

- Registrar nuevos usuarios

  El administrador puede crear nuevos usuarios definiendo el nombre de usuario, la contraseña, el rol y si lo desea los demás datos almacenados.

- Editar usuarios

  El administrador puede en cualquier momento modificar los datos básicos de cualquier usuario.

- Actualizar datos

  Un usuario puede modificar en su perfil todos los datos que a él mismo se refieren con excepción del identificador, el estado y el rol.

- Ingresar información de documentos

  El administrador puede ingresar la información completa que se refiere a un documento que ha sido almacenado en el repositorio de documentación correspondiente.

- Listar documentos

  El administrador debe poder listar los documentos registrados dentro del sistema.

- Editar información sobre documentos

El administrador del sistema puede en cualquier momento modificar los datos de los documentos que han sido previamente registrados.

- Editar permisos sobre los documentos
  El administrador del sistema puede modificar los permisos dados a determinado documento de acuerdo al rol del usuario.

- Listar los tipos de documentos
  El administrador puede listar los tipos de documentos por los cuales se están seleccionando los mismos.

- Editar los tipos de documentos
  El administrador puede crear o modificar los tipos de documentos almacenados en la base de datos.

- Listar los roles
  El administrador puede listar los perfiles existentes, guardados en la base de datos.

- Editar los roles
  El administrador puede crear o modificar los roles disponibles para los usuarios del sistema.

- Crear un nuevo repositorio
  El administrador puede crear nuevos repositorios y asignar permisos a los usuarios que pertenecen o tienen relación directa con el proyecto si estos ya fueron registrados.

- Crear copias de seguridad de los repositorios
  El administrador puede crear de forma sencilla una copia de seguridad de los repositorios existentes.

- Iniciar sesión
  Un usuario puede registrar su sesión en el momento que lo desee.

- Cerrar sesión
  Un usuario que ha registrado su sesión puede salir de ella cuando lo desee, si no lo hace su sesión termina cuando cierra la aplicación.

- Acceder a programas y manuales
  Debe existir un espacio en el portal desde el cual los usuarios puedan acceder al software libre que se utilizara como cliente gráfico del programa de control de versiones, así como el manual adecuado para su utilización.

- Acceder a la ayuda

  Cualquier usuario puede acceder a la ayuda de la aplicación y si lo desea contactar al administrador de la misma.

8.1.3 Requisitos no funcionales del sistema

- Ambiente de Desarrollo

  La aplicación debe ser desarrollada utilizando herramientas de desarrollo Web de libre distribución.

- Plataforma hardware

  Servidor:
  Procesador de 1.6 Ghz
  Memoria Ram de 512 Mb
  Disco Duro de 80 Gb
  Tarjeta de Red
  Unidad de Cd
  Monitor, teclado, Mouse, UPS

  Clientes:
  Procesador de 500 Mhz
  Memoria de 128 Mb
  Disco Duro de 10 Gb
  Tarjeta de Red
  Monitor, teclado, Mouse
  Unidad de entrada de datos (CD o floppy)

- Plataforma software

  Servidor:
  Sistema operativo Linux Red Hat Enterprise 3
  Postgresql 8.0
  Apache 2.5.0 o superior
  PHP
  Neon
  Subversion
  WxWigets 2.2
  Rapidsvn

  Clientes:
  - Sistemas operativos Linux
    RapidSVN

  - Sistema operativo Windows XP
    TortoiseSVN

- El sistema podrá ser accedido por varios usuarios al mismo tiempo ya que está diseñado para ser puesto en red.

- La utilidad de la aplicación se verá afectada por la capacidad de almacenamiento tanto de la base de datos como de los repositorios, dados por el servidor en el cual se alojará el proyecto.

- Los usuarios del sistema deben tener acceso periódico a Internet para que los cambios realizados se vean reflejados en todo el sistema.

8.2 Diseño

8.2.1 Diseño Global

En esta fase se analiza el contenido del portal y su interacción con la herramienta de control de versiones que se utilizará, se describen los elementos que harán parte de las páginas y la manera como los datos serán presentados a los usuarios.

Este diseño debe ser acorde con los requerimientos del sistema y la capacidad del hardware y software con el que se trabajara.

Diseño de la interfaz

En el diseño de la interfaz debe destacar la comodidad del usuario y la funcionalidad del sitio, pues estos son los factores que influyen en mayor parte para que la aplicación no sea abandonada por parte de las personas a las cuales ayuda en su labor.

Para el desarrollo de la interfaz se deben tener en cuenta algunos aspectos como:

- Distribución consistente: Las pantallas deben ser consistentes respecto al texto, los colores y las imágenes utilizadas. De esta manera no habrán grandes saltos de imágenes que puedan distraer a los usuarios.

- Previsión de acciones: Se deben tener en cuenta todas las posibles acciones de los usuarios para evitar errores y si estos ocurren mostrar los mensajes respectivos para que el usuario pueda corregir los datos.

- Contenido de las pantallas: Las páginas deben contener la información necesaria para que cualquier usuario entienda el funcionamiento. No deben estar saturadas de imágenes o información, si se necesita mostrar mucha información esta puede ser separada en varias páginas conectadas entre si.

- Navegación: El paso de una página a otra debe ser coherente con la información que requiere el usuario y los comandos realizados por parte del sistema.

La estructura del portal se dividió en tres partes para hacer más cómoda la navegación por parte de los usuarios y de acuerdo a estándares establecidos.

El diseño de la interfaz del portal será la siguiente:

- En la parte superior estarán ubicados los logotipos de la universidad y la escuela de ingeniería de sistemas, así como en título del portal. Los logotipos serán vínculos directos a las páginas correspondientes. Una barra ubicada debajo de ésta, contendrá opciones tales como volver al inicio, cerrar sesión, obtener ayuda, y cerrar la aplicación. En la parte izquierda de esta barra se encuentra la ubicación de la página visible dentro de la estructura del portal.

Figura 7. Tabla superior del prototipo Web.

- En una barra en la parte izquierda se encuentran los servicios a los cuales los usuarios pueden acceder de manera general, estos serán visibles en todo el recorrido por el portal. Los servicios ubicados en este nivel son: búsquedas, proyectos, software y administrador. En la parte inferior de este menú se encuentra el espacio correspondiente para iniciar sesión si el usuario no ha realizado esta operación.

Figura 8. Barra Lateral del prototipo Web.

- El área de trabajo corresponde al resto de espacio dentro de la página, en esta área el usuario recibirá la información solicitada y se llevaran a cabo las operaciones requeridas.

8.2.2 Modelo de datos

Los siguientes son los requisitos utilizados en el desarrollo:

De las tablas

- Los nombres deben ser escritos en minúscula, sin tildes, sin caracteres especiales y en plural.
- Si los nombres se componen de más de una palabra, estas estarán separadas por un guión de piso "_".

De los campos

- Los nombres de los campos deben ser escritos en minúscula y sin tildes ni caracteres especiales.
- Todo campo lleva el nombre de la tabla o en su defecto una abreviatura como prefijo, separado por un guión de piso "_".

De las llaves foráneas

- Las llaves foráneas recibirán el nombre que tienen en la tabla de origen y un sufijo que indique la tabla de destino.

De la programación

- La programación se llevara a cabo según los lineamientos del lenguaje PHP.
- Las constantes se escriben con mayúscula.
- Evitar comentarios obvios.

De las variables

- Los nombres de las variables serán escritos totalmente en minúscula, sin tildes ni caracteres especiales.
- Para las variables contador se utilizan normalmente la letras, “i”, “j”, ”k”, ”m”, y ”n”.

De la funciones y clases

- Las funciones y clases tienen la primer letra en mayúscula y a continuación minúsculas.
- El nombre de las funciones debe ser en lo posible un verbo.

8.2.3 Diseño detallado

En esta etapa se hace una descripción más precisa de las características que hacen parte de la estructura de la aplicación.

Diseño de la Base de Datos

Dada la información de los requisitos especificados y el análisis del diseño global de la aplicación se llega a la conclusión que la base de datos debe ser relacional, y su respectivo diagrama se muestra a continuación.

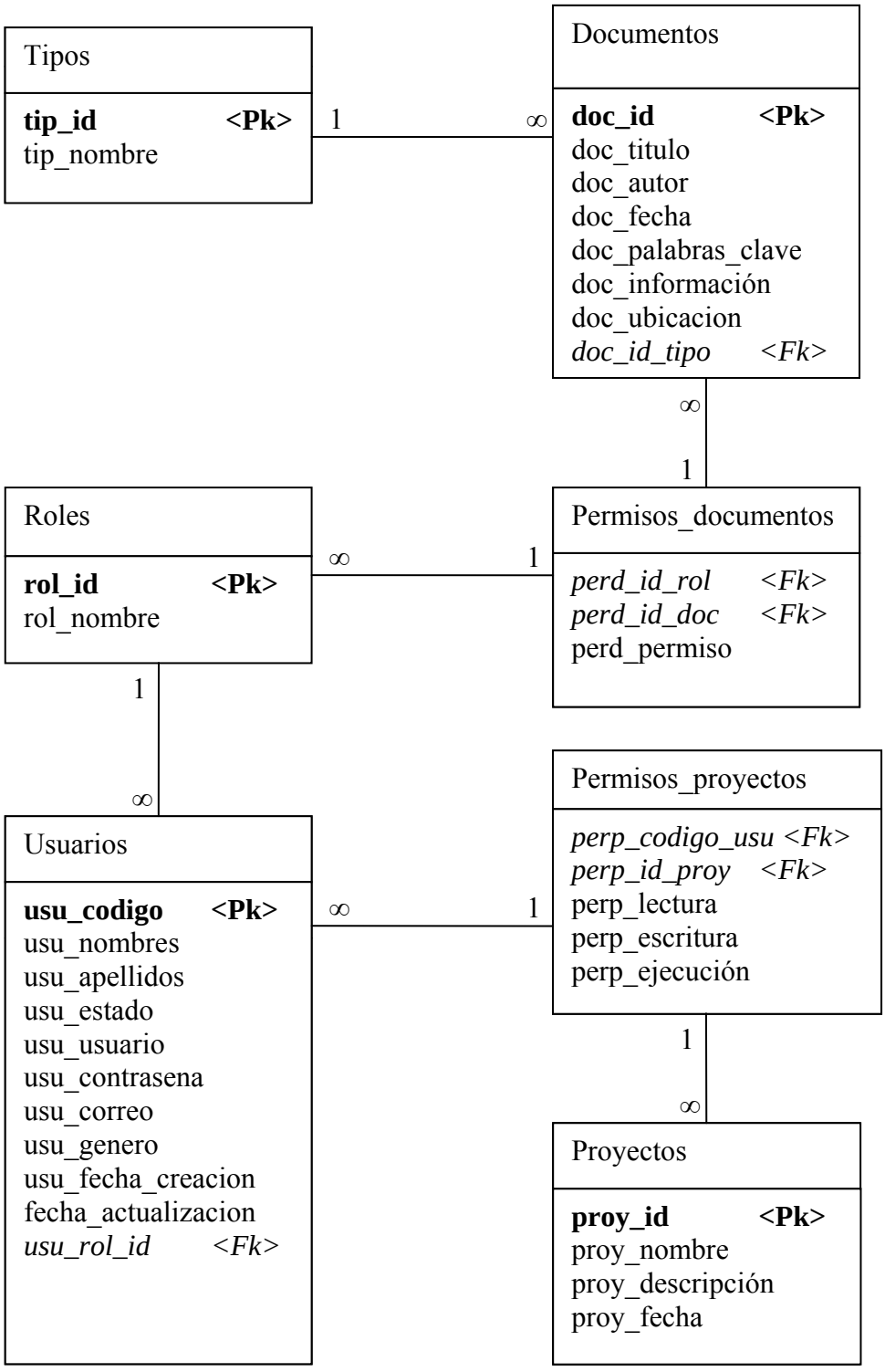


Figura 9. Diagrama Entidad – Relación.

La siguiente es la descripción de cada una de las tablas que hacen parte del diagrama entidad – relación mostrado.

- Tabla: usuarios
  Descripción: Contiene la información básica dada por los usuarios una vez se encuentran registrados en el sistema, esta información es valida tanto para la documentación como para el desarrollo de los proyectos.

- Tabla: roles
  Descripción: Esta tabla almacena los tipos de roles que un usuario puede tener en el caso de que se encuentre activo.

- Tabla: documentos
  Descripción: Esta tabla contiene información referente a los documentos almacenados en el repositorio de documentación de Cormorán los cuales pueden o no ser accedidos por diferentes tipos de usuarios.

- Tabla: permisos_documentos
  Descripción: Esta tabla relaciona de forma detallada los permisos que son otorgados a cada rol para el acceso a un documento determinado.

- Tabla: proyectos
  Descripción: Esta tabla almacena la información básica de los proyectos que son creados en los repositorios para que sean administrados mediante el uso de control de versiones.

- Tabla: permisos_proyectos
  Descripción: Esta tabla relaciona a cada proyecto con los usuarios que a el pueden acceder, así como los tipos de permisos que tienen cada uno de estos usuarios.

- Tabla: tipos
  Descripción: Esta tabla esta relacionada a cada documento, e indica el tipo al que el mismo pertenece.

Diseño de la presentación de los datos a usuarios

- Inicio y finalización de sesión

  En cualquier momento un usuario puede hacer el registro para iniciar su sesión, esto tendrá privilegios en los resultados de las búsquedas de acuerdo con el rol del usuario dentro del sistema. Así mismo cuando el usuario lo desee puede finalizar su sesión, lo cual hará que el portal retorne al inicio.

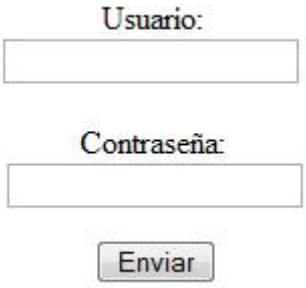


Figura 10. Inicio de sesión

- Cambio de contraseña

  Una vez el usuario se ha registrado para iniciar sesión tiene la opción de ingresar a la página de cambio de contraseña, en la cual este dato es remplazado por otro tanto en la base de datos como en el acceso a los repositorios:

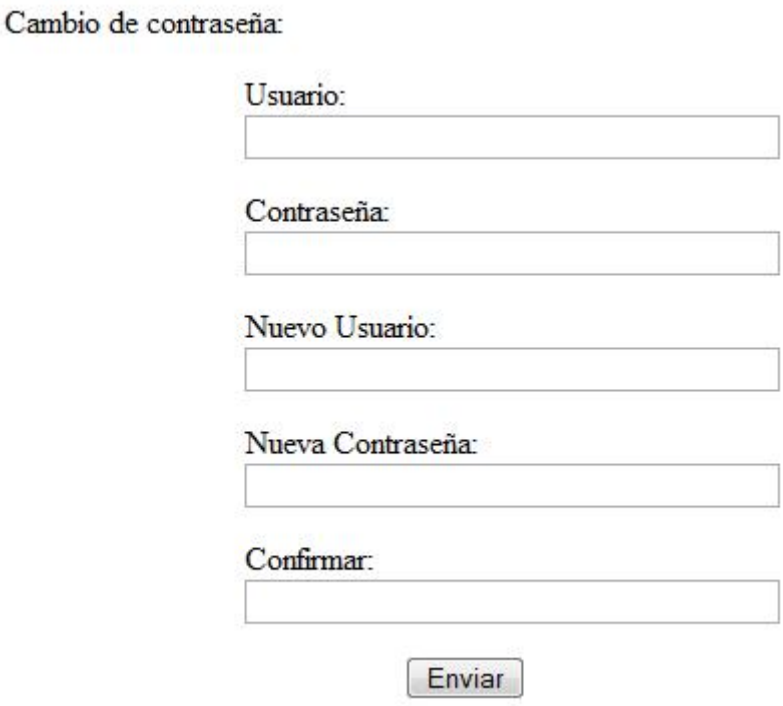


Figura 11. Cambio de contraseña

- Búsqueda sencilla de documentos:

  En esta página los usuarios buscan los documentos con un parámetro sencillo y los resultados son generales. La interfaz de petición de datos es la siguiente:

Figura 12. Formulario de búsqueda simple.

- Búsqueda avanzada

  Cuando el usuario conoce detalles sobre la información que esta buscando puede introducirlos en el siguiente formato para hacer más rápida y eficiente su petición:

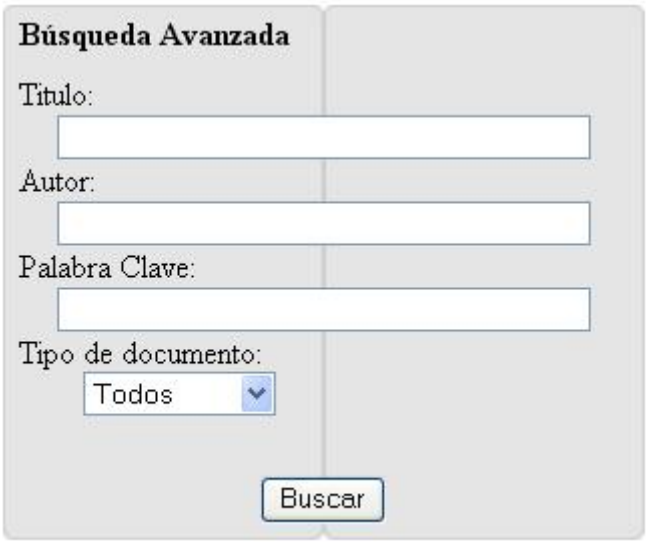


Figura 13. Criterios para búsqueda avanzada

- Resultado de las búsquedas

  Una vez el sistema ha comprobado los datos introducidos por el usuario y realizado la búsqueda en la base de datos, muestra los resultados obtenidos dando las características más significativas, las cuales son guardadas en la base de datos y un vínculo a la ubicación en el repositorio para que el usuario pueda acceder al documento.

- Listado de proyectos

  En el módulo de proyectos el usuario puede ver el listado de proyectos que se encuentran bajo el sistema de control de versiones en el servidor.

Este listado va acompañado del tipo de base de datos del repositorio. En el caso del servidor Cormorán el tipo es FSFS, puesto que el sistema operativo Red Hat Linux Enterprise 3 no soporta las bases de datos Berkley utilizadas también por el sistema de control de versiones.

- Autenticación de usuarios del repositorio

  Si el usuario intenta acceder al repositorio se realiza la autenticación por parte del sistema de control de versiones, si el usuario y contraseña corresponden el usuario podrá navegar por todas las carpetas y archivos que este contenga. De lo contrario este informará que la autenticación es necesaria. La interfaz de autenticación es la siguiente:

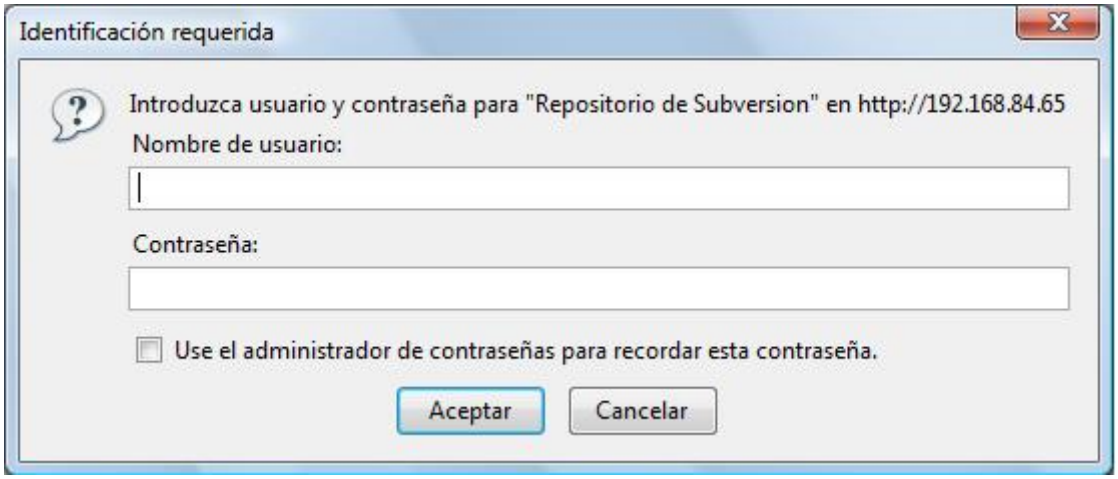


Figura 14. Ventana de autenticación de usuario de los repositorios

- Carga de sitios a prueba y producción

  Frente al listado de los proyectos guardados en el repositorio, se da la opción al usuario si el mismo tiene autorización, de cargar un portal a un sitio de prueba o definitivamente a producción.

Figura 15. Listado de Repositorios y permisos de actualización.

- Descarga de software

  El usuario que quiere que su proyecto sea administrado bajo el control de versiones implementado en la EISI puede acceder al software

utilizado como cliente gráfico de la herramienta, así como los manuales desarrollados para explicar su instalación y funcionamiento.

- Ingreso de un nuevo documento

El administrador del portal puede hacer el ingreso de un nuevo documento para que los usuarios puedan accederlo desde el módulo de búsquedas, dando los datos básicos relacionados, así como los permisos otorgados a cada uno de los roles que pueden tomar los usuarios del sistema. Para esta operación se utiliza el siguiente formato:

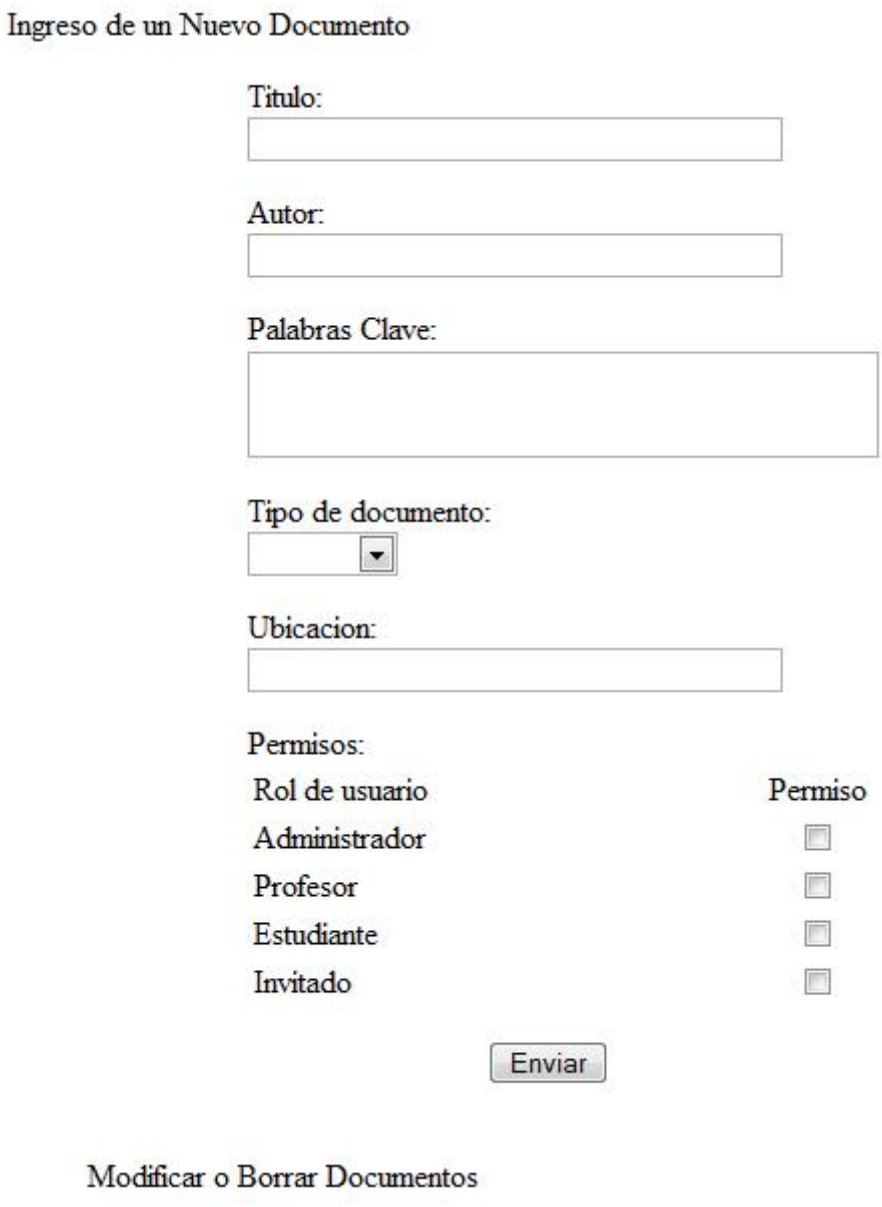


Figura 16. Datos para el ingreso de un nuevo documento

- Editar datos de los documentos

  El administrador puede listar y modificar los datos de un documento que ha sido registrado con anterioridad. Para este fin utiliza el mismo formato de la figura anterior.

- Ingreso de un nuevo usuario

  El administrador puede realizar el registro de un nuevo usuario, ya sea para las búsquedas de documentación o para los proyectos bajo el control de versiones. Todos los datos son los mismos al igual que el nombre de usuario y la contraseña utilizados. Para esto el administrador llena el siguiente formato con los datos suministrados por el usuario:

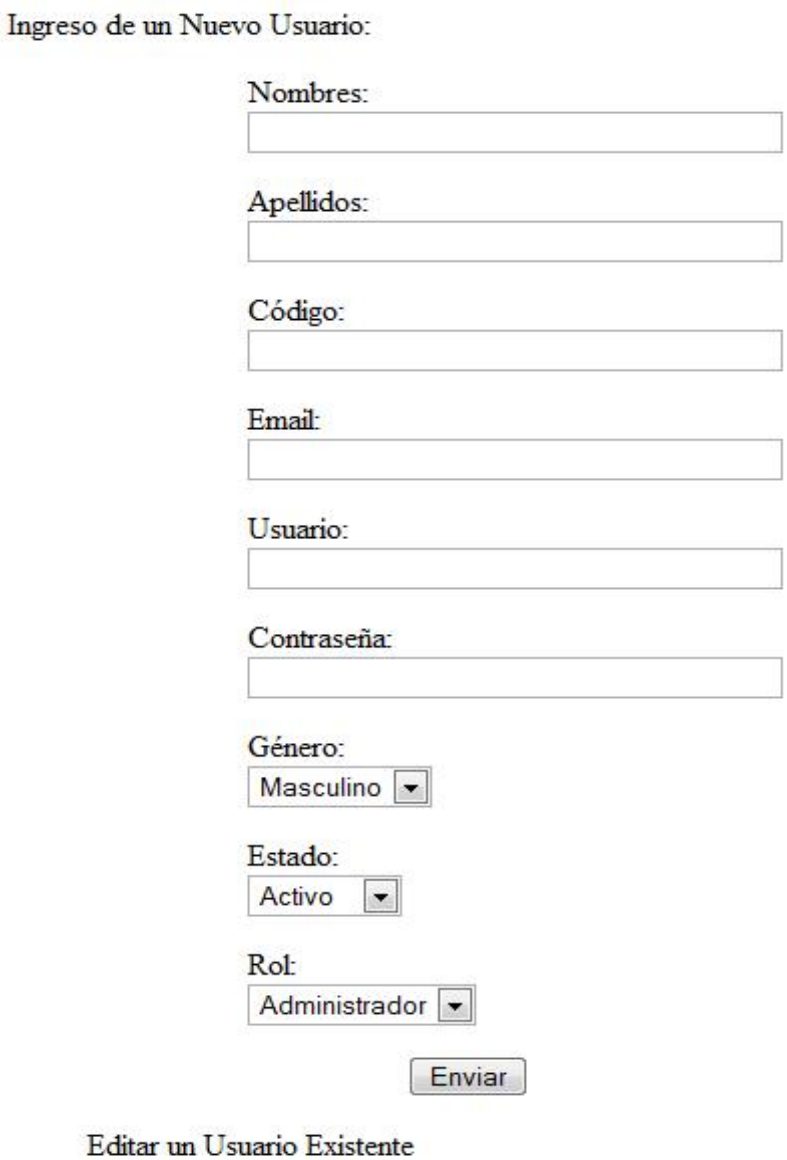


Figura 17. Datos para el ingreso de un nuevo usuario

- Editar usuarios

  El administrador tiene la capacidad de modificar los datos básicos de los usuarios, así como de los datos pertinentes a su estado en el sistema. Para esto se utiliza el mismo formato de la figura anterior.

- Crear repositorios

  El administrador puede crear nuevos repositorios para poner un proyecto bajo el control de versiones. Además crea los permisos correspondientes para todos los usuarios que hacen parte del mismo. La siguiente figura muestra la interfaz con el usuario (Administrador).

Figura 18. Datos para la creación de un nuevo repositorio.

- Copia de seguridad de los repositorios

  El administrador puede realizar la copia de seguridad de los repositorios en el momento que sea conveniente a través de las funciones dadas por la herramienta sin tener que proveer al sistema ningún parámetro.

- Contáctenos

  El usuario en cualquier momento puede ingresar a la página “Contáctenos” en la cual puede obtener información sobre como ponerse en contacto con el administrador de la herramienta y responsable de servidor.

- Ayuda

  Un usuario puede entrar en cualquier momento a la página de ayuda, en la cual se explica la estructura y cada una de las funciones del portal.

8.2.4 Documentación del sistema

Los documentos incluidos en el proyecto son: Manual de usuario del Cliente Windows, Manual de usuario del cliente Linux, Manual de usuario del portal, Estructura del portal, Manual del controlador de versiones y el Manual del administración.

- Manual del usuario del Cliente Windows
  Está dirigido a los desarrolladores en ambiente Windows que requieren el servicio de control de versiones para sus proyectos y la documentación de los mismos.

- Manual del usuario del Cliente Linux
  Está dirigido a los desarrolladores en ambiente Linux que requieren el servicio de control de versiones para sus proyectos y la documentación de los mismos.

- Manual de usuario del portal
  Está dirigido a los usuarios del portal y contenido también en la ayuda del mismo.

- Estructura de archivos del portal
  Esta dirigido al administrador del portal, quien estará encargado de la manipulación de las carpetas y archivos del portal y el manejo de la base de datos.

- Manual del controlador de versiones
  Documento dirigido al administrador del servidor, guía el proceso de instalación de cada uno de los paquetes necesarios para el correcto funcionamiento del controlador de versiones. Además la explicación de los comandos necesarios para manejar de forma correcta los repositorios desde de básico hasta los casos especiales.

- Manual del administrador
  Este documento está dirigido al administrador, los programadores o personal técnico encargado del soporte en el servidor, incluye toda la documentación del portal y el curso normal de las acciones para que este funcione correctamente.

8.3 Implementación

En esta etapa se lleva a cabo la codificación y depuración del portal, se implementan las páginas de acuerdo con el análisis y diseño descritos en las sesiones anteriores.

Para este objetivo se siguen los parámetros definidos en el modelo de datos a fin de tener un producto de mejor calidad y la posibilidad de dar escalabilidad a los servicios y la aplicación en general para dar paso a futuros proyectos.

Se realiza el desarrollo de la aplicación para permitir a los usuarios sacar el mayor provecho de la misma de una manera sencilla y amigable, puesto que es un gran cambio para la metodología que se lleva a cabo en los proyectos actualmente.

Se desarrollaron los formatos de entrada y salida acordes al diseño adelantado, para dar al usuario una forma fácil de comunicación con el sistema y la aplicación en general. También se desarrollaron los menús, la validación de los datos proporcionados y las salidas correspondientes, los mensajes de error y la ayuda.

En esta etapa de forma paralela al desarrollo de la aplicación Web se dio seguimiento a los procesos de instalación y configuración de las herramientas y el servidor para poder tener una interacción completa con los repositorios y sus opciones en el caso de una posible migración del servidor a otro sistema operativo. De esta forma se reestablecería en poco tiempo el servicio a la comunidad.

Se desarrollo con el lenguaje interpretado PHP y con el manejador de base de datos Postgresql, y como no es necesario un conector entre ellos la transmisión de los datos es más veloz.

La aplicación tiene un peso total de 6.35 MBytes con los archivos de control de versiones. A continuación se muestra la estructura de los archivos que fueron desarrollados.

Estructura de archivos del portal

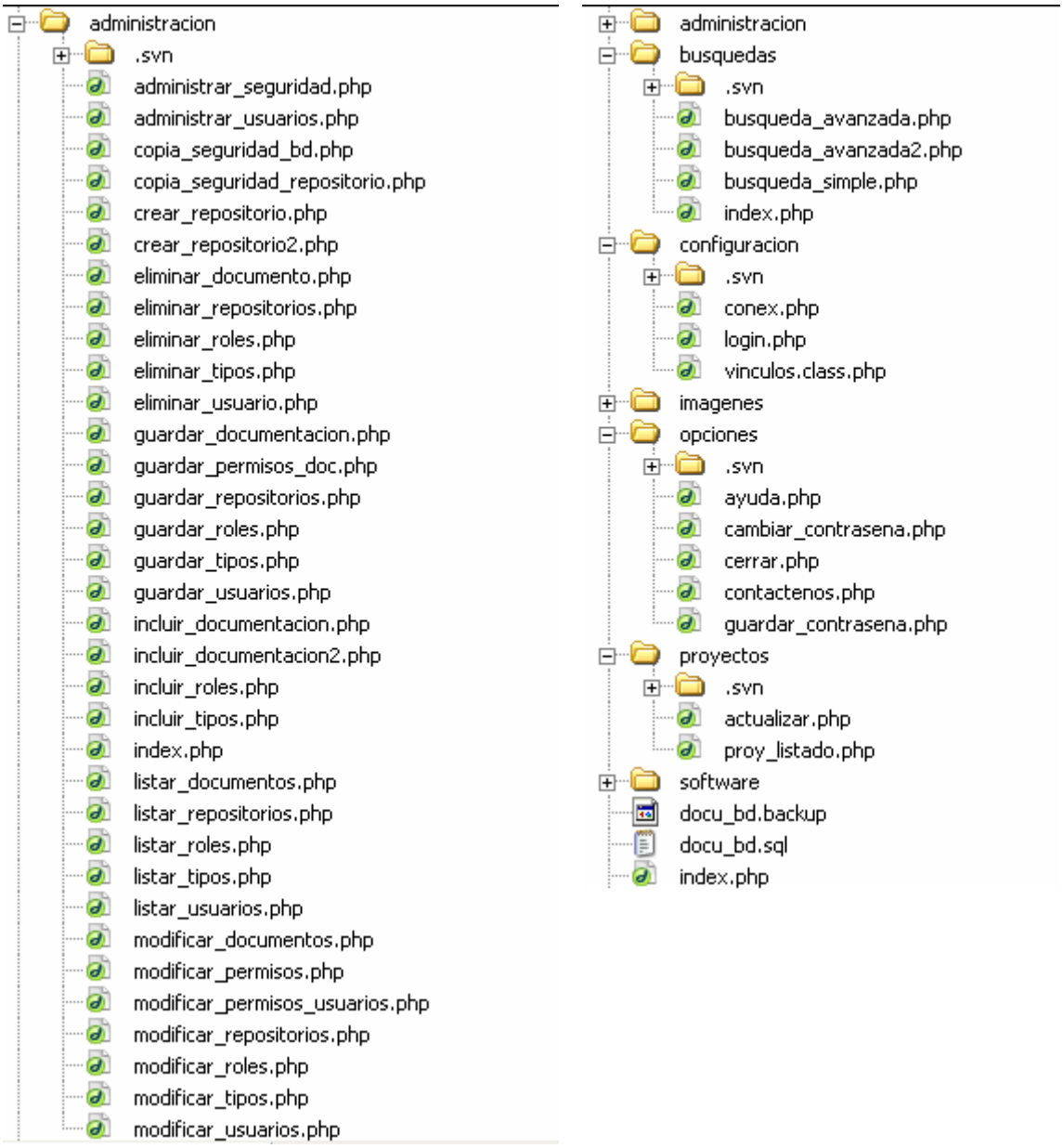


Figura 19. Estructura de Archivos

8.3.1 Codificación

A continuación se describen las funcionalidades que tiene el portal, referentes a cada servicio.

- Página: index.php

Funcionalidad: Esta página da la bienvenida a la aplicación y explica de forma sencilla cada uno de los servicios que puede obtener un usuario (no administrador), los cuales son búsquedas, proyectos y software.

Búsquedas

Este servicio es prestado a cualquier persona que lo necesite, sin embargo los resultados de las búsquedas son selectivos según el rol del usuario que ha iniciado la sesión, si el usuario no inicio sesión o es un usuario invitado los resultados de las búsquedas serán los documentos que no comprometan la seguridad de alguna aplicación o proyecto. Según la complejidad del rol del usuario las búsquedas aumentan en descripción y seguridad.

- Página: busquedas/index.php
  Funcionalidad: Esta página contiene principalmente un formulario sencillo que se compone de un campo de texto y un botón en el cual el usuario puede introducir un parámetro para realizar una búsqueda en forma general. En esta página también se encuentra el vínculo hacia la página de búsqueda avanzada.

- Página: busquedas/avanzada.php
  Funcionalidad: En esta página se muestran varios campos y opciones, de las cuales el usuario puede especificar las que desee para realizar una búsqueda más precisa.

- Página: busquedas/resultados.php
  Funcionalidad: Esta página muestra los resultados obtenidos de las búsquedas realizadas en cualquiera de las páginas anteriores. En ella se describen las características principales del documento que cumple con todos los requisitos que el usuario especificó, y un vínculo para poder abrir el documento en cuestión.

Proyectos

Esta sección es solo para usuarios vinculados a algún proyecto bajo la administración del sistema de control de versiones. Un visitante podrá ver la lista de proyectos pero no podrá ingresar a ninguno. La autenticación de este sitio es directamente realizada por el sistema de control de versiones.

- Página: proyectos/proy_listado.php
  Funcionalidad: Esta página muestra un listado actualizado de los repositorios dentro del sistema de control de versiones. Cada uno de estos repositorios se refiere a un proyecto en desarrollo bajo el control del sistema. Muestra también el tipo de base de datos utilizado. Cada ítem corresponde a un vínculo que será autenticado por el controlador de versiones para su respectivo ingreso.

Software

El servicio de software es prestado a cualquier persona que desee descargar los clientes gráficos del sistema de control de versiones. Los clientes gráficos corresponden a los diferentes sistemas operativos. Cualquier usuario puede descargarlos sin problemas puesto que estos programas tienen licencia libre. En este módulo también es posible acceder a los respectivos manuales de usuario.

- Página: software/index.php
  Funcionalidad: En esta página un usuario, sin importar su rol, puede descargar los clientes gráficos del sistema de control de versiones, dependiendo del sistema operativo en el que desarrolle sus aplicaciones. También se encuentra el vínculo a los respectivos manuales.

- Página: software/manual_tortoise.php
  Funcionalidad: En esta página el usuario puede visualizar o guardar el manual de usuario del cliente gráfico del controlador de versiones desarrollado para Windows XP, desarrollado para este proyecto.

- Página: software/manual_rapid.php
  Funcionalidad: En esta página el usuario puede visualizar o guardar el manual de usuario del cliente gráfico del controlador de versiones desarrollado para Windows y Linux, desarrollado para este proyecto.

Administración

El vínculo a este módulo solo es visible si en usuario que se ha registrado en la sesión es administrador. En este módulo se hace realizan las operaciones criticas del sistema.

- Página: administracion/index.php
  Funcionalidad: En esta página el administrador puede ver las todas opciones que tiene para gestionar.

- Página: administración/incluir_documentacion.php
  Funcionalidad: El administrador introduce en la base de datos la información referente a un documento que ha sido introducido al control de versiones en el repositorio de documentación del servidor.

- Página: administracion/administrar_seguridad.php
  Funcionalidad: En esta página el administrador puede crear un usuario tanto para el acceso a las búsquedas sobre la base de datos, como para

la autenticación en los repositorios, para este fin se utiliza el comando “htpasswd”.

- Página: administracion/modificar_usuario.php
  Funcionalidad: En esta página el administrador puede modificar los datos básicos dados por un usuario, así como los referentes al sistema asignados con anterioridad, tales como rol y estado.

- Página: administracion/roles.php
  Funcionalidad: En esta página el administrador pude acceder a la información de los roles existentes y acceder a la creación y modificación de los mismos.

- Página: administracion/tipos.php
  Funcionalidad: En esta página el administrador puede acceder a la información sobre los tipos de documentos almacenados en la base de datos, así como podrá acceder a creación, borrado y edición de los mismos.

- Página: administracion/crear_repositorio.php
  Funcionalidad: El administrador puede crear un nuevo repositorio para los proyectos que se desarrollan en la EISI y dar acceso al mismo a los usuarios que han sido registrados.

- Página: administracion/copia_seguridad_repositorio.php
  Funcionalidad: En esta opción el administrador puede realizar una copia de seguridad de los repositorios del controlador de versiones en el momento que el decida. Esta copia se realiza en “caliente”, esto quiere decir que no es necesario detener las entradas y salidas de datos de los repositorios para hacer la operación.

Opciones

Estas son algunas de las opciones que los usuarios tienen en cualquier momento durante la estadía en la aplicación.

- Página: opciones/ayuda.php
  Funcionalidad: En esta página los usuarios pueden consultar sobre las dudas referentes a la funcionalidad de la aplicación.

- Página: opciones/contactenos.php
  Funcionalidad: Desde esta página los usuarios pueden comunicarse con el administrador a través del correo electrónico. Para obtener respuesta a las inquietudes o sugerencias el usuario debe poner un correo valido al cual el administrador podrá comunicarse.

- Página: opciones/cambiar_contrasena.php
  Funcionalidad: Un usuario puede cambiar el nombre de usuario y la contraseña que le ha sido dada por parte del administrador por primera vez, de igual manera el usuario puede cambiar esta contraseña cuantas veces lo desee.

- Página: opciones/cerrar_sesion.php
  Funcionalidad: Esta página muestra la confirmación al usuario cuando esta ha finalizado su sesión.

- Inicio
  Este vínculo regresa al usuario a la página “index.php”.

- Cerrar
  En esta opción la aplicación se cierra y si el usuario no cerró sesión el sistema la finaliza.

8.3.2 Manejo de Sesiones

Para que el portal sea una aplicación confiable el sistema debe asociar las peticiones con el usuario que en verdad las realizó. Para esto se utiliza el manejo de sesiones, el cual se inicia en el momento en el que el usuario escribe su nombre de usuario y contraseña y el sistema verifique su registro en la base de datos. El seguimiento de la estadía en la aplicación se realiza a través de variables de sesión en las que se almacena la información del usuario. Este seguimiento termina cuando el usuario cierra sesión o cierra la aplicación.

- Página: configuracion/login.php
  Funcionalidad: Contiene las funciones que se utilizan en todo el portal para el manejo de sesiones de usuario. Así como la función que cambia el estado de un usuario a inactivo cuando ha introducido mal la contraseña un número determinado de veces, dado en la configuración.

Conexión a la base de datos

Para la utilización de la base de datos en postgresql se necesita tener una conexión activa, para ello se utiliza una página con las funciones necesarias.

8.4 Pruebas

La fase final de cada una de las iteraciones se refiere a las pruebas; éstas son diseñadas con el fin de verificar el correcto funcionamiento de cada uno de los servicios ofrecidos, evaluar la calidad del producto desarrollado y evitar las posibles fallas que se pudieran presentar.

El desarrollo del portal a forma de módulos permite hacer el primer filtro a medida que se desarrolla cada uno de ellos. Cada vez que se termina uno de estos servicios se asegura su funcionalidad y se reduce la probabilidad de errores. Después de la programación se realizan las pruebas al sistema integrado.

Para el desarrollo de las pruebas es necesario llenar registros en la base de datos. Estos registros se realizan con datos simulados y al término de las mismas son eliminados para comenzar con la inserción real de los datos que se utilizaran en el sistema.

Se tuvieron en cuenta para realizar las pruebas:

- Corroborar que todos los caminos posibles de un proceso obtengan la salida correspondiente a la entrada dada.

- Detección las fallas en la interfaz de usuario.

- Verificar la correcta carga y descarga de los datos.

- Verificar la navegabilidad desde y hacia todas las páginas de la aplicación.

- Detectar los posibles errores que pueden ocurrir en la conexión a la base de datos.

- Evaluar la calidad de las validaciones de los datos que son ingresadas a la base de datos.

- Verificación de las sentencias SQL

- Verificación de las consultas SQL

- Ejecución de los comandos en Linux

- Envío de los formularios.

8.4.1 Pruebas a los servicios

Servicio de búsquedas

- Ingreso al servicio desde el menú principal

- Carga del diseño de la página (imágenes y texto)

- Criterios de la búsqueda: parámetros simples

- Validación de los criterios de la consulta: No debe estar en blanco

- Verificación del rol del usuario que realizo la petición

- Resultados obtenidos: Verificación de los resultados arrojados por la consulta, datos relacionados con los documentos que cumplen con los parámetros de búsqueda. Verificación de los permisos dados para el rol del usuario.

- Verificación del correcto funcionamiento del vínculo de los documentos de la salida.

- Verificación del vínculo a la página de búsquedas avanzadas.

- Carga del diseño de la página (imágenes y texto)

- Criterios de la búsqueda: Titulo, autor, palabras clave y tipo de documento.

- Validación de los criterios de la búsqueda: el usuario debió escribir al menos uno.

- Verificación del rol del usuario que realizo la petición

- Resultados obtenidos: Verificación de los resultados arrojados por la consulta, datos relacionados con los documentos que cumplen con los parámetros de búsqueda. Verificación de los permisos dados para el rol del usuario.

- Verificación del correcto funcionamiento del vínculo de los documentos que se obtuvieron como resultado.

Servicio de Proyectos

- Verificación del ingreso desde el menú principal.

- Carga del diseño de la página (imágenes y texto).

- Verificación del listado de proyectos.

- Verificación del tipo de repositorio de cada proyecto.

- Verificación de los vínculos a cada uno de los repositorios.
- Autenticación por parte del sistema de control de versiones.
- Verificación de la copia de archivos de los proyectos.

Servicio de Software

- Verificación del ingreso desde el menú principal.
- Carga del diseño de la página (imágenes y texto)
- Verificación de los vínculos de las herramientas a descargar para cada sistema operativo.
- Verificación de los vínculos a los manuales de las herramientas para los proyectos de la escuela.
- Carga del diseño de la página del manual correspondiente.
- Verificación del manual correspondiente a cada herramienta.

Servicio de administración

- Verificación del usuario y contraseña dados por el administrador.
- Verificación del ingreso desde el menú principal.
- Carga del diseño de la página (imágenes y texto).
- Verificación de los vínculos a cada una de las opciones que puede obtener el administrador.
- Carga del formulario de ingreso y edición de los documentos.
- Validación de los datos dados por el administrador.
- Verificación de la inserción de los datos en la base de datos.
- Carga del formulario de ingreso y edición de datos de usuarios.
- Validación de los datos ingresados por el administrador.

- Verificación de la inserción de los datos en la base de datos.

- Verificación de la inserción del nombre de usuario y contraseña en un archivo de autenticación del sistema de control de versiones.

- Carga del formulario de nombre de nuevo repositorio y usuarios autorizados.

- Validación de los parámetros ingresados por el administrador.

- Verificación de la inserción de los datos en la base de datos.

- Verificación de la creación del nuevo repositorio.

- Verificación de la creación de los permisos en el archivo de autenticación del sistema de control de versiones.

- Verificación de la copia de seguridad de los repositorios del sistema de control de versiones.

8.4.2 Pruebas de integración

Luego de terminada la aplicación por completo se verifica que en todas las páginas donde sea necesario lo siguiente:

- Se realiza correctamente la carga y presentación de los menús de acuerdo al rol de usuario registrado.

- El sistema realiza correctamente las consultas a la base de datos de acuerdo con los paramentos dados por el usuario que solicita la acción.

- Cada usuario puede acceder solo a las páginas autorizadas para su rol.

9 Pruebas Integrales

Para realizar las pruebas correspondientes al sistema de control de versiones el portal Web fue desde el principio administrado sobre un servidor de prueba con este sistema *SubVersion*. Para la administración desde Linux se utilizó RapidSVN y para la administración desde Windows XP fue utilizado TortoiseSVN. Los dos clientes gráficos para facilitar las operaciones sobre el repositorio del proyecto.

Para este fin fue creado el repositorio “subversion” y se realizaron las siguientes operaciones:

- Creación del Usuario “angelica” con rol estudiante.

- Creación del Usuario “Manuel” con rol profesor.

- Creación del repositorio “subversion” con permisos de lectura, escritura y ejecución para el usuario “Manuel” y permisos de lectura y escritura para el usuario “angelica”.

- Importar (import): Se realizó la importación de la versión inicial del proyecto.

- Ingreso al repositorio por el módulo proyectos del portal Web.

- Obtener (checkout): Se obtuvo una copia de trabajo para realizar los cambios sobre el proyecto.

- Se llevaron a cabo modificaciones en la copia local.

- Actualizar (update): Se realiza una sincronización de los cambios hechos al repositorio desde la última entrada por otros usuarios o desde otros equipos.

- Confirmar (commit): Se realiza después de la actualización, consiste en sincronizar los cambios hechos de forma local al repositorio.

- Renombrar (rename): Se utiliza para cambiar el nombre de una carpeta o archivo y no perder el historial que se ha creado de él.

- Añadir (add): Se utiliza para poner nuevos archivos bajo el control de versiones.

- Eliminar (delete): Se utiliza para borrar un archivo o carpeta del control de versiones.

- Mostrar registro (log): Muestra el historial que se va creando con cada una de las confirmaciones hechas; está dividido por revisiones. La complejidad de este historial depende directamente de la responsabilidad de los desarrolladores del proyecto.

- Actualizar al sitio de prueba: crea una copia de trabajo en el espacio de prueba para los portales de Cormoran. (usuario “angelica”)

- Actualizar al sitio: Exporta (export), al sitio de producción dentro del servidor Cormoran. (usuario “Manuel”)

10 Conclusiones

- Después de realizar un análisis detallado sobre diferentes herramientas de control de versiones, y basado en las necesidades de la EISI, se determinó que *SubVersion* era la más apropiada, puesto que brinda a los usuarios las facilidades que se buscan en este proyecto con una interacción sencilla y un tiempo de espera adecuado.

- La implementación de un servicio de control de versiones de software sin costo para la EISI, es posible gracias a la utilización de herramientas de libre distribución como *Linux*, *PHP*, *Postgres*, *apache* y *SubVersion*.

- Al manejar políticas de control de versiones en cualquier ambiente de desarrollo, es posible centralizar los procesos de una forma eficiente, y este tipo de manejo permite tener un enfoque metodológico en la creación de proyectos y seguir modelos de desarrollo conservando versiones anteriores con gran facilidad.

- Con la implementación del sistema, las personas involucradas en los proyectos de la EISI, tienen la posibilidad de ver tanto sus cambios, como los cambios realizados por otro miembro del grupo, en cualquier momento, desde cualquier lugar conectado a Internet a través de la página Web desarrollada y sin necesidad de instalar alguna aplicación adicional.

- Cuando los proyectos sean portales Web alojados en el servidor Cormorán, los miembros de proyectos tendrán la posibilidad de actualizar una versión de pruebas y una versión estable, con una operación simple en la página Web del proyecto, mientras cuenten con la autorización adecuada para llevar a cabo la operación.

- Se tiene disponible documentación en línea del servidor, con la seguridad adecuada para cada tipo de usuario del sistema, y la implementación del sistema de búsqueda desarrollado facilita que su utilización sea más eficaz.

- La creación de manuales de usuario de las herramientas utilizadas, especialmente enfocados al desarrollo de software y documentación de los proyectos de nuestra Escuela, y manual de administración sencillos facilitarán la labor realizada por la persona encargada del servidor, así como de los usuarios del sistema en general.

- Con sistemas de control de versiones se cambia el modo de realizar software tradicional, a desarrollo de software por comunidades permitiendo la sincronización y la independencia del espacio y el tiempo de los desarrolladores.

- Utilizando en conjunto *SubVersion* y el portal web desarrollado, se pasará de tener usuarios del sistema operativo, a registros de usuarios en la base de datos y de esta forma se aumenta la seguridad en el servidor y se pueden verificar los cambios realizados por cada miembro del equipo de trabajo en forma individual.

- Es posible recuperar el sistema en cualquier momento ya que la información de actualización se encuentra de forma centralizada y distribuida, el repositorio puede sincronizarse con los últimos cambios de cualquier copia de trabajo de los proyectos y se puede programar un sistema de respaldo seguro y sencillo de realizar.

11 Recomendaciones

- Respecto a los sistemas de control de versiones, nuestra Escuela y Facultad deberían considerar formar a los alumnos y profesores en el uso de las herramientas de este tipo. En este sentido los estudiantes pueden recibir formación para la realización de las prácticas y proyectos de clase; Además la Escuela no tendría que incidir en costos de licencias puesto que existen excelentes soluciones de software libre, como la presentada en este proyecto.

- A medida que sea probada la efectividad del proyecto se podrá realizar una integración con el portal de la EISI o poner un vínculo de acceso a la página y mejorar el diseño visual de la misma para hacerla más atractiva a los usuarios de la comunidad.

- Continuar con el desarrollo de otras versiones del software y la implementación de servicios como el de estadísticas que permiten analizar el trabajo realizado por cada usuario dentro de un grupo de desarrollo, el nivel de uso de las herramientas de control de versiones, y en el módulo de proyectos la actualización de sitios por directorios o archivos individuales.

12 Bibliografía

- Beck, Kent.. Extreme Programming Explained: Embrace Change: Addison-Wesley Pub Co, 1999.

- Carcamo, José.. Bases de datos relacionales: Ediciones UIS, 1997.

- Collins-Sussman, Ben.. Fitzpatrick, Brian W... Pilato, C. Michael.. Version Control with SubVersion: For SubVersion 1.3 : Published (TBA) 2006.

- Flowler, Martin.. Refactoring: Improve the Design of Existing Code: Addison-Wesley Pub Co, 1999.

- Küng, Stefan.. Onken, Lübbe.. Large, Simon.. TortoiseSVN: Un cliente de SubVersion para Windows:Version 1.3.5, 2006.

- Newkirk, James.. Robert C. Martin.. La Programación Extrema en la práctica: Addison-Wesley Iberoamericana Espanya, S.A. - 2002

- Piattini, Mario.. Calvo, José.. Cervera, Joaquín.. Fernandez, Luis.. Análisis y Diseño de Aplicaciones Informáticas de Gestión. Alfaomega, 2004.

- Ratschiller, Tobias.. Gerken, hill.. Creación de aplicaciones Web con PHP 4: Prentice Hall, 2001.

- Tanenbaum, Andrew S.. Sistemas Operativos Modernos. Mexico: Prentice Hall, 1993.

Bibliografía en Internet

- http://subversion.tigris.org
- http://tortoisesvn.tigris.org
- http://rapidsvn.tigris.org
- http://hunter.campbus.com
- http://httpd.apache.org
- http://www.xprogramming.com
- http://en.wikipedia.org/wiki/List_of_revision_control_software
- http://said.fundacite-merida.gob.ve/evento/desarrollo_colaborativo.pdf

Anexo A. MANUAL DE RAPIDSVN

RapidSVN es una interfaz grafica multiplataforma del sistema de control de versiones SubVersion. Funciona sobre Windows, la mayoría de distribuciones de Linux, pronto disponible para Mac OS y otros sistemas operativos.

Utiliza wxWidgets como plataforma grafica, razón por la cual es preferible que este instalado en la maquina de destino.

Aunque es una herramienta a la vez sencilla y poderosa, no es posible crear repositorios, gestionar acceso, ni realizar copias de seguridad. Para llevar a cabo estas operaciones se desarrollo el portal Web, además de proveer acciones que solo se pueden ejecutar en el servidor.

A continuación se describen la instalación y el ciclo de trabajo regular, además de otras funciones utilizadas con cierta frecuencia.

INSTALACIÓN EN LINUX

Existen varias formas de instalación del paquete dependiendo de la distribución en la que se este realizando.

Para la mayoría de distribuciones es posible encontrar con los instaladores de SubVersion, RPMs que permiten que la instalación y configuración no tome casi nada de tiempo, su mayor desventaja es que son instaladores de mayor tamaño que los de código fuente.

- Para Red Hat Linux Enterprise 3 y otras distribuciones se encuentra un RPM de la última versión estable dentro de los instaladores de SubVersion: http://subversion.tigris.org/project_packages.html

    1. Ubicar el RPM adecuado para la distribución utilizada.
    2. Descargarlo.
    3. Ubicarse en el directorio del RPM ya sea por comandos o de forma grafica.

4. De forma grafica, hacer doble clic sobre el icono y aceptar la instalación. En modo consola utilizar el comando:
   RMP –i rapidsvnX.X.X.rpm.

- Otro método es bajando y compilando el código fuente. Este se puede obtener de la página: http://www.rapidsvn.org/download/

  El paquete se llama normalmente rapidsvn-x.x.x.tar.gz

  1. Estando en una consola como usuario root, ubicarse en la carpeta que contiene el archivo.
  2. Ejecutar los sigientes comandos:
     gunzip rapidsvn-x.x.x.tar.gz
     tar –xf rapidsvn-x.x.x.tar
     cd rapidsvn.x.x.x
     ./configure
     make
     make install

- En la distribución Fedora se puede utilizar el siguiente comando y la instalación y configuración se hace correctamente.

  yum install -y rapidsvn

- En las distribuciones Debian y derivados sucede lo mismo con la siguiente instrucción.

  apt-get install rapidsvn

  ó

  aptitude install rapidsvn

INSTALACIÓN EN WINDOWS

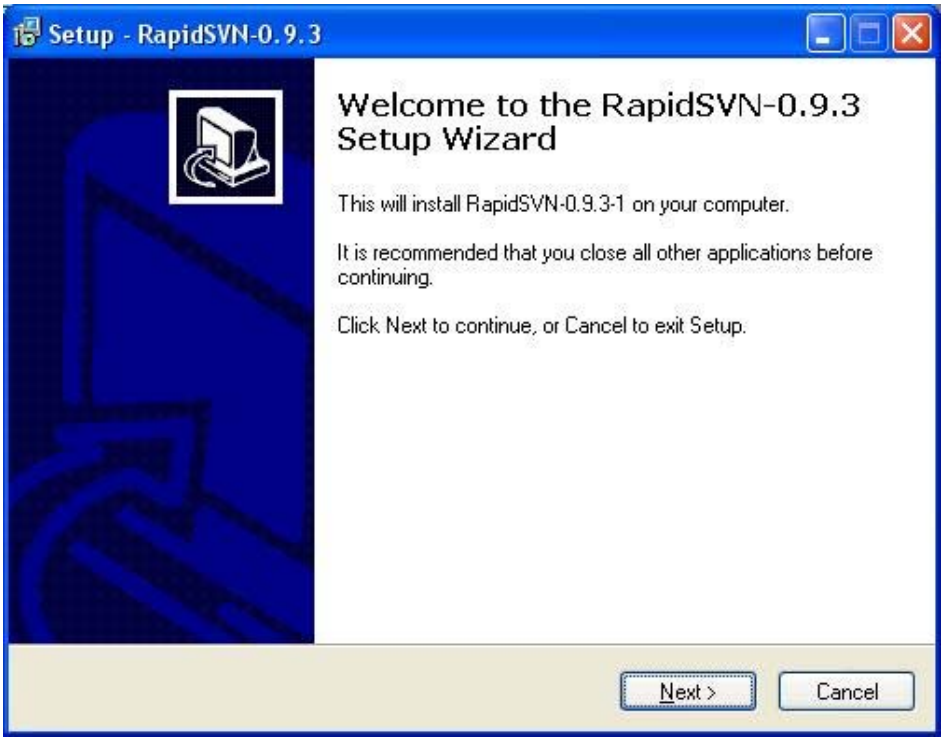
Para instalar el programa RapidSVN en Windows es necesario descargar el instalador binario para win32 ó win64, según el caso. Es posible bajar la última versión de la página http://www.rapidsvn.org/download.

El paquete se llama normalmente rapidsvn-x.x.x-x.exe

1. Dar doble clic sobre el icono

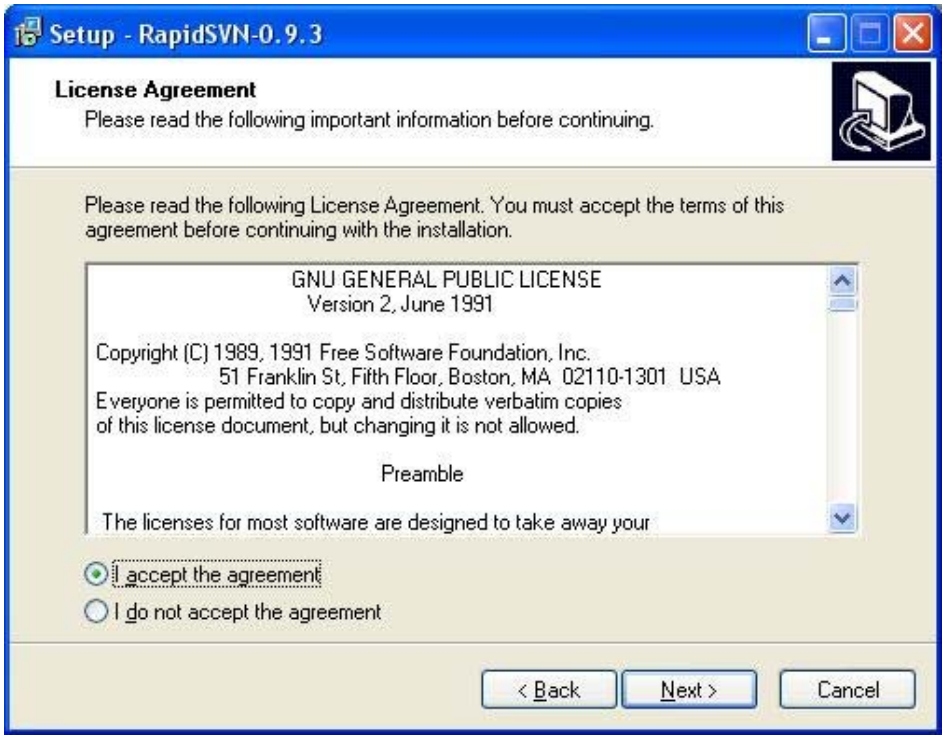
2. Se selecciona el lenguaje de la instalación (Inglés por defecto)

Se observa la pantalla de bienvenida a la instalación.



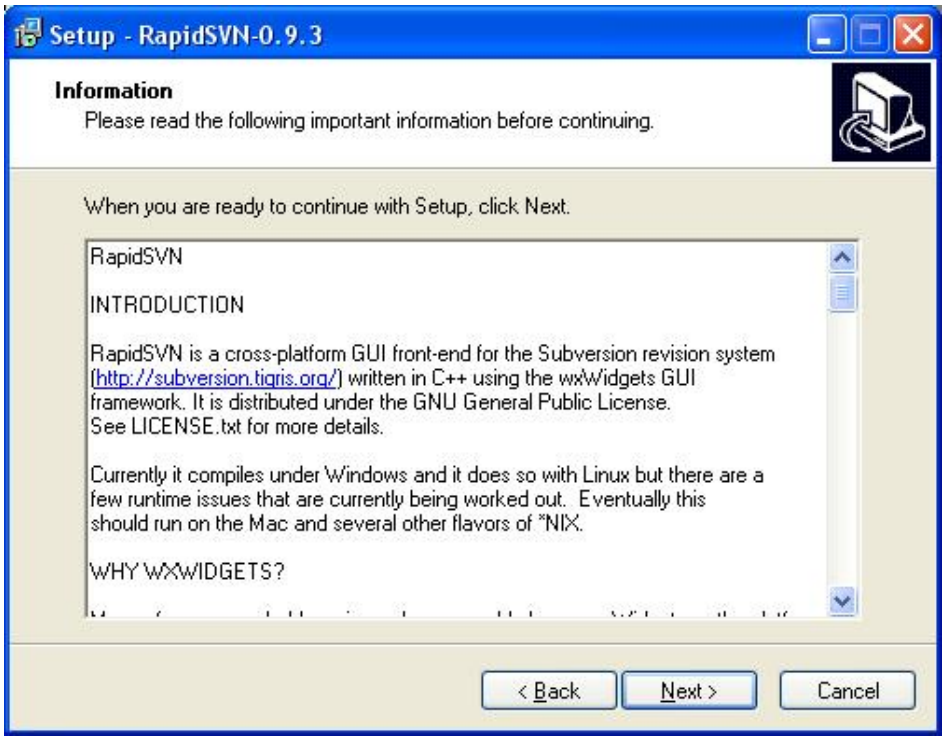
3. Para continuar oprimir “Next”

A continuación se muestra la pantalla de la licencia.



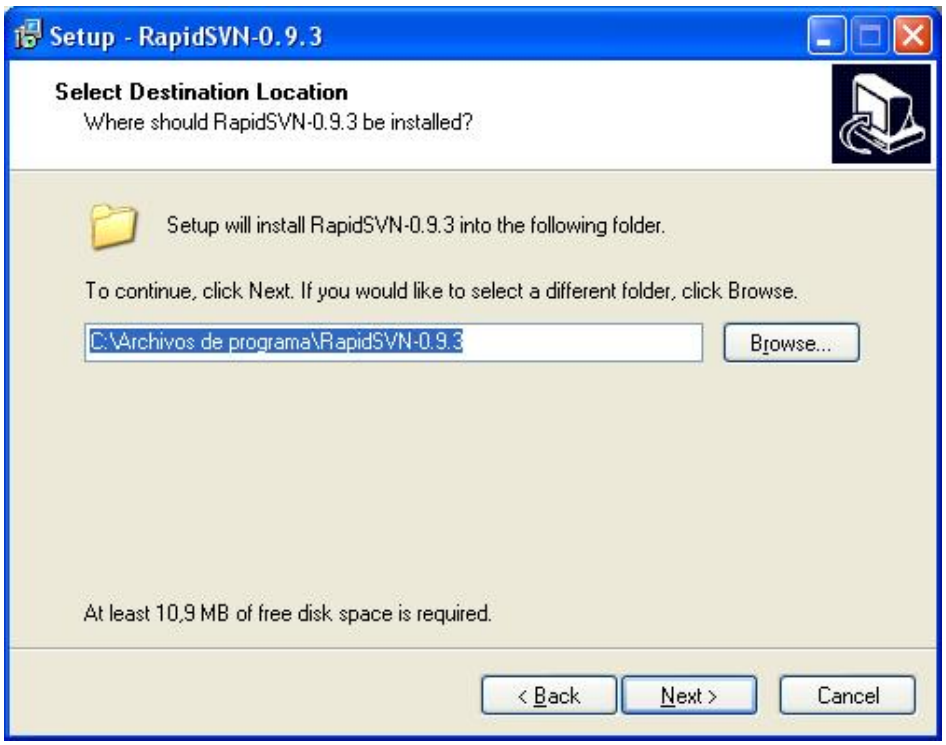
4. Para continuar se debe escoger “I accept the agreement” y oprimir “Next”.

Después se muestra una pantalla de información de RapidSVN



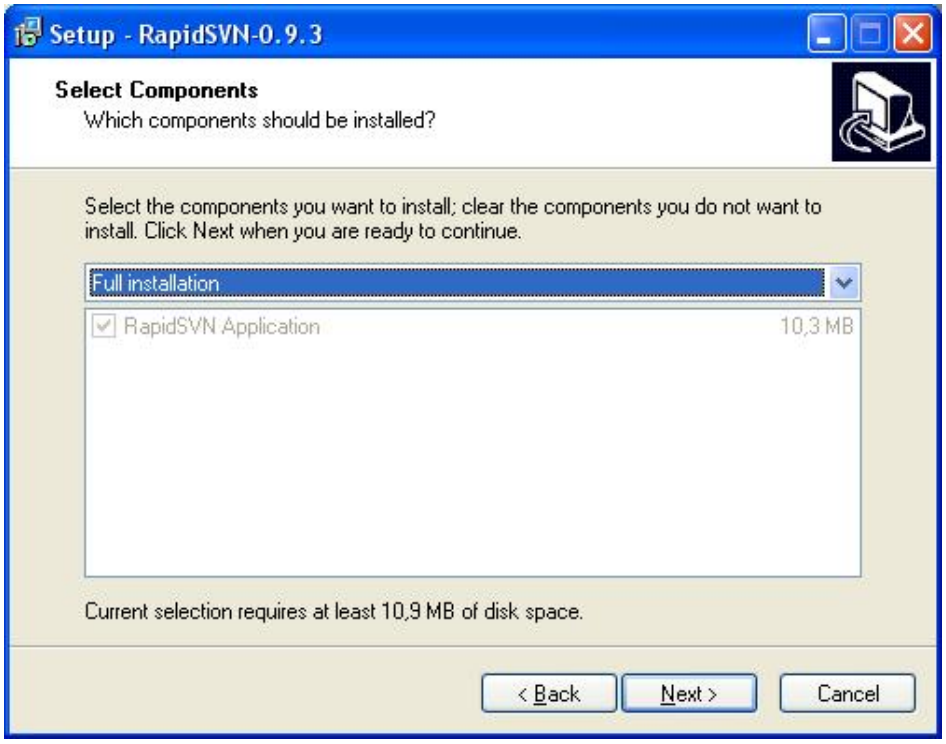
5. Para continuar oprimir “Next”

En la siguiente pantalla se escoge el destino de la aplicación.

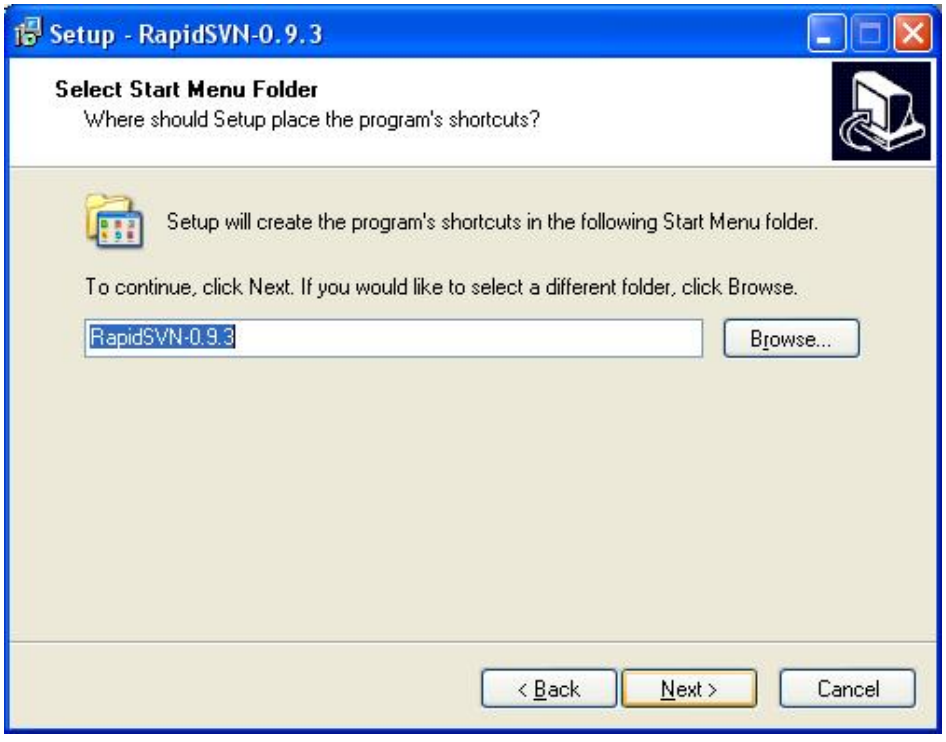


6. Normalmente el archivo por defecto es C:/Archivos de programa/RapidSVN-X.X.X, pero este archivo puede ser cambiado.

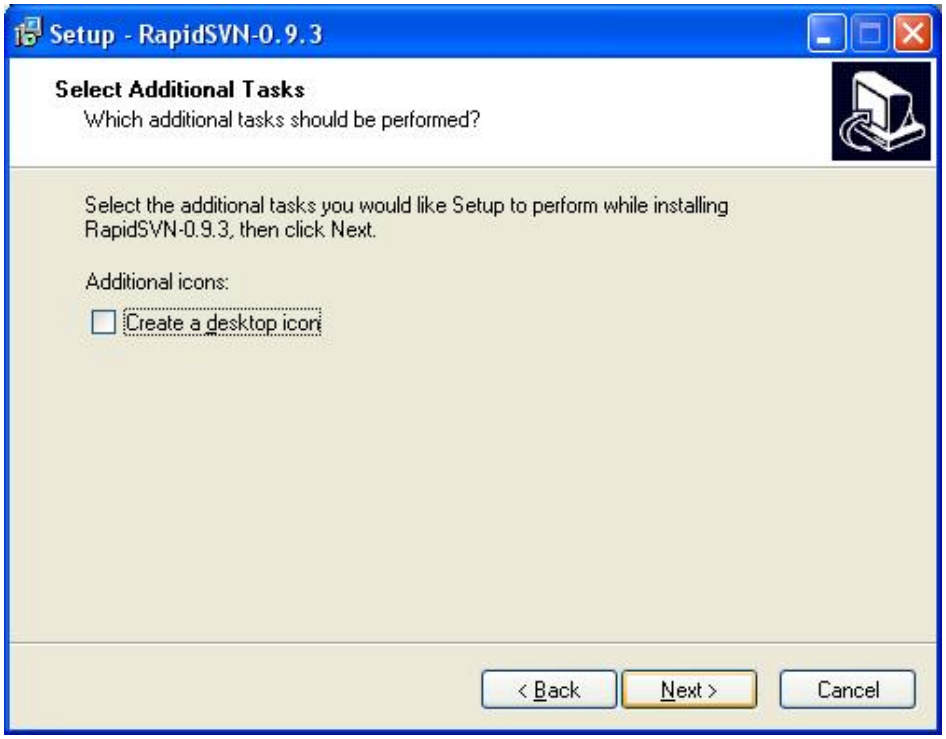
7. En la siguiente pregunta se pregunta sobres los componentes de la aplicación escoger “Full installation” en el menú y oprimir “Next”.



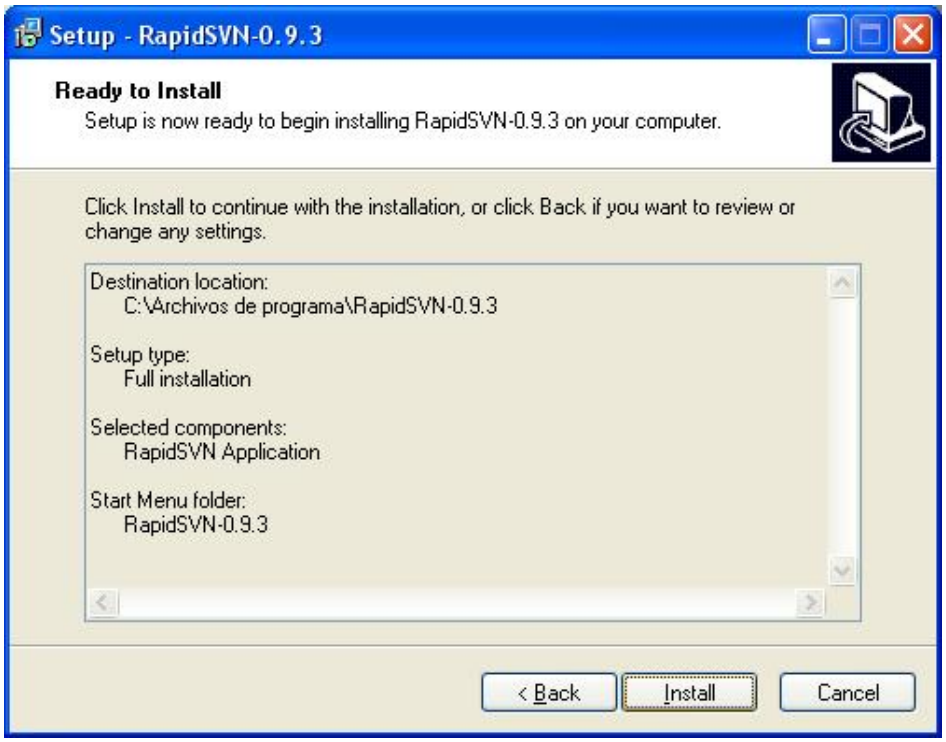
8. En la siguiente pantalla se escoge la carpeta para el menú. Clic en “Next” para continuar



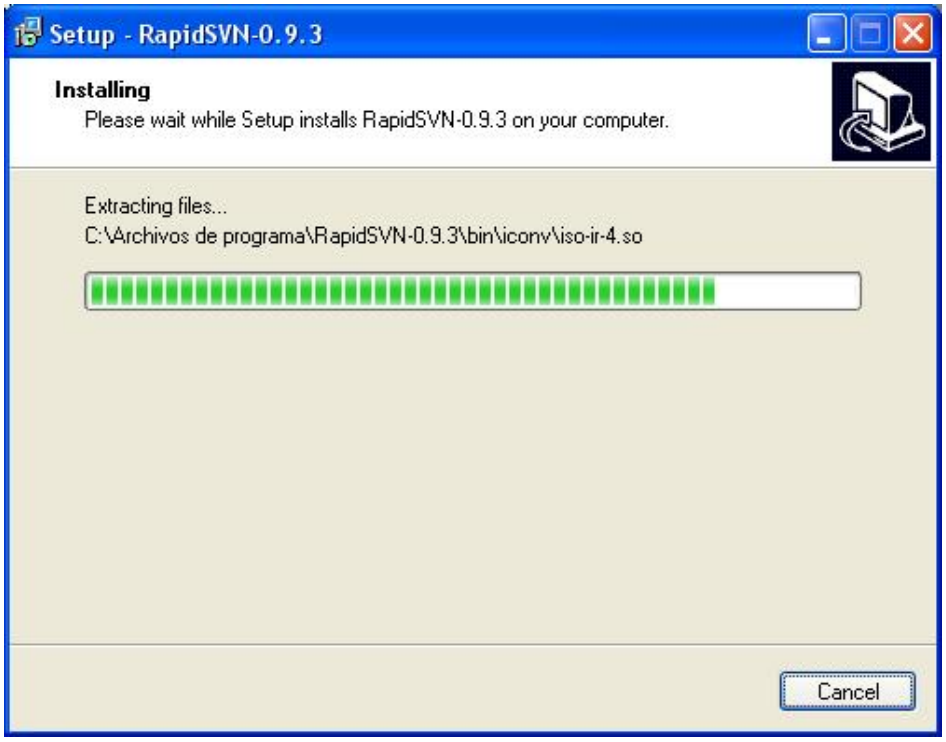
9. En la siguiente pantalla se escoge el icono de escritorio. Clic en “Next” para continuar.



10. En la siguiente pantalla se muestra lo que se ha seleccionado. Si las opciones son correctas, hacer clic en “Install”.

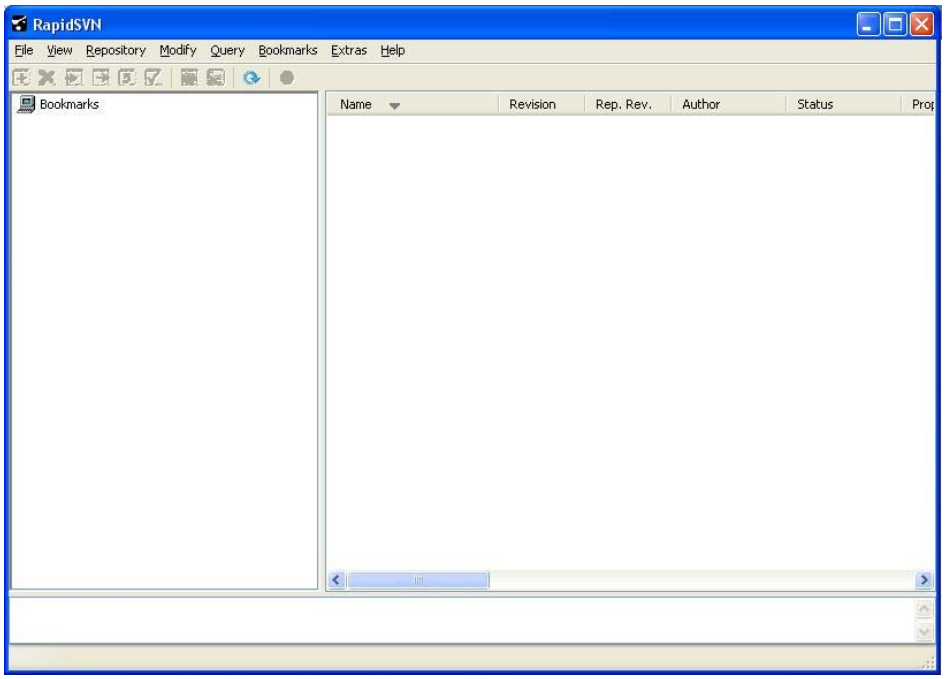


La siguiente pantalla muestra el proceso de instalación en el computador



11. La siguiente ventana confirma la realización de la instalación. Para terminar clic en “Finish”.

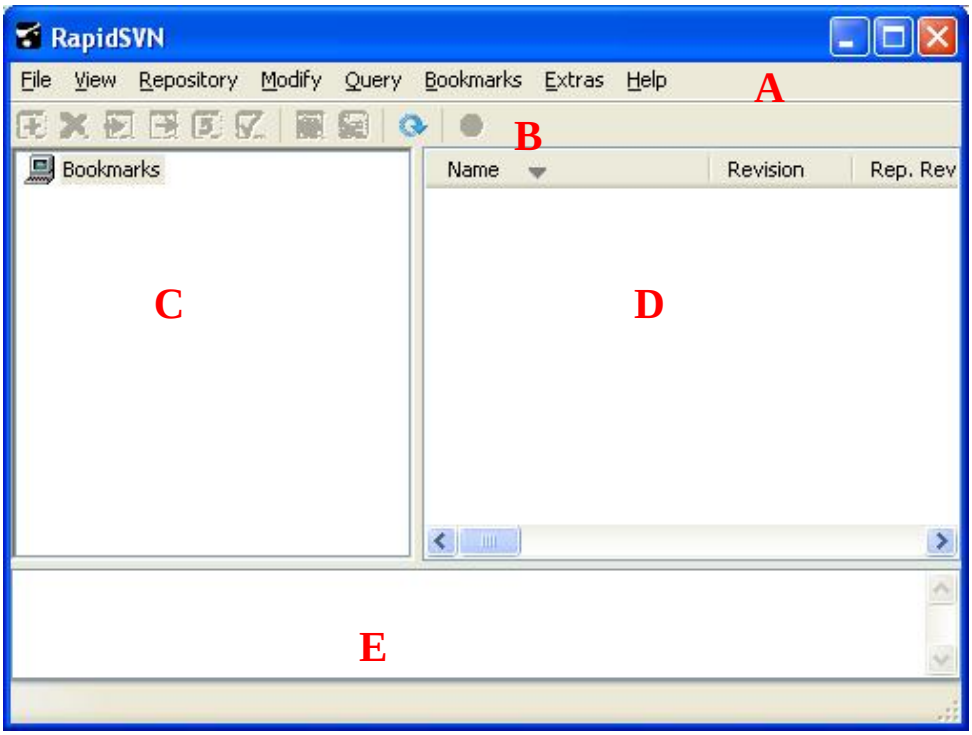
Al terminar la aplicación esta correctamente instalada y lista para empezar el trabajo.



INTERFAZ DE USUARIO DE RAPIDSVN

Para iniciar el programa es posible ejecutar el comando rapidsvn en una consola ó buscar en el menú Programación de Aplicaciones.

La interfaz de usuario de RapidSVN es muy simple y se compone de 5 áreas principales.



Menú Principal (A)

Da acceso a todas las opciones del programa

Menú de Iconos (B)

Da acceso rápido a las opciones de actualización, envió, resolución de conflictos, obtención de información y otras.

Lista de Marcas (C)

Marcas del repositorio (carpetas)

Lista de Carpetas y Archivos (D)

Desde este panel se accede a los archivos locales

Panel de Actividad (E)

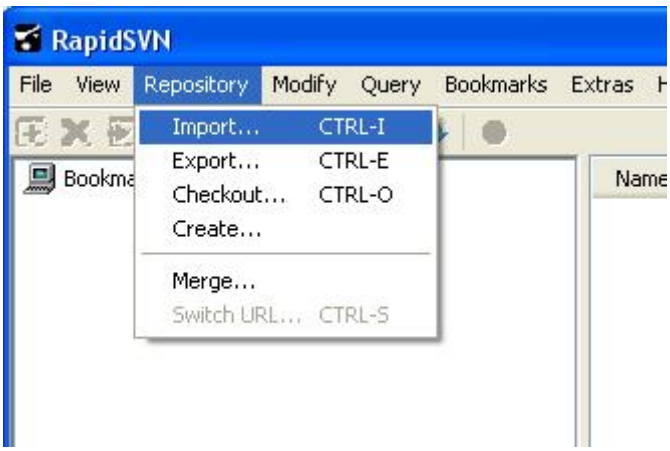
Muestra la actividad entre la copia local y el repositorio central.

IMPORTAR EL CONTENIDO INICIAL

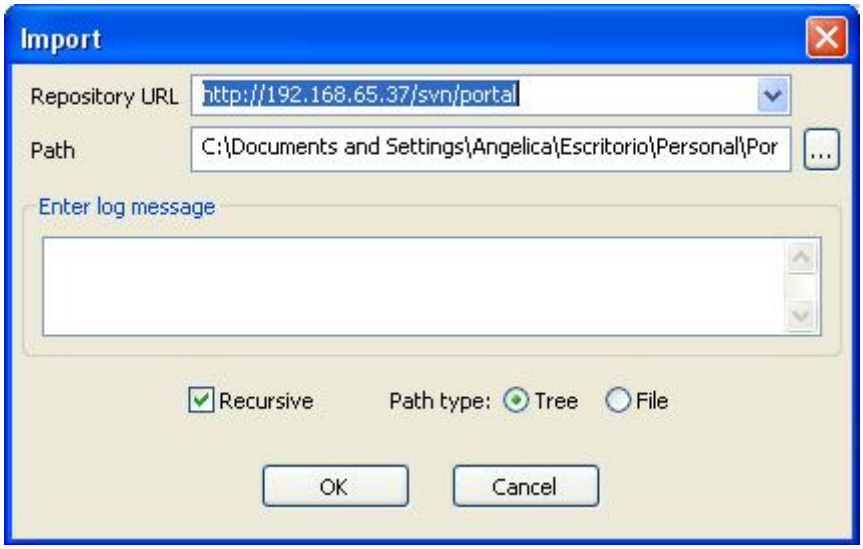
Inicialmente el repositorio esta completamente vació y muchas veces es necesario introducir en el la información que el proyecto tiene inicialmente. Aunque es recomendable empezar el proyecto de cero y no hacer este procedimiento. Se aclara que esta operación se hace solamente una vez.

(NOTA: Este término puede ser un poco confuso. Importar significa subir los datos al servidor).

1. Seleccionar Import del menú Repository.



2. Luego de esto se abrirá la siguiente ventana. En la URL se pone la dirección del repositorio que da el administrador.



Para la información del “Path” se puede hacer clic en [...] para que se abra la siguiente ventana.

3. En esta ventana buscar la carpeta que contiene el proyecto que se va a mantener en el sistema de control de versiones.

4. Para continuar hacer clic en “OK” el proceso se mostrara en el panel de actividades y mostrara que se a completado exitosamente con un la palabra “Ready”.

Cuando estos pasos se han realizado, la información ya se encuentra en el repositorio central, pero es necesario obtener una copia de esta para trabajar en ella de forma local. Esto se vera en el siguiente capitulo.

CREAR UNA COPIA DE TRABAJO LOCAL

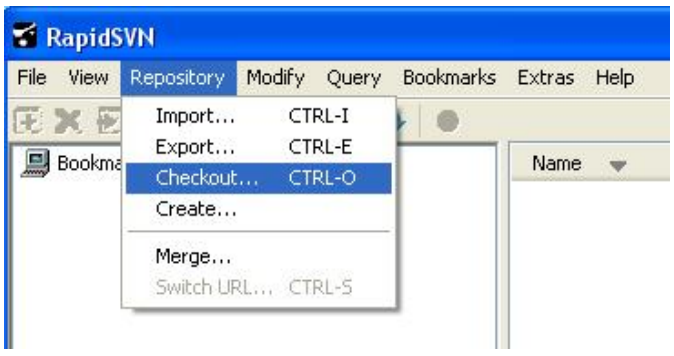
A fin de editar o crear aplicaciones y documentos, será necesario crear una copia local de lo contenido en el repositorio central para trabajar de forma sobre él. Esta copia es completamente independiente del repositorio central, lo que significa que cualquier cambio que se haga sobre ella no tendrá efecto en el repositorio central hasta que ellos no hayan sido enviados. A continuación se encuentran los pasos para crear una copia local en su computador.

Para realizar esta y las demás operaciones es necesario tener una conexión al menos temporal a Internet. Los cambios pueden tener lugar sin conexión alguna.

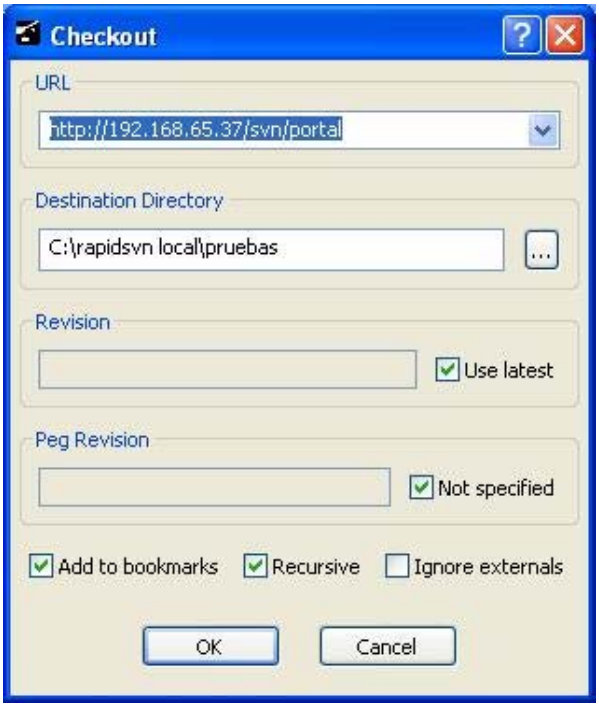
1. Crear una carpeta para contener la copia local. Para efectos de este manual llamaremos esa carpeta RapidSVN local.

2. Abrir la ventana inicial de RapidSVN

3. Seleccionar el menú Repository y la opcion Checkout.



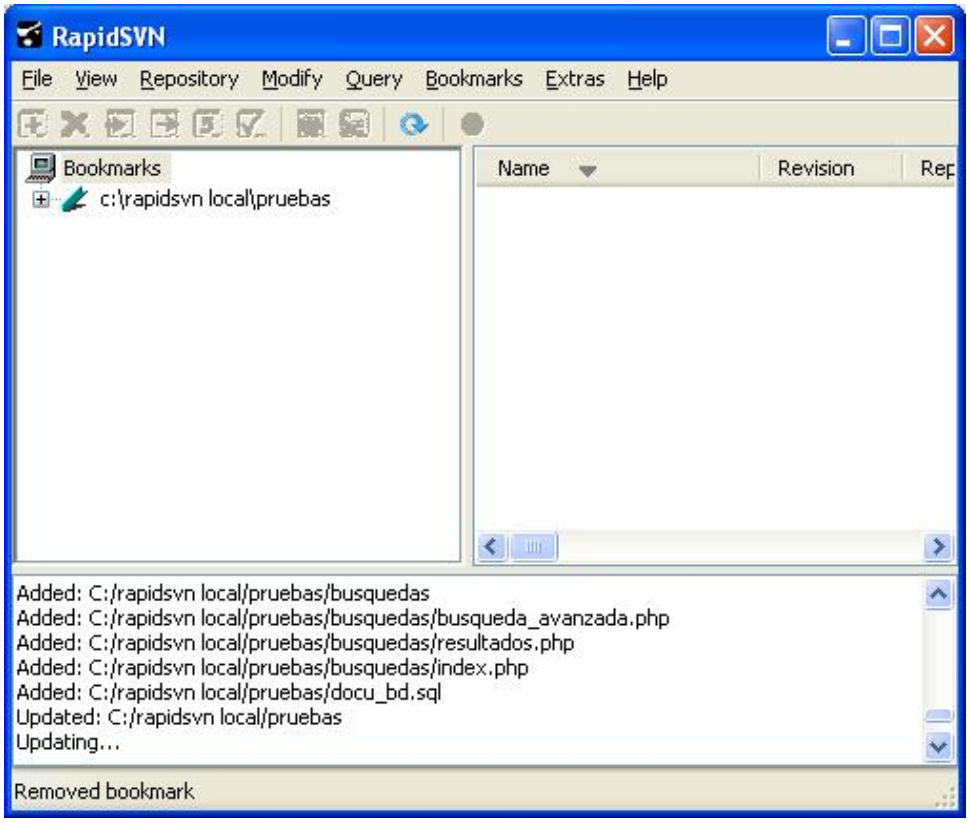
Al realizar esta operación aparece la siguiente ventana:



4. En la opción URL se introduce la dirección dada por el administrador. Por defecto esta URL se compone de la dirección de Cormoran "/" el nombre del repositorio central, este será confirmado por el administrador en el momento de la creación del repositorio.

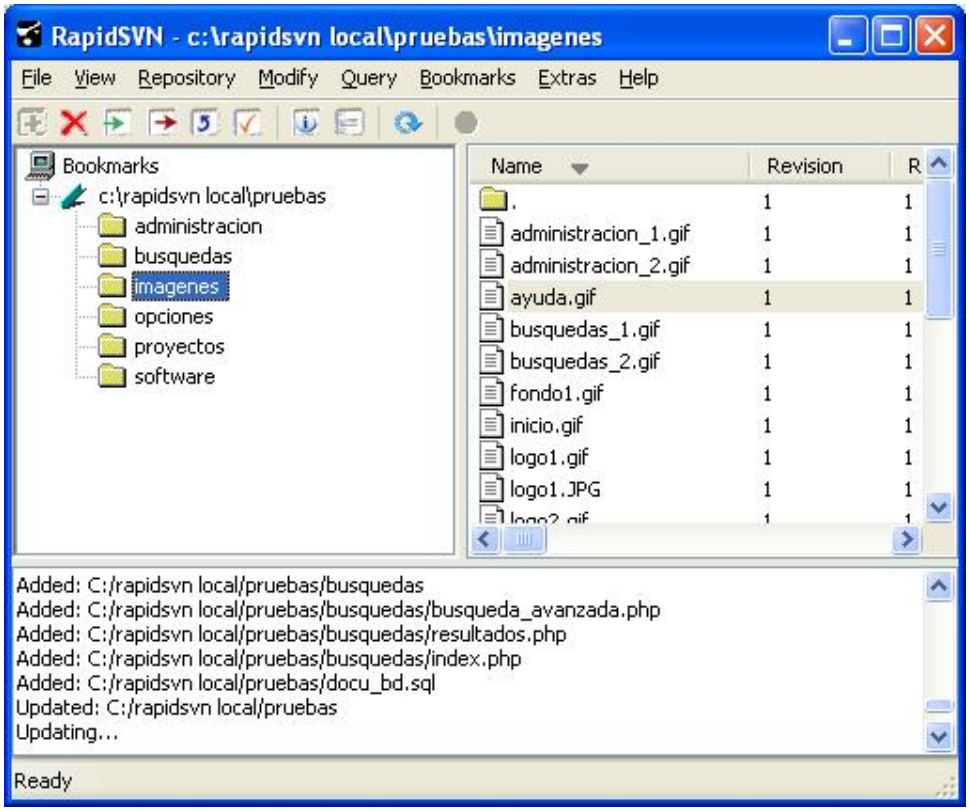
5. En la opción Destination Directory se busca con [...] la carpeta que se creo en el paso 1. Si se creo la carpeta pruebas esta es la carpeta que se debe escoger. Los demás parámetros se dejan por defecto a menos que se desee obtener una copia local con cambios antiguos.

6. Al oprimir el botón OK empiezan a descargarse los archivos desde el repositorio central. Estos archivos empezaran a aparecer en la parte inferior de la pantalla, en el panel de actividad.



7. Cuando los archivos son descargados a la copia local, se vera que aparece una nueva marca la lista de Bookmark.

8. Al hacer clic en el símbolo + que aparece al lado izquierdo de la marca, se abrirá la lista de todas las carpetas y archivos que han sido descargados. Desde aquí se puede navegar en los archivos como desee.



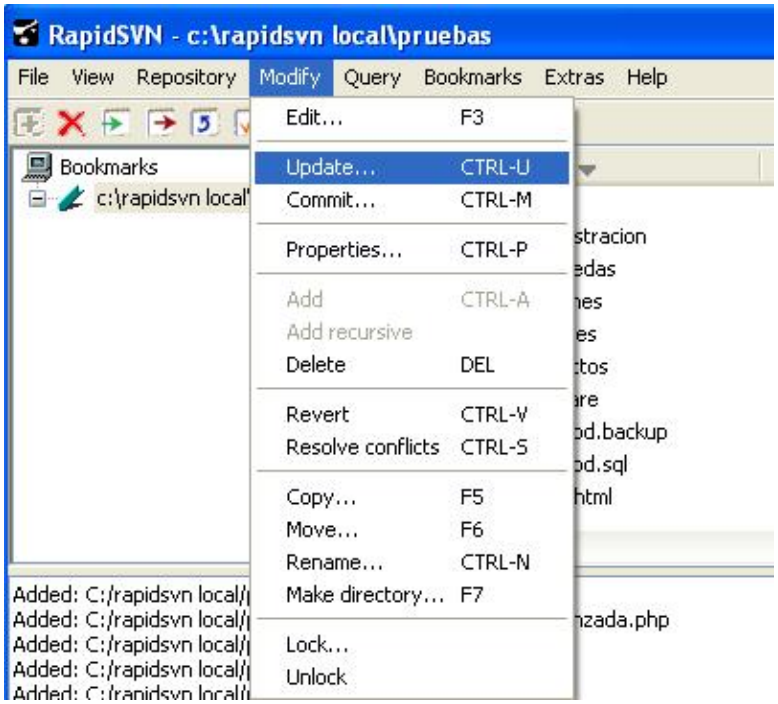
ACTUALIZAR LA COPIA LOCAL CON LOS CAMBIOS DE OTROS

Si se esta trabajando en grupo, normalmente es necesario incorporar los cambios hechos por otras personas a la copia local. El proceso de obtener estos cambios desde el servidor que contiene el repositorio central, se llama actualizar.

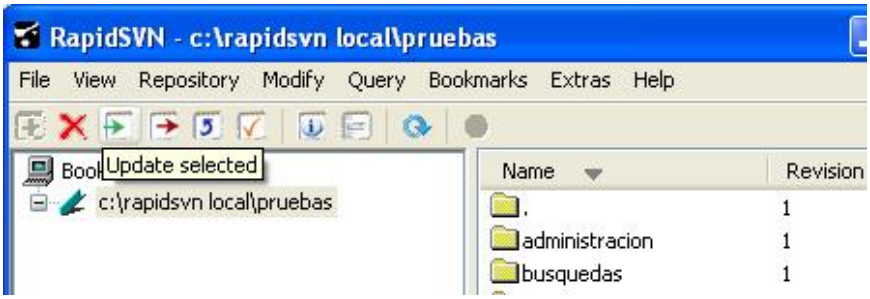
1. En la ventana principal de RapidSVN es posible realizar la actualización de varias formas. Pero primero se debe seleccionar el archivo sobre el cual se quiere realizar la actualización. Aunque se recomienda que sea el archivo sea la misma marca.

2. Es posible realizar la operación de tres formas y todas tienen el mismo resultado

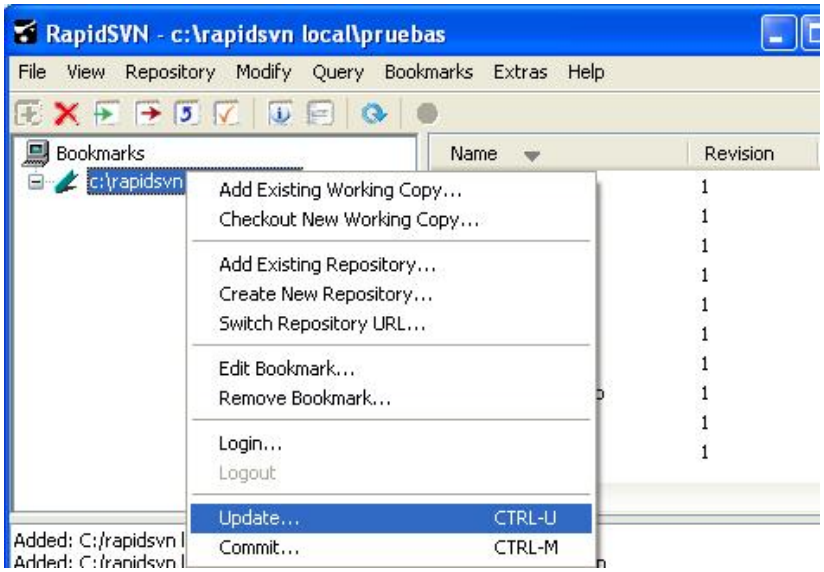
- Desde el menú principal



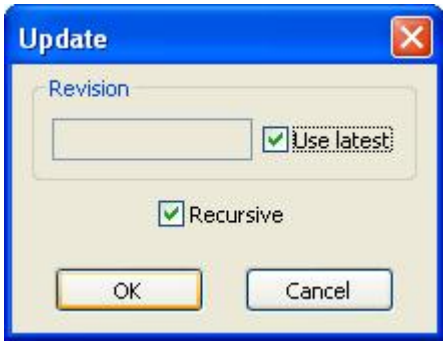
- Desde el menú de Iconos



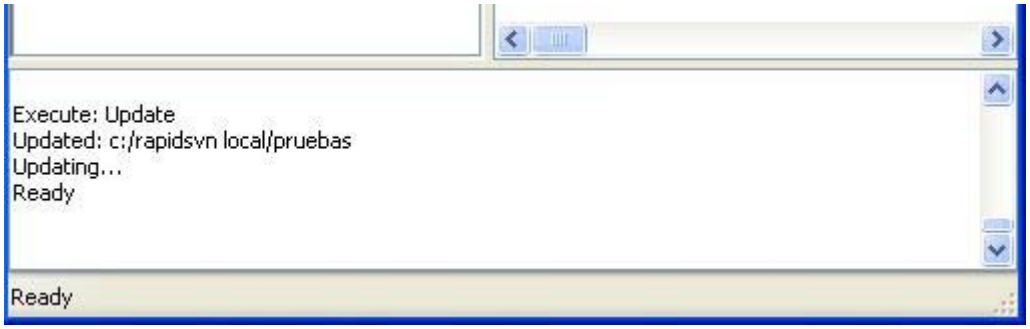
- Con el menú contextual (Clic Derecho)



Estas opciones abren una nueva ventana donde se pregunta si se quiere actualizar a la última revisión (opción por defecto) o alguna de las anteriores.



3. Seleccionar una de las opciones y hacer clic en OK. Después de esto aparecen los archivos que están siendo actualizados en el panel de actividades. Cuando se ha completado la operación aparece el texto Ready. Todos los cambios de otros han sido mezclados en la copia local.



CONFIRMAR CAMBIOS

Cuando se han realizado satisfactoriamente los cambios necesarios en la copia de trabajo local, es necesario reenviar la información de vuelta al repositorio central. Esta operación se llama confirmar o commit.

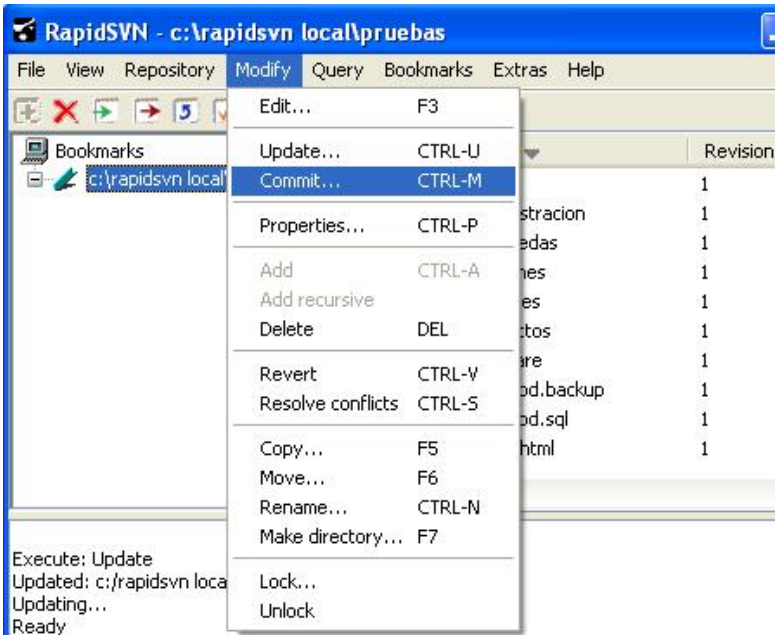
Cuando un archivo ha sido modificado en la ventana de SubVersion el estado cambia a modified.

1. Realizar actualización completa (pasos en el capitulo anterior). Esta operación es necesaria para verificar si existen conflictos.

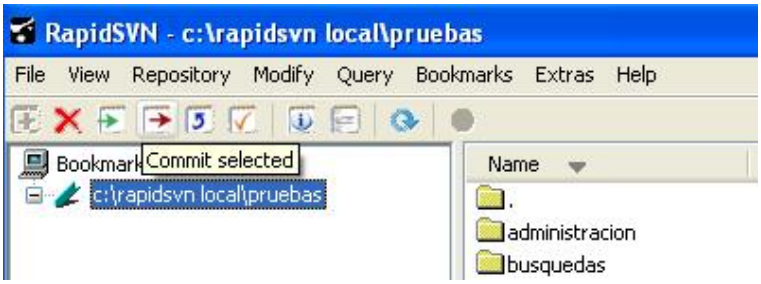
2. De ser necesario realizar la resolución de conflictos, aunque esto generalmente no sucede.

3. Para hacer la confirmación también existen tres formas:

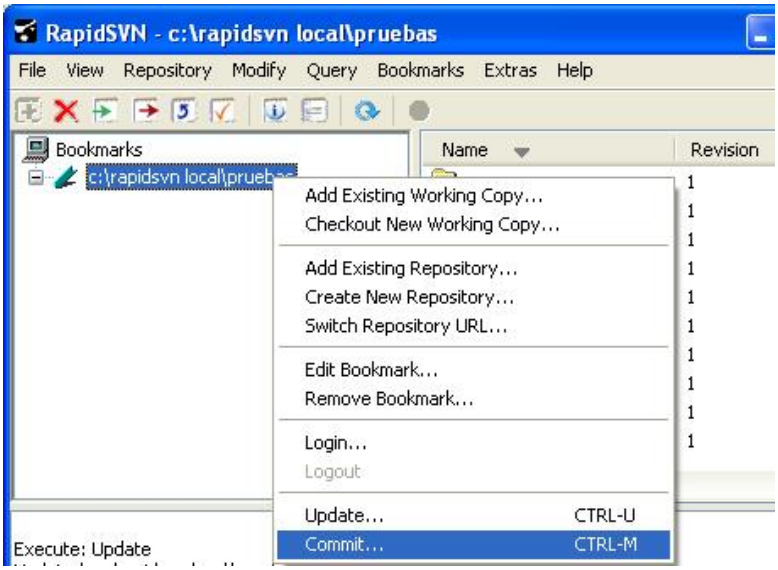
- Desde el menú principal



- Desde el menú de iconos



- Desde el menú contextual (Clic derecho)



Con esta operación la confirmación de cambios se abre un dialogo con los cambios realizados a los archivos.

4. Se realiza la autenticación de usuarios.

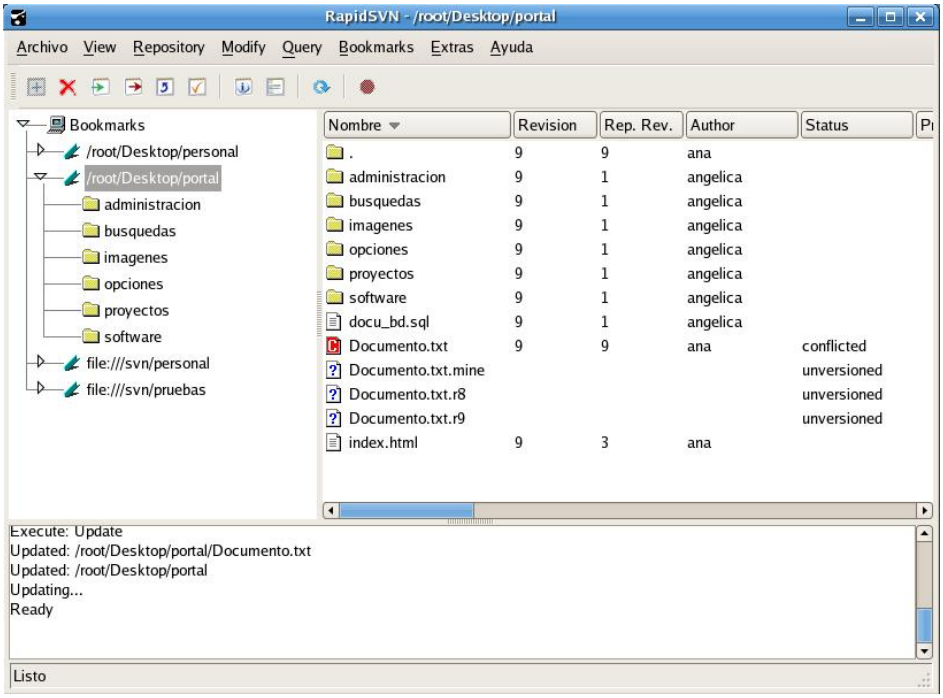
5. Se deja un mensaje con la descripción de los cambios realizados desde la última actualización para tener el historial.

RESOLUCIÓN DE CONFLICTOS

Ocasionalmente se encontrara un conflicto entre los archivos de alguien más y los propios. Esto ocurre cuando dos personas modifican la misma parte de un documento. SubVersion no puede resolver por si solo el conflicto y lo deja para la persona que realiza la actualización. Estos conflictos pueden ser evitados si antes de trabajar en un archivo se hace una actualización, y justo después de terminar se confirman los cambios.

Un conflicto se indica en el panel de actividades, y hasta que el conflicto no sea resuelto no es posible realizar la confirmación del archivo.

El conflicto se muestra con una C como icono del documento

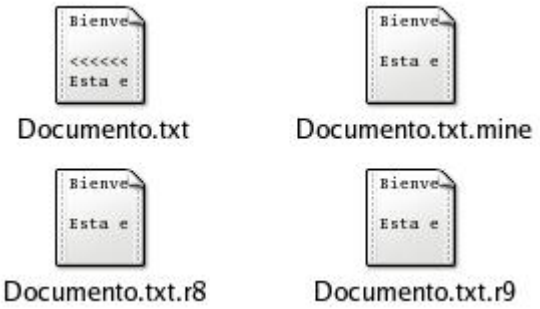


1. Tres nuevos archivos son creados con cada conflicto, estos tienen el mismo nombre del original pero con otras extensiones agregadas.

   Archivo.mine
   Archivo.r*
   Archivo.r*

   Los * son los números de revisión en los que se presenta el conflicto.

   Un ejemplo de las extensiones son las siguientes:



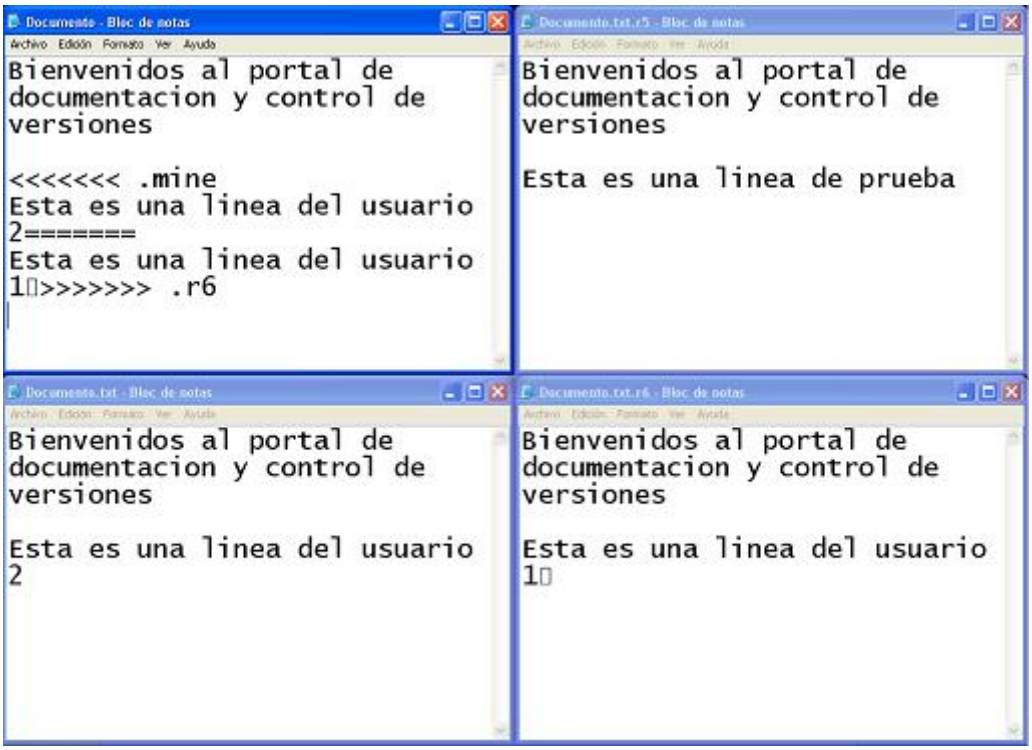
2. Cuando se abre el archivo original se encuentra una estructura parecida a la siguiente:

   <<<<<<<<<<nombre_del_archivo

Cambios locales
Código mezclado del repositorio
>>>>>>>>> Revisión

El archivo .mine es el archivo editado por el usuario que tiene el conflicto.

Los archivos .r* contienen el código de la revisión anterior y la actual. Es decir la revisión de la cual los dos usuarios actualizaron originalmente, y la revisión que contiene los cambios del otro usuario y que produjo el conflicto.



3. Es necesario decidir que texto o código debe quedar y hacer los respectivos cambios, esto generalmente se hace en consenso entre los autores de los cambios.

4. Cuando la edición del archivo se completa, guardar el archivo y usar el comando Resolve del menú commit o del menú de iconos. Esto enviara los cambios nuevamente al repositorio.

5. Agregar un comentario sobre la operación realizada para mantener el historial de mensajes.

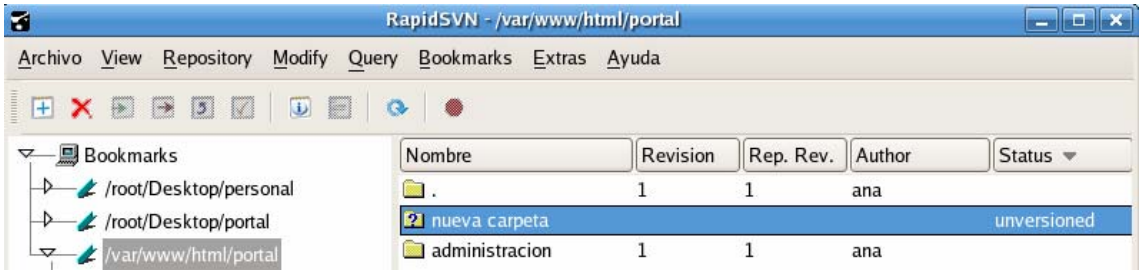
Con esto el archivo debe retornar a su estado normal.

AGREGAR NUEVAS CARPETAS Y ARCHIVOS

Todos los nuevos archivos y directorios necesitan ser añadidos al repositorio central. Sin embargo, antes de que esto pase todos estos archivos deben ser agregados en la copia de trabajo local.

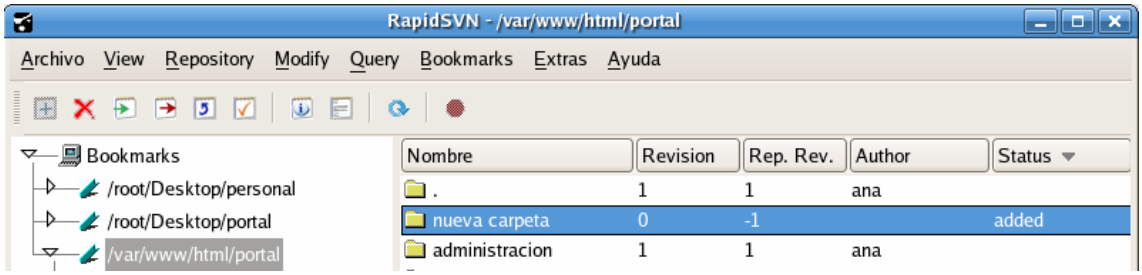
Para agregar una nueva carpeta que contiene uno o más archivos a la copia de trabajo local:

1. Ubique la carpeta en la copia local a través del árbol de RapidSVN. Ahora la carpeta debe tener un signo de pregunta en el icono.



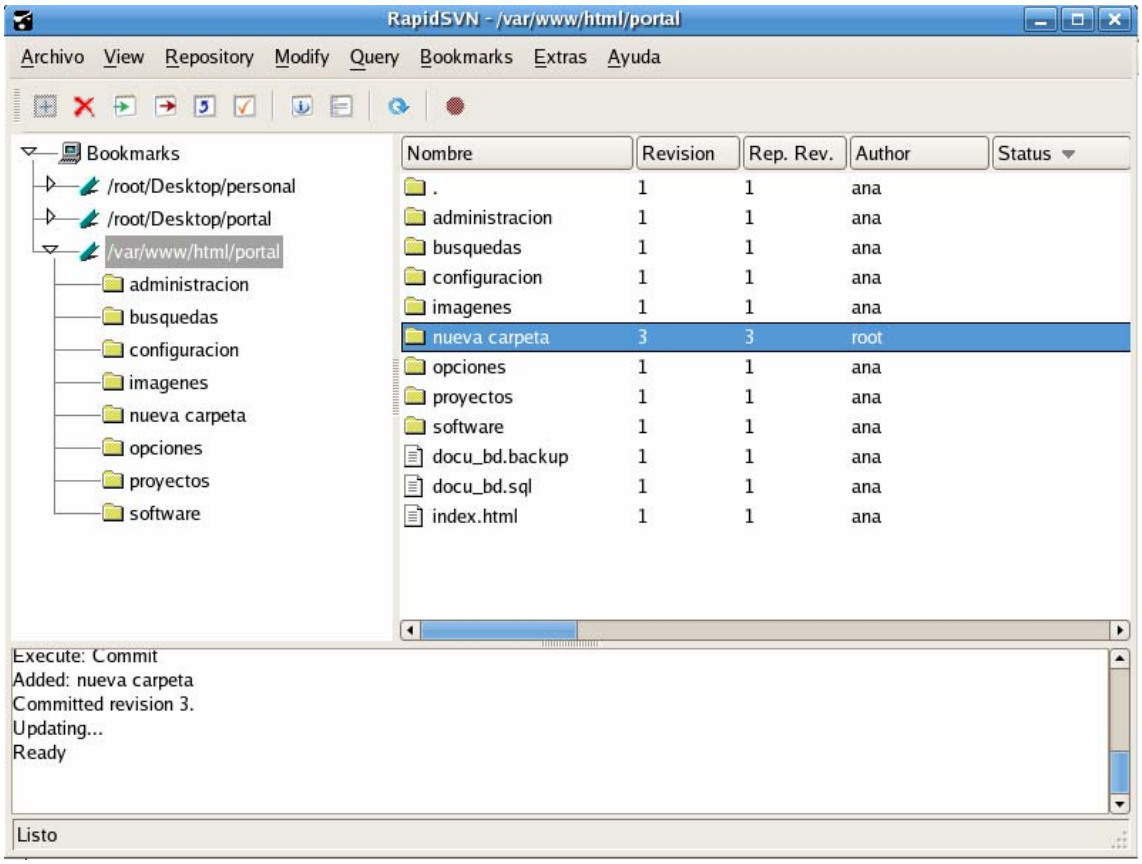
2. Seleccione la carpeta haciendo clic en el nombre. El icono que aparecía en gris en la parte izquierda (un +) ahora se hace visible.

3. Haciendo clic en este signo azul se añade la nueva carpeta a la copia local, y el icono vuelve a ser normal. Si esto no pasa puede refrescar la pantalla de RapidSVN usando el incono de la flecha azul redonda. En la parte derecha. El estado del archivo camia a added.



4. Cuando todos los archivos necesarios han sido agregados aparecerán con una “A” en el icono, se debe realizar una confirmación para que los cambios tengan efecto en el repositorio central.

5. Se debe realizar la confirmación, y dejar un mensaje de registro para el historial.



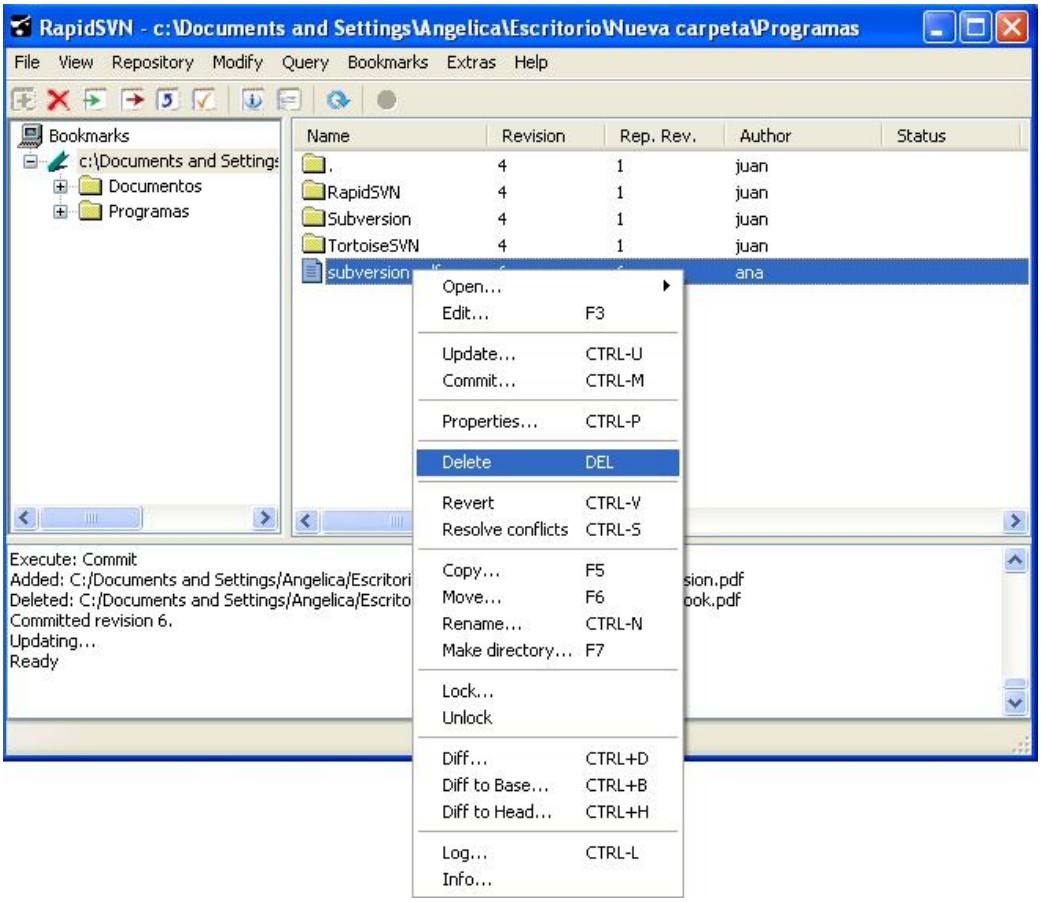
Al terminar la operación todos los archivos volverán al estado “Normal”.

BORRAR, MOVER Y RENOMBRAR ARCHIVOS Y CARPETAS.

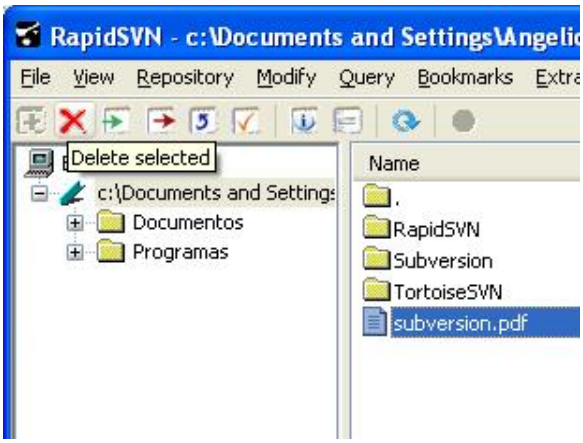
Borrar Archivos

1. Existen varias formas de borrar un archivo o carpeta.

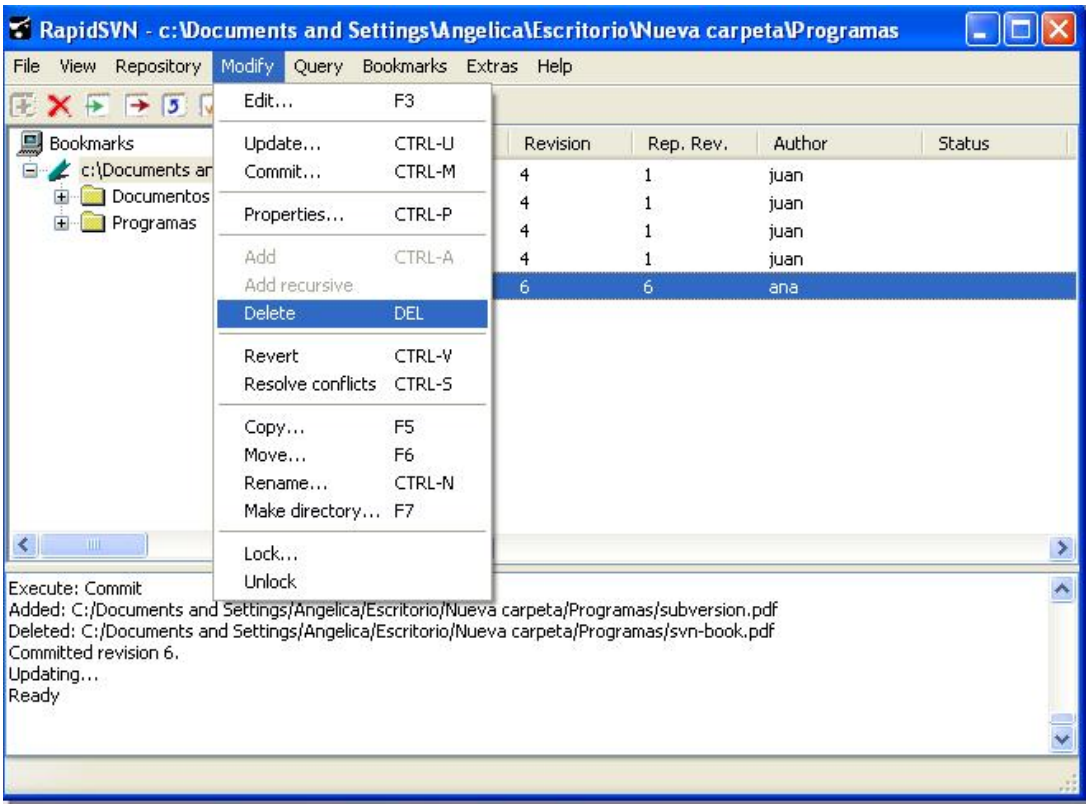
- Clic derecho en el nombre del archivo o carpeta y seleccionar la opción Delete del submenú.



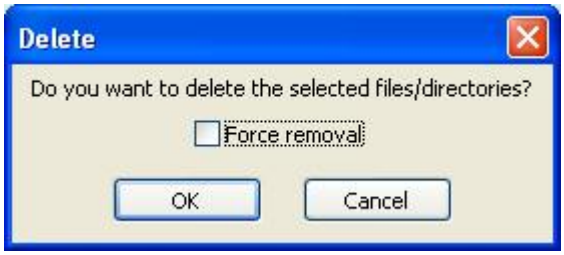
- Seleccionar el archivo y hacer clic en el icono con la X en el menú de iconos.



- Seleccionar el archivo y hacer clic en la opción Delete del Modify del menú principal.

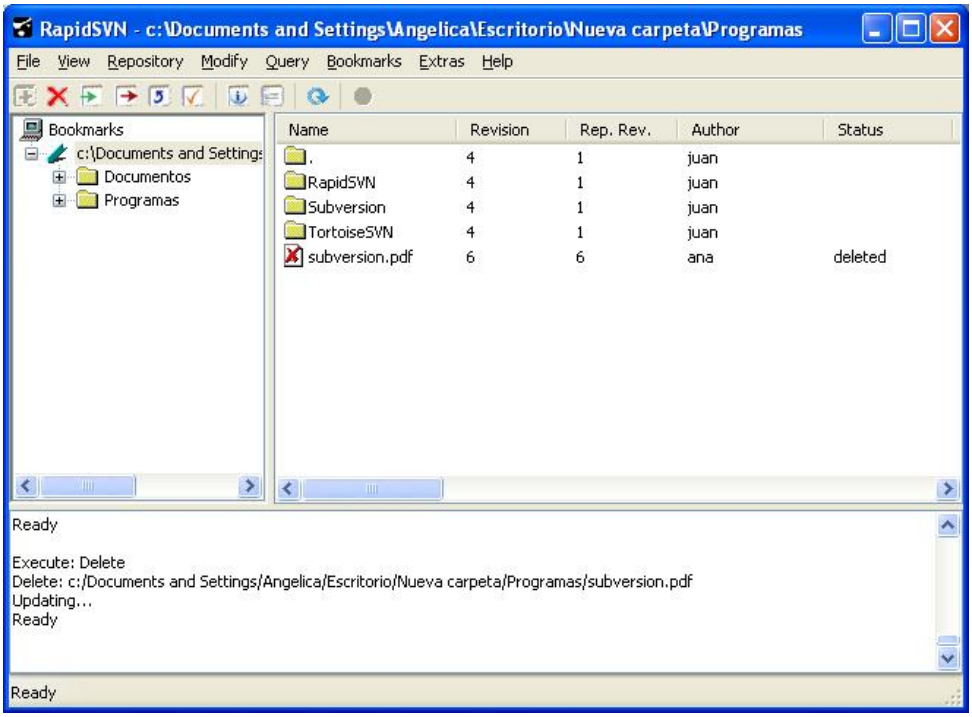


2. Cualquiera de las opciones anteriores abrirá un cuadro de dialogo en el que se pedirá que se confirme la acción.



3. Después aparecerá un cuadro para poner un comentario sobre la razón para borrar los archivos.

4. Los archivos seleccionados en este momento serán marcados con una X en el icono. En este punto es posible deshacer la acción haciendo clic en el icono Revert. Aparecerá una X de borrado sobre el icono de la carpeta.

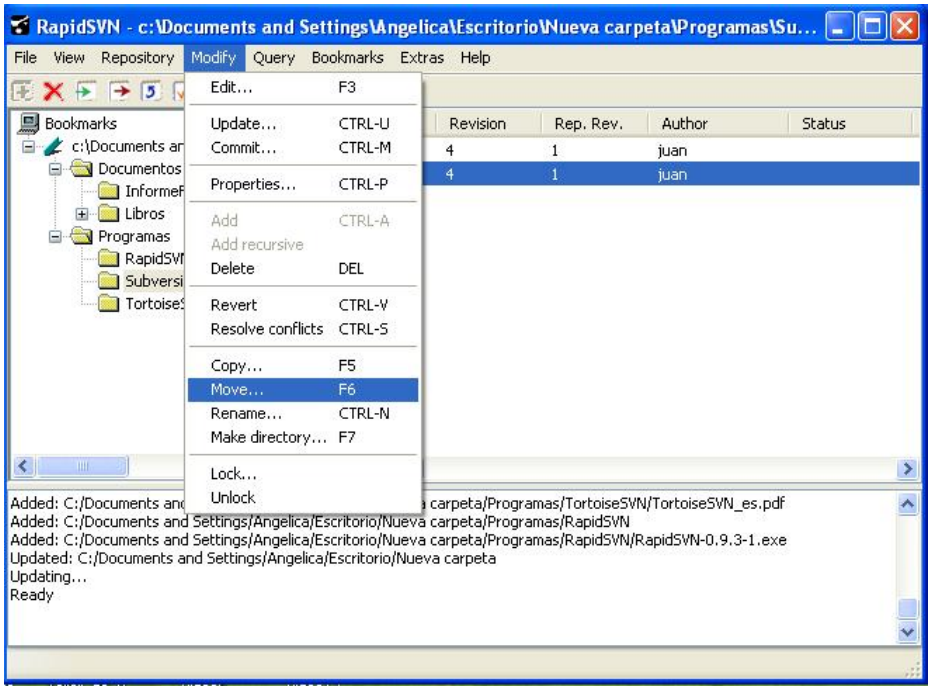


5. Para que los cambios sean realizados es necesario hacer una confirmación, con todos los pasos que esto indica.

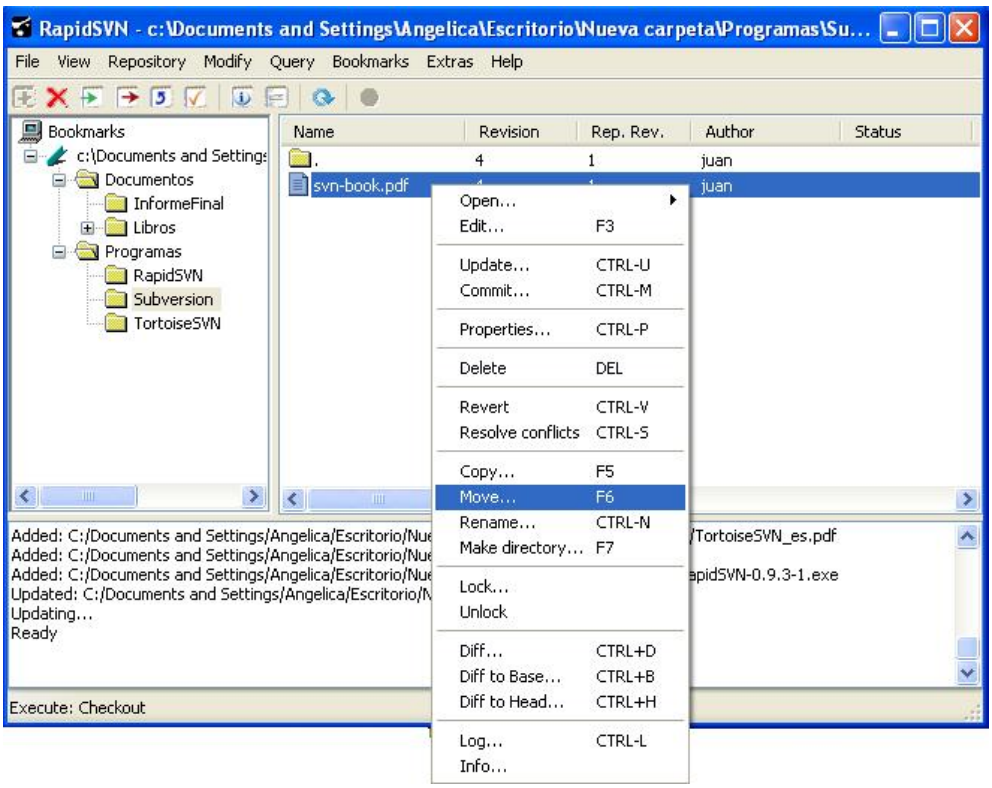
Mover archivos

1. Existen varias formas de mover los archivos

- En el menú principal Modify se selecciona la opción Move.



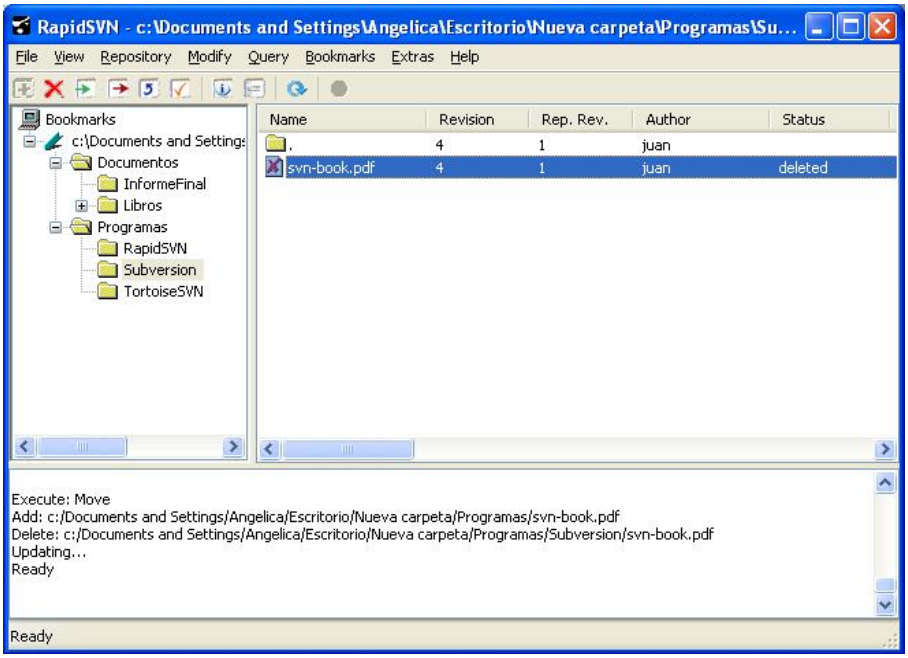
- Hacer Clic derecho en el archivo que se quiere mover.

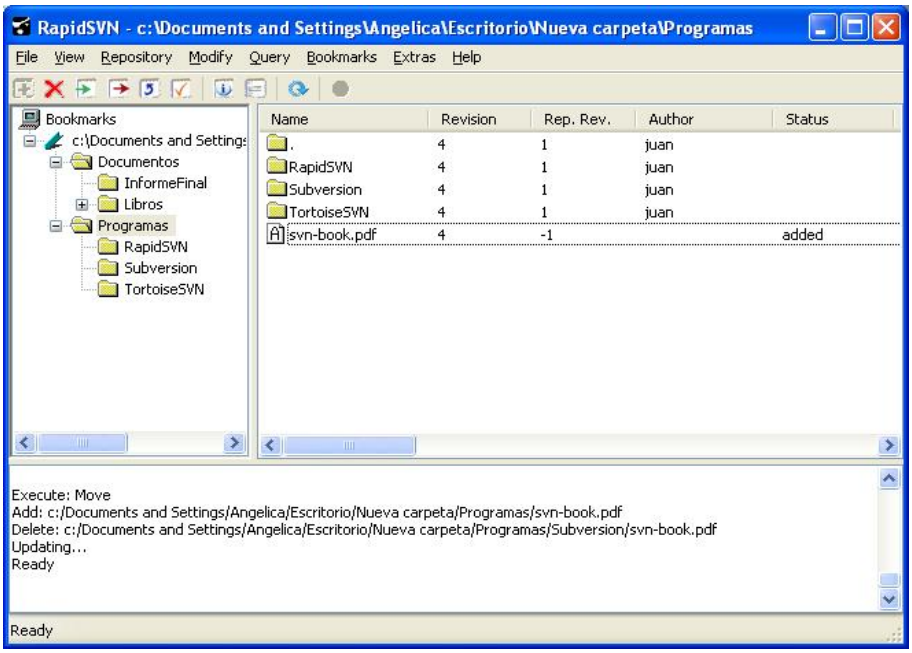


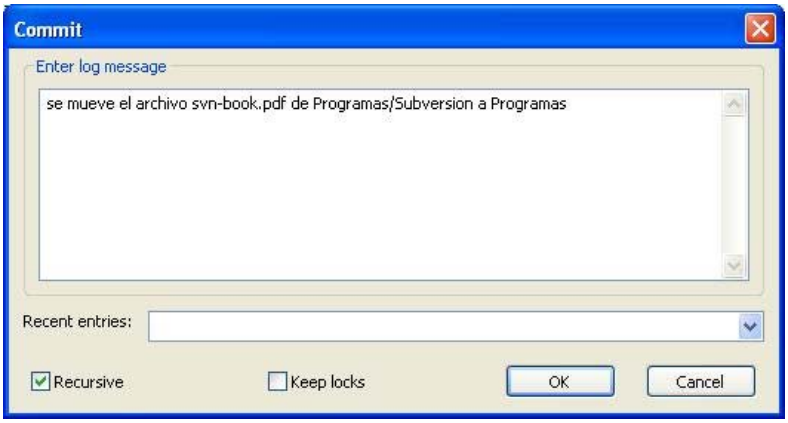
2. Se abrirá un cuadro de dialogo en la que se preguntara el destino del archivo que quiere mover.

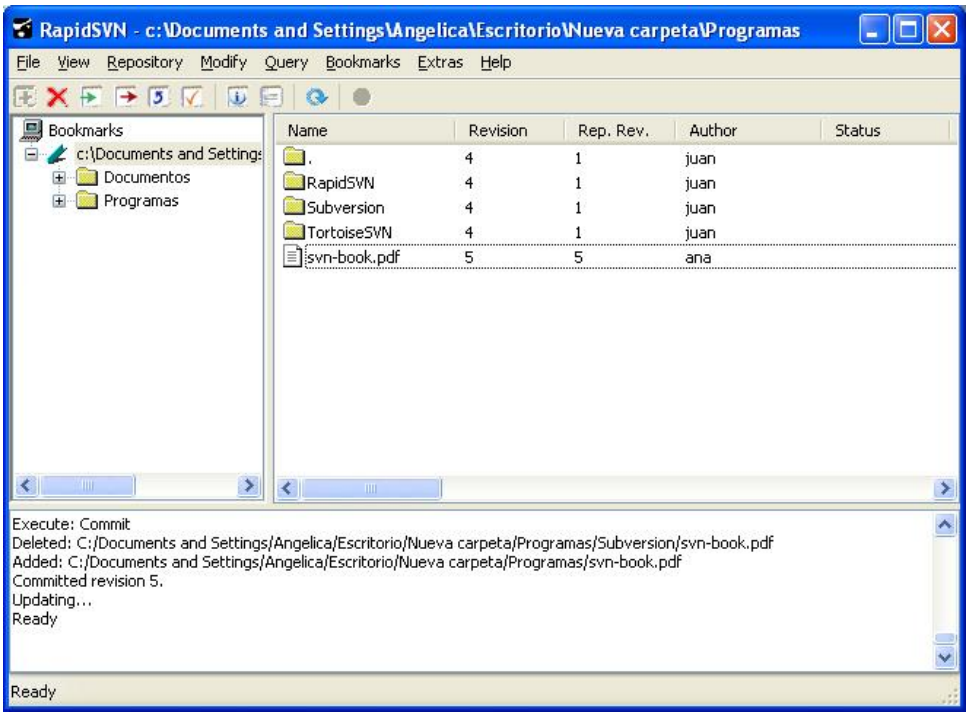
Al hacer esto el archivo o carpeta cambiara su ubicación. En la carpeta original el archivo aun se encontrara pero con una X en su icono. Y el archivo en la nueva ubicación también estará, pero con una “A” en su icono.

3. Aun se necesita que los cambios se realicen en el repositorio central. Para esto se realiza una confirmación de los dos cambios.


RapidSVN - c:\Documents and Settings\Angelica\Escritorio\Nueva carpeta\Programas\Su...
File View Repository Modify Query Bookmarks Extras Help
Bookmarks
c:\Documents and Settings
Documentos
InformeFinal
Libros
Programas
RapidSVN
Subversion
TortoiseSVN
Name Revision Rep. Rev. Author Status
. 4 1 juan
svn-book.pdf 4 1 juan deleted
Execute: Move
Add: c:/Documents and Settings/Angelica/Escritorio/Nueva carpeta/Programas/svn-book.pdf
Delete: c:/Documents and Settings/Angelica/Escritorio/Nueva carpeta/Programas/Subversion/svn-book.pdf
Updating...
Ready
Ready


RapidSVN - c:\Documents and Settings\Angelica\Escritorio\Nueva carpeta\Programas
File View Repository Modify Query Bookmarks Extras Help
Bookmarks
c:\Documents and Settings
Documentos
InformeFinal
Libros
Programas
RapidSVN
Subversion
TortoiseSVN
Name Revision Rep. Rev. Author Status
. 4 1 juan
RapidSVN 4 1 juan
Subversion 4 1 juan
TortoiseSVN 4 1 juan
svn-book.pdf 4 -1 added
Execute: Move
Add: c:/Documents and Settings/Angelica/Escritorio/Nueva carpeta/Programas/svn-book.pdf
Delete: c:/Documents and Settings/Angelica/Escritorio/Nueva carpeta/Programas/Subversion/svn-book.pdf
Updating...
Ready
Ready

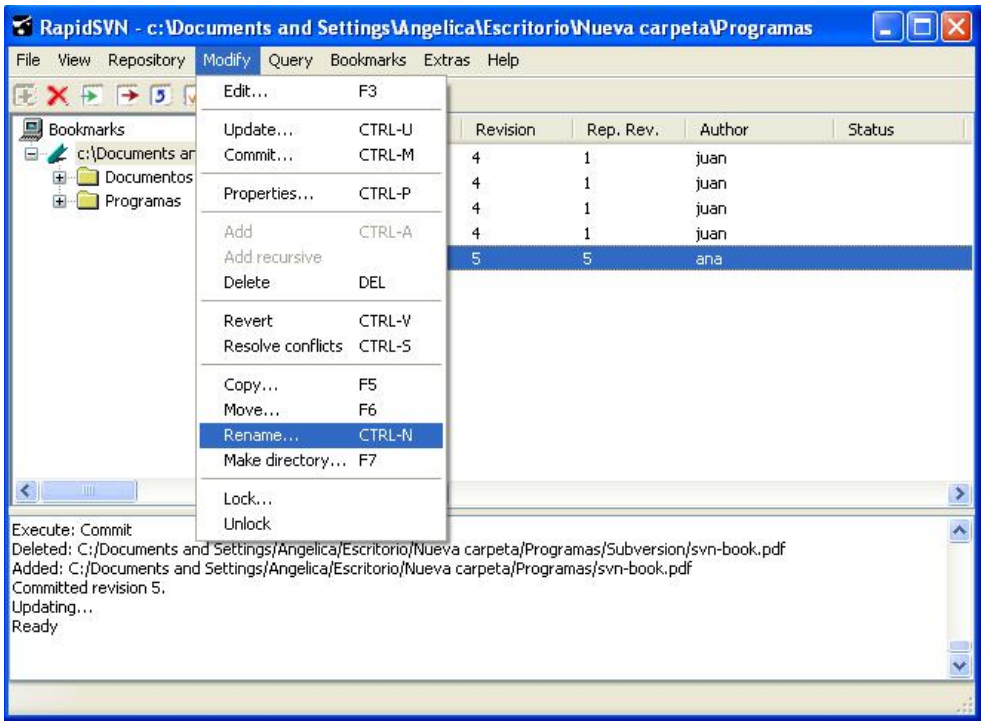

Commit
Enter log message
se mueve el archivo svn-book.pdf de Programas/Subversion a Programas
Recent entries:
Recursive
Keep locks
OK
Cancel



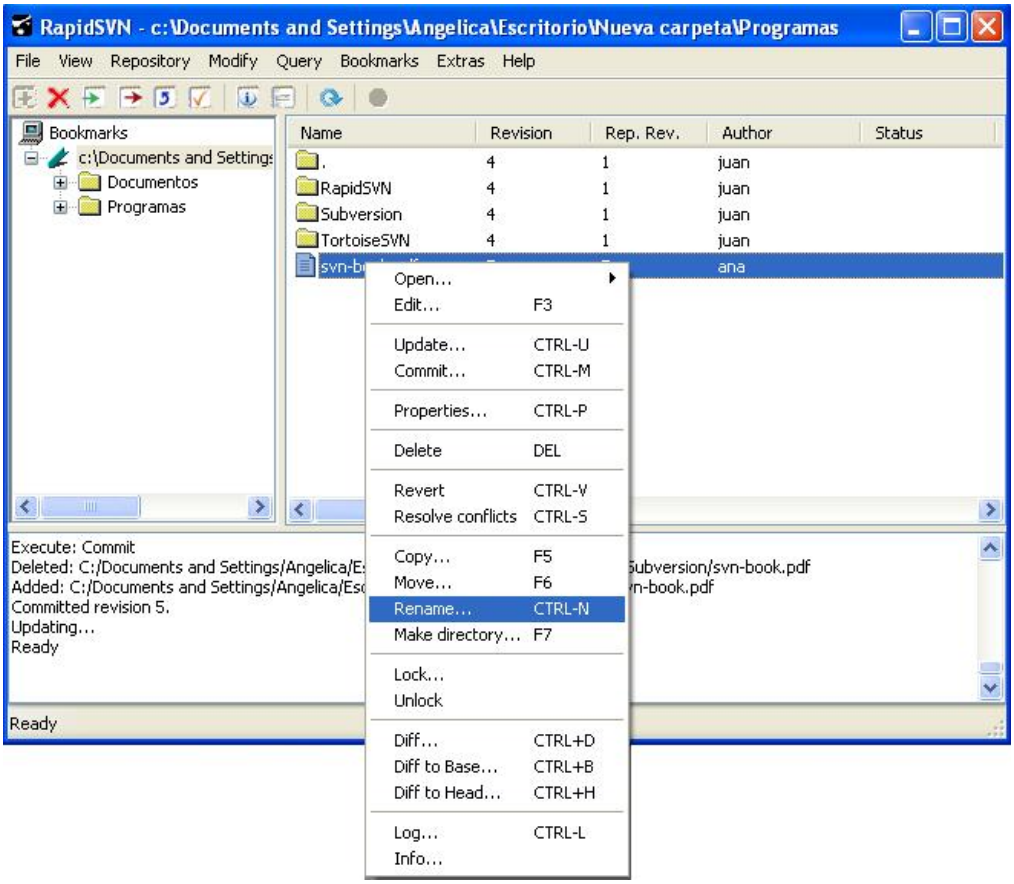
Para renombrar un archivo el procedimiento es el mismo.

1. Es posible renombrar los archivos por varios caminos

- En el menú Modify se selecciona la opción Rename.

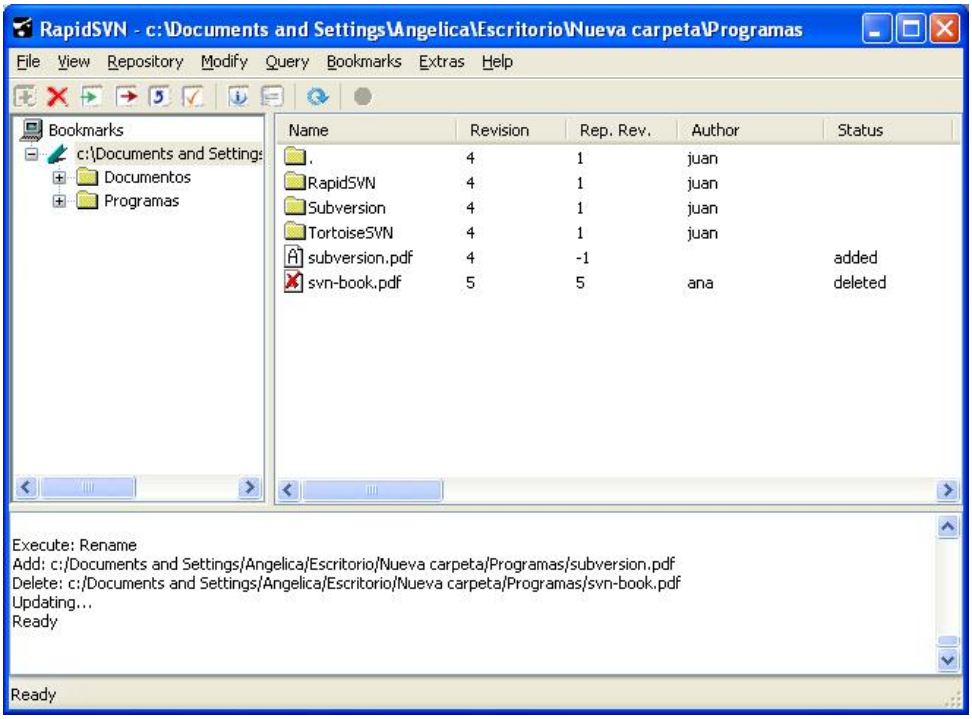


- Desde la opción Rename dando clic derecho en el nombre del archivo.

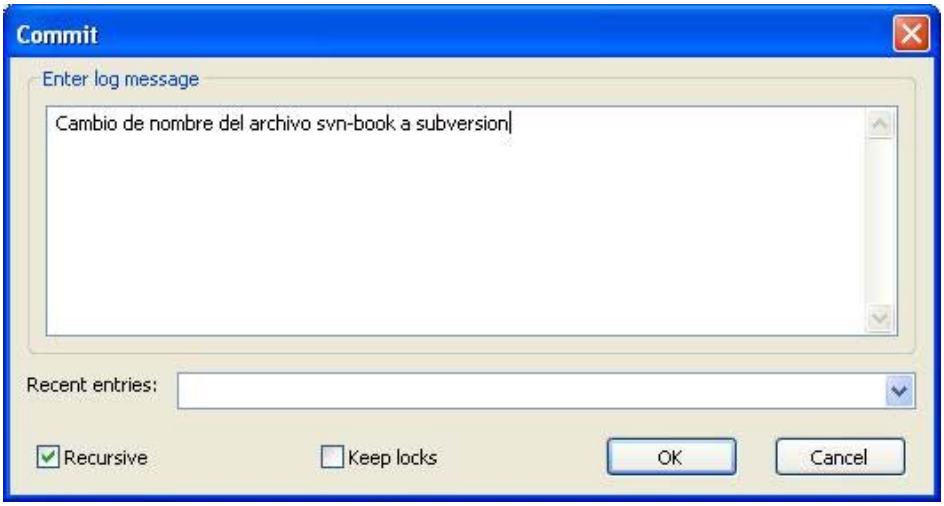


2. Se solicita el nuevo nombre del archivo en una ventana nueva.

3. Una vez aceptado, aparece un archivo con el nombre anterior con un icono de borrado y uno con el nuevo nombre con un icono con una “A” y el estado como añadido.



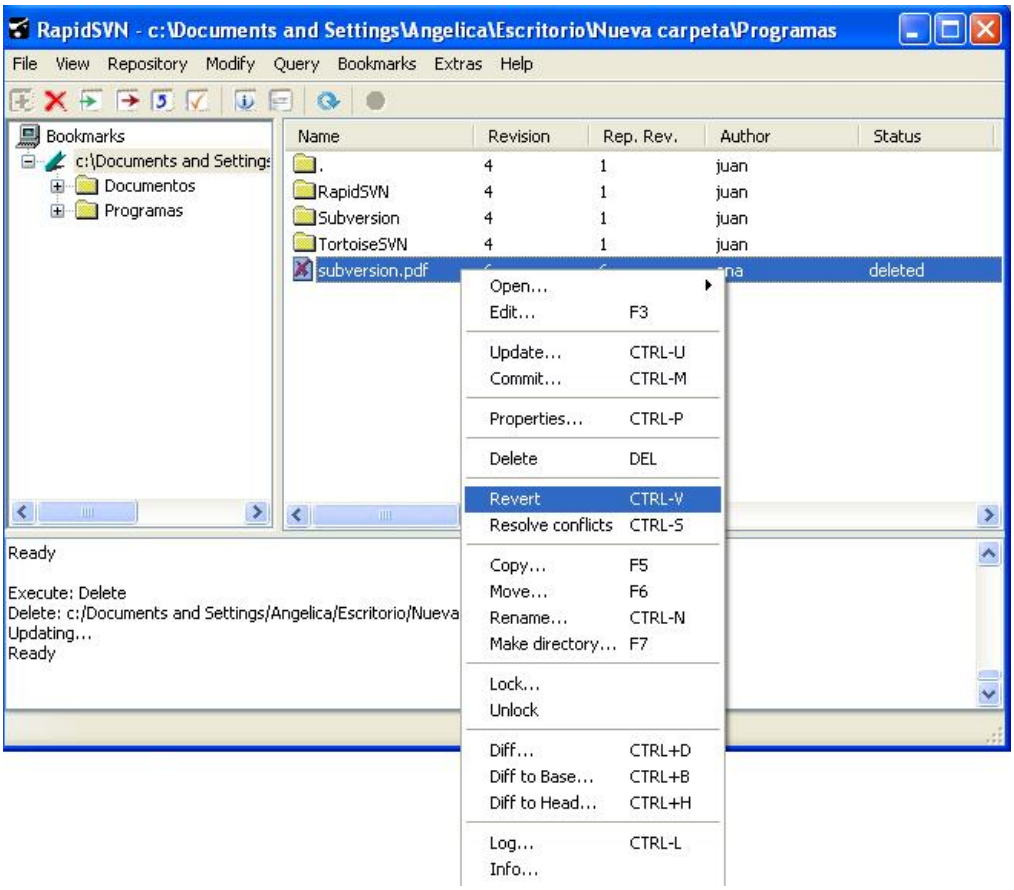
4. Ahora es necesario hacer la sincronización con el repositorio central, para que la operación quede completa. El siguiente es el registro dejado cuando se realiza el commit.



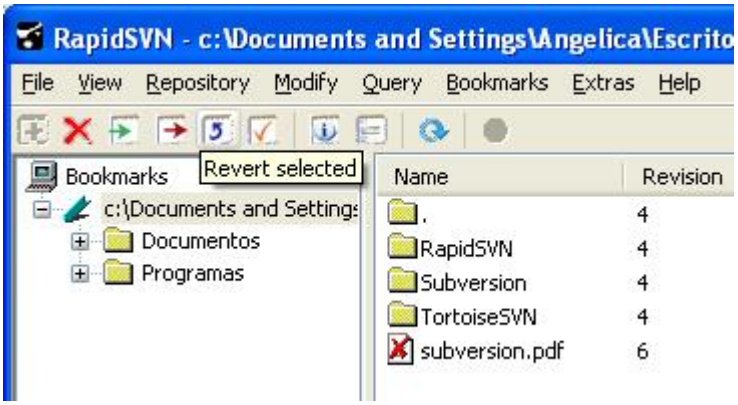
CANCELAR LOS CAMBIOS EN LOS ARCHIVOS

1. Si usted quiere cancelar todos sus cambios desde la ultima actualización debe seleccionar el archivo y

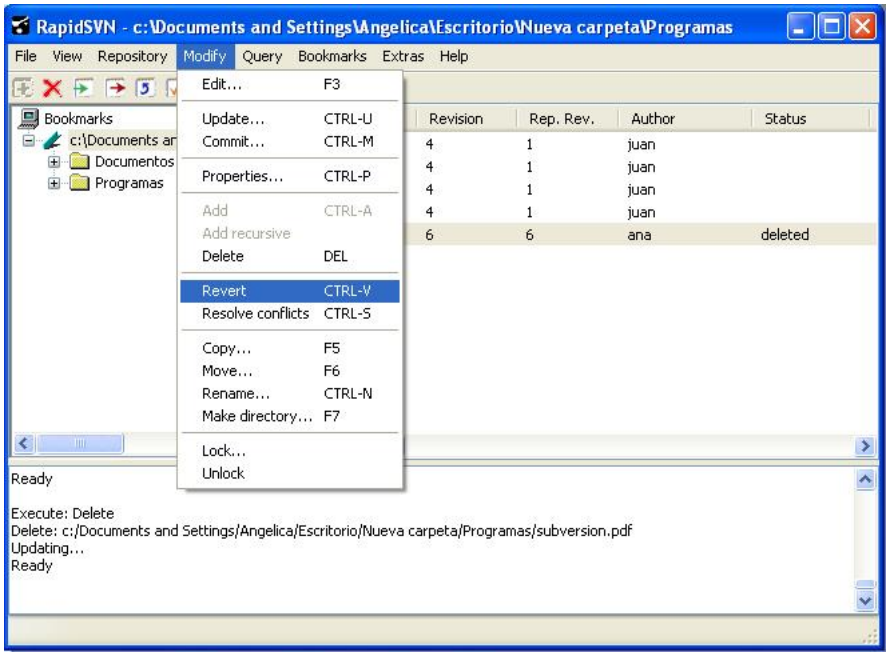
- Dar clic derecho y seleccionar la opción Revert.



- Dar clic en el icono Revert del menú de iconos.

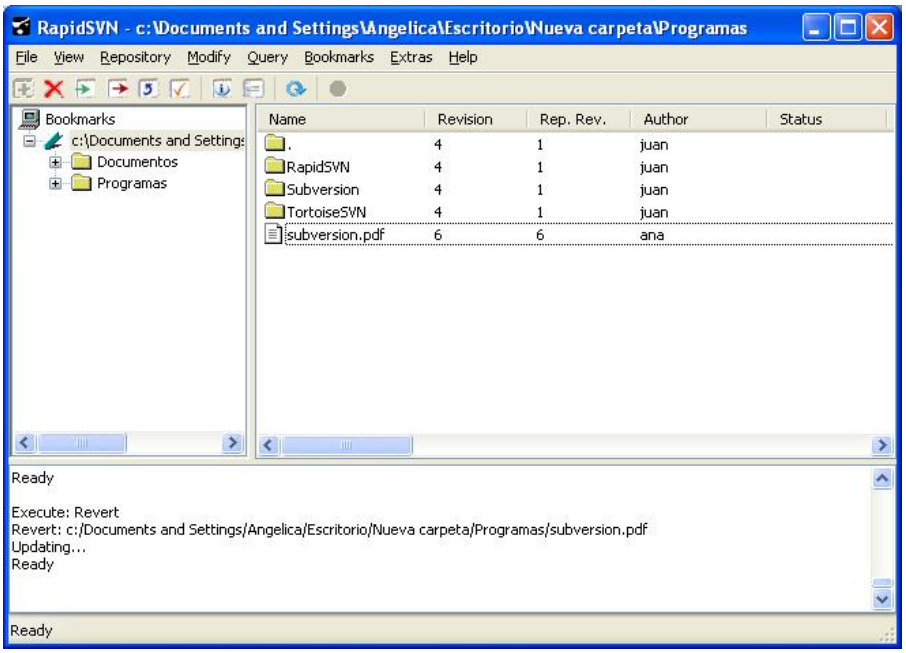


- Seleccionar la opción Revert del menú Modify.



2. Es necesario confirmar la operación.

Hay que tener en cuenta que los cambios se perderán y el archivo volverá totalmente al estado anterior.

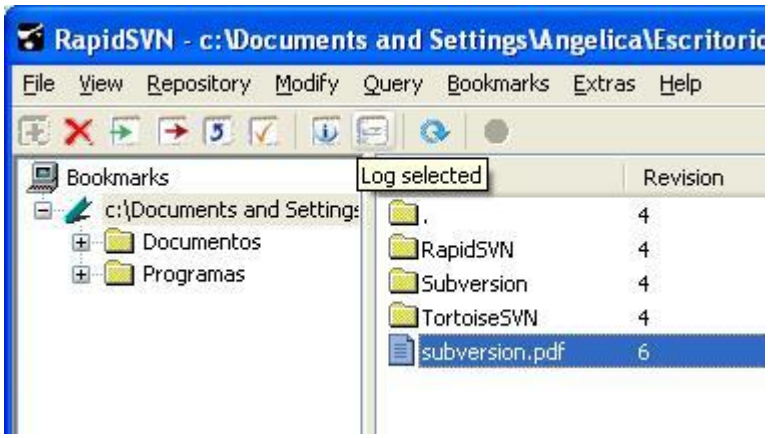


OBTENER INFORMACIÓN ACERCA DE ARCHIVOS Y DIRECTORIOS

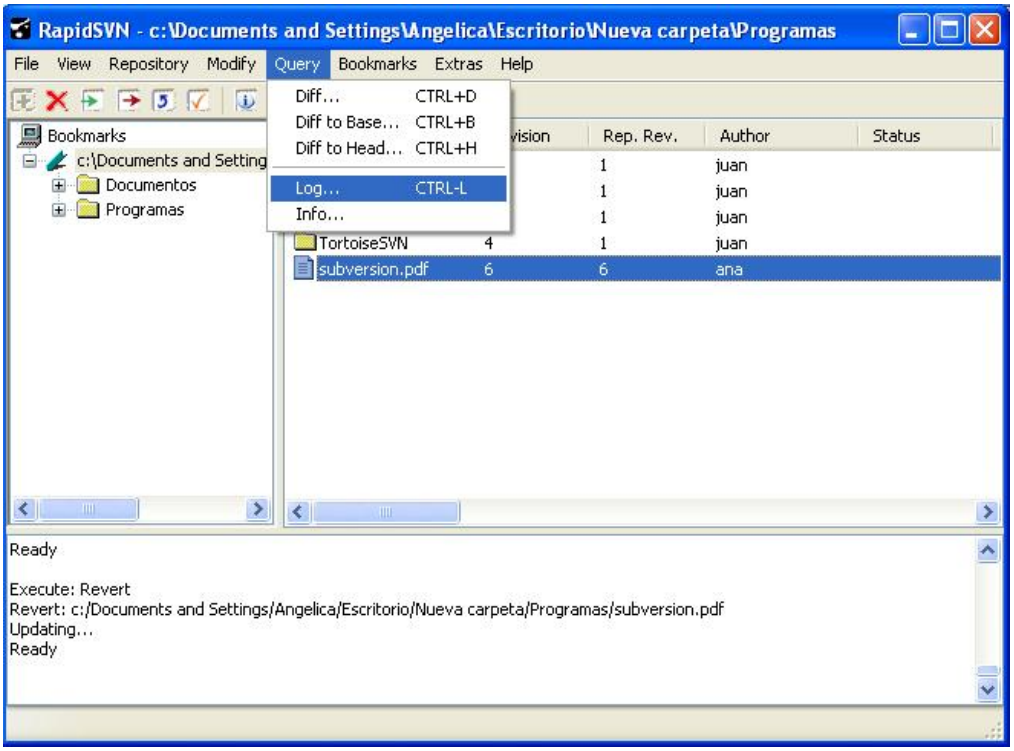
Hay ocasiones en las cuales es necesaria más información acerca de un archivo partícula. Esta información esta disponible en dos opciones: Log History e Information.

1. Log History

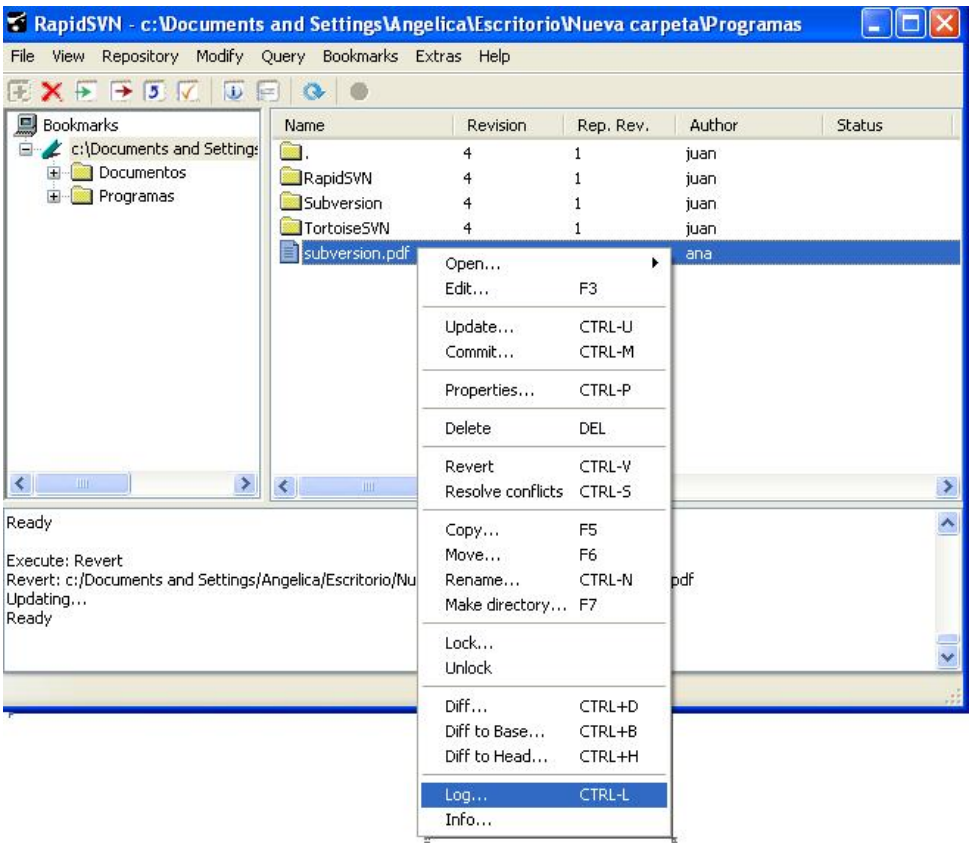
- En el menú de iconos



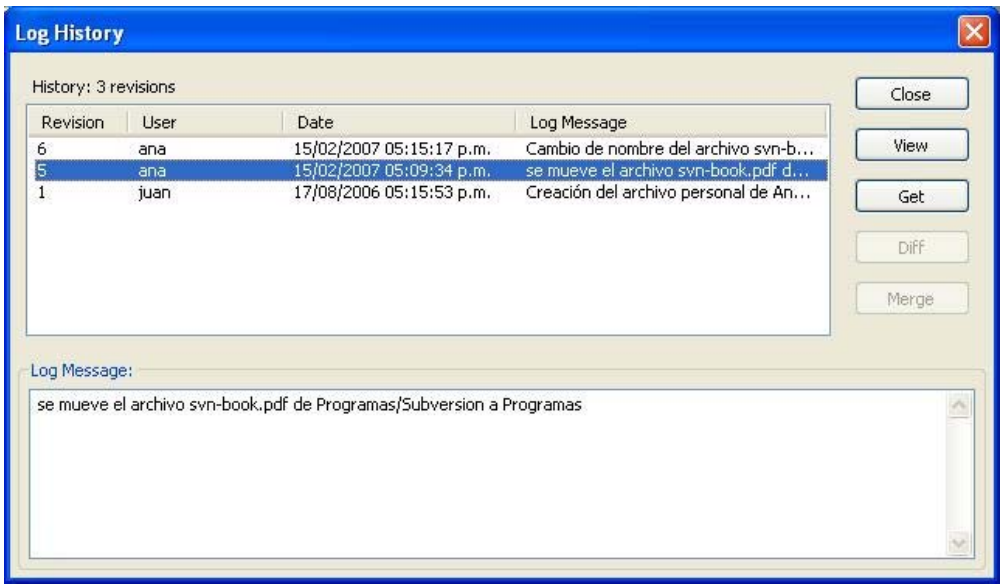
- En el menú Query



- En la opción Log del clic derecho.

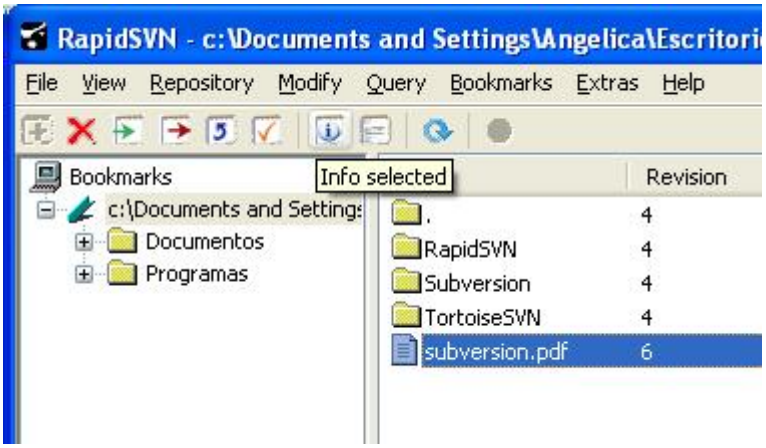


A continuación se abre una ventana como la siguiente con información dejada en las revisiones donde fue modificado dicho documento.

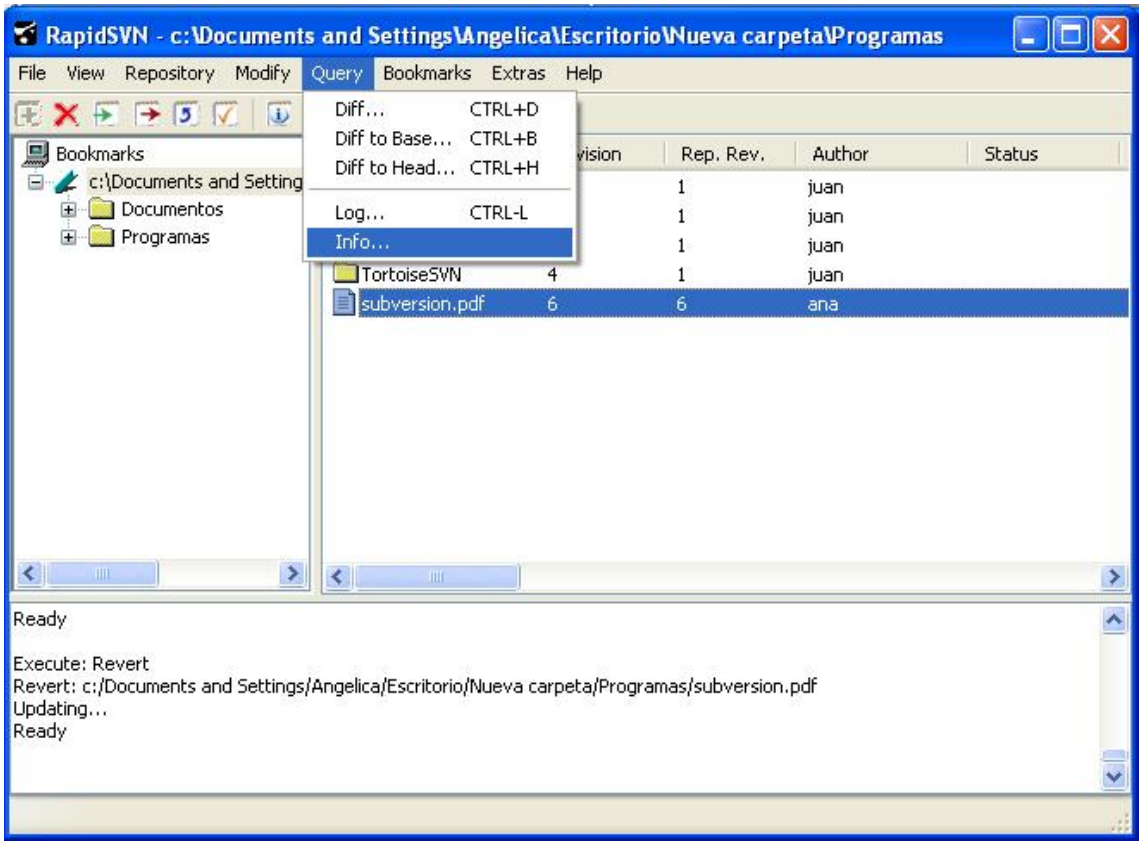


2. Información

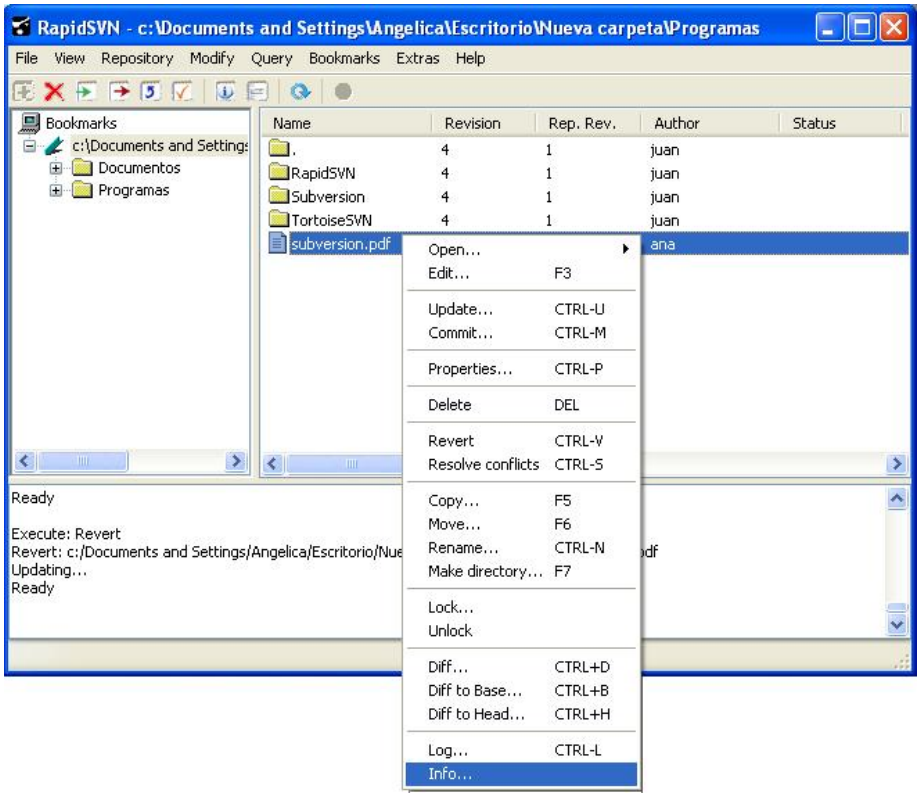
- En el menú de iconos



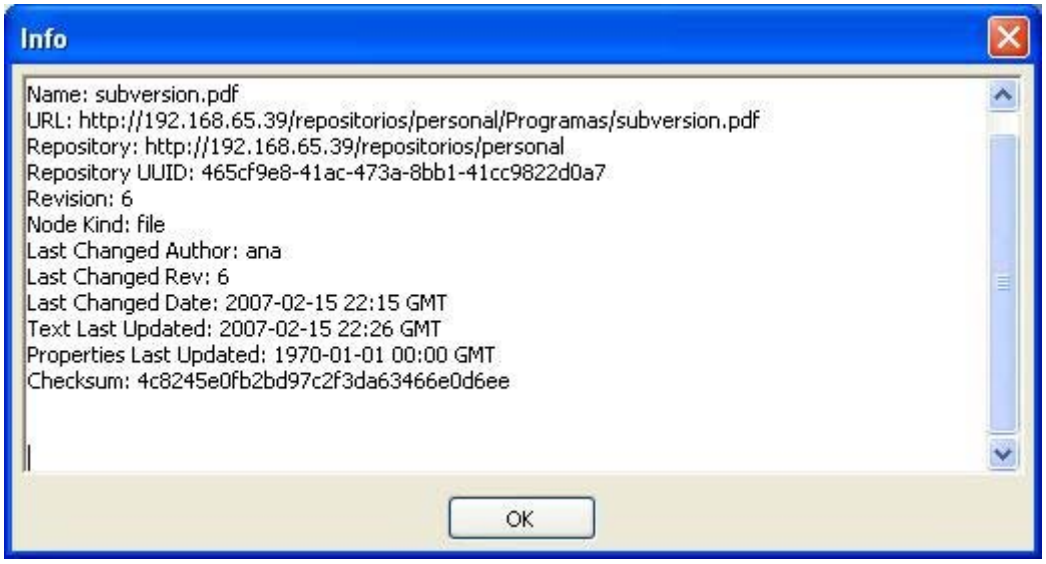
- En el menú Query



- La opción Info en el clic derecho



Cualquiera de las anteriores opciones abre una ventana parecida a la siguiente con la información relativa al archivo seleccionado.



SOLUCION A ERRORES FRECUENTES

1. Error:
Error while performing action: La copia de trabajo "/copia/de/trabajo" esta bloqueada.

Causa:
Existen varias causas, la más común es que alguno de los usuarios hubiera tenido un conflicto relacionado con alguno de los archivos en esta copia de trabajo.

Solución:
Ejecutar el comando Cleanup.

2. Error:
Error while performing action: El directorio '/copia/de/trabajo/.svn' con la información de administración de la cdT no está.

Causa:
La carpeta que contiene este archivo esta involucrada en la copia de trabajo pero la carpeta ".svn" del mismo no se encuentra, pudo haber sido borrada o movida.

Solución:
Copiar una carpeta ".svn" de otro lugar y realizar la operación 'delete' sobre los archivos que no están y 'add' sobre los existentes. Se perderá el Log de estos archivos.

3. Error:
Error while performing action: Falló el commit (detalles a continuación) El archive 'repositorio/archivo' ya existe.

Causa:
Otro usuario ya había añadido este archivo al mismo repositorio, no se puede efectuar la misma operación nuevamente.

Solución:
Borrar los archivos desde la ubicación del directorio en el sistema operativo y actualizar nuevamente la copia de trabajo.

Anexo B. MANUAL DE TORTOISESVN

TortoiseSVN es un cliente gratuito de código abierto para el sistema de control de versiones SubVersion. Esta desarrollado para integrarse con el shell de Windows, por lo tanto solo puede ser utilizado en los sistemas operativos Windows a partir de XP.

TortoiseSVN maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un repositorio central, prácticamente lo mismo que un servidor de ficheros ordinario, salvo que recuerda todos los cambios hechos en el tiempo. Esto permite recuperar versiones anteriores y examinar la historia de cuando y como cambiaron los datos.

La creación de repositorios y otras características solo se encuentran disponibles de forma local, por lo tanto en Windows. Para las acciones necesarias en el servidor se utiliza el portal desarrollado.

A continuación se describe su uso, así como el ciclo regular de trabajo.

INSTALACIÓN

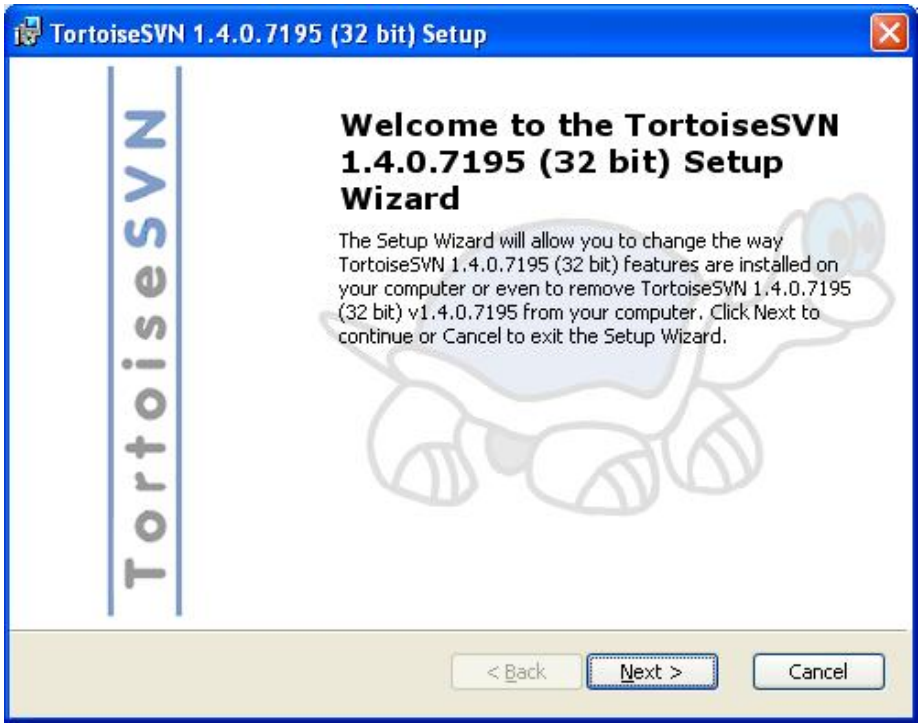
1. Para hacer la instalación del TortoiseSVN en español es necesario bajar el paquete de instalación, de la página web http://tortoisesvn.tigris.org

2. Además, el paquete para que sea posible trabajarlo en español es LanguagePack-1.4.0.7195-RC1-win32-es, que se puede bajar de la misma página, así como el manual de referencia completa.

3. Una vez se tienen los paquetes, al ejecutarlos, estos guían la instalación de una forma muy sencilla.

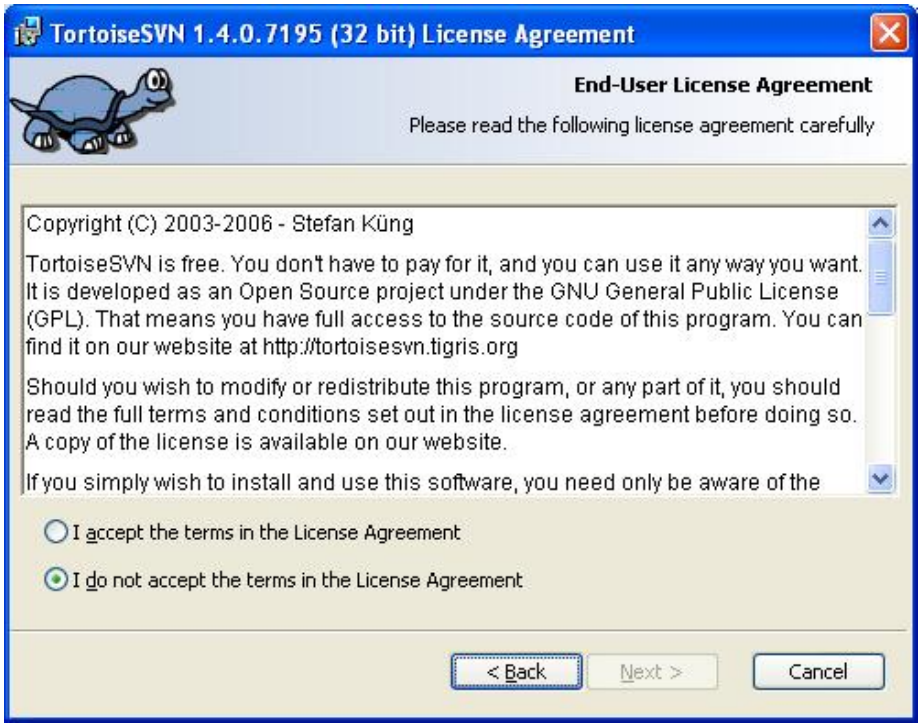
4. Dar doble clic sobre el icono

5. Se observa la pantalla de bienvenida a la instalación.



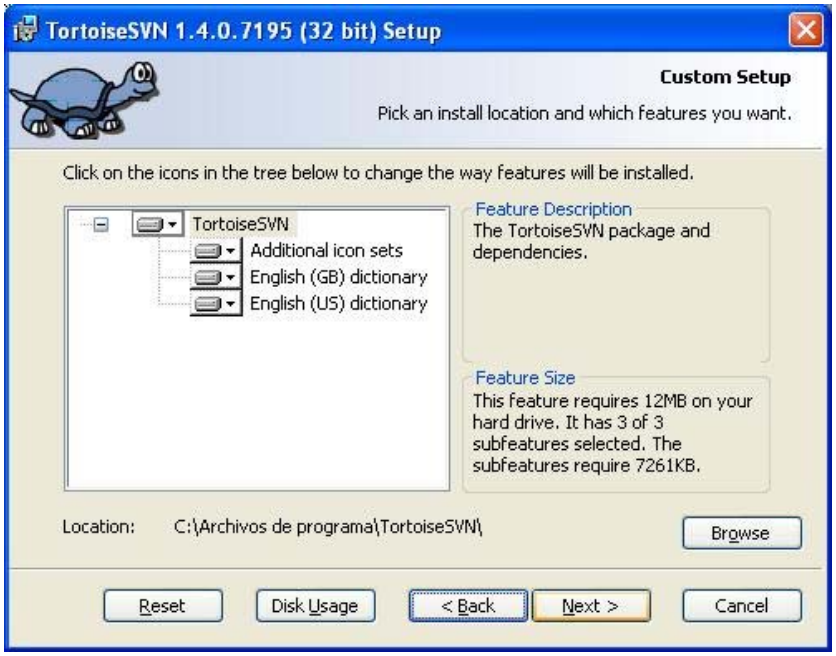
Para continuar oprimir “Next”

A continuación se muestra la pantalla de la licencia.



6. Para continuar se debe escoger “I accept the terms in the License Agreement” y oprimir “Next”.

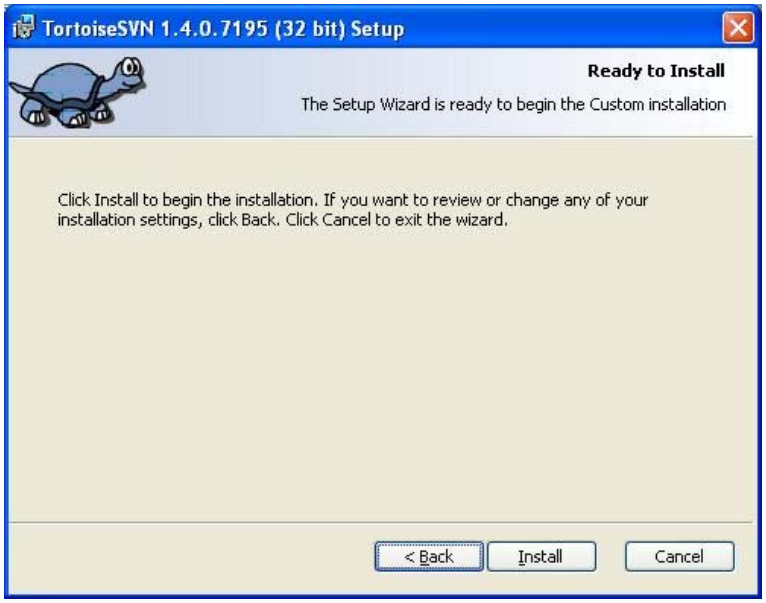
En la siguiente pantalla se escoge el destino de la aplicación y se muestran los paquetes que serán instalados.



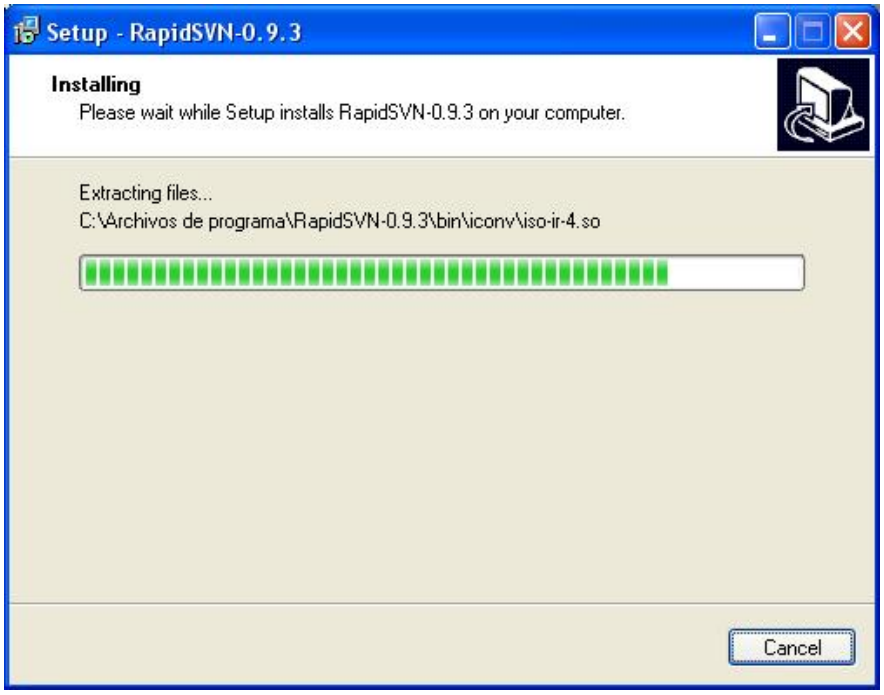
Normalmente el archivo por defecto es C:/Archivos de programa/TortoiseSVN, pero este archivo puede ser cambiado.

7. Para continuar oprimir “Next”

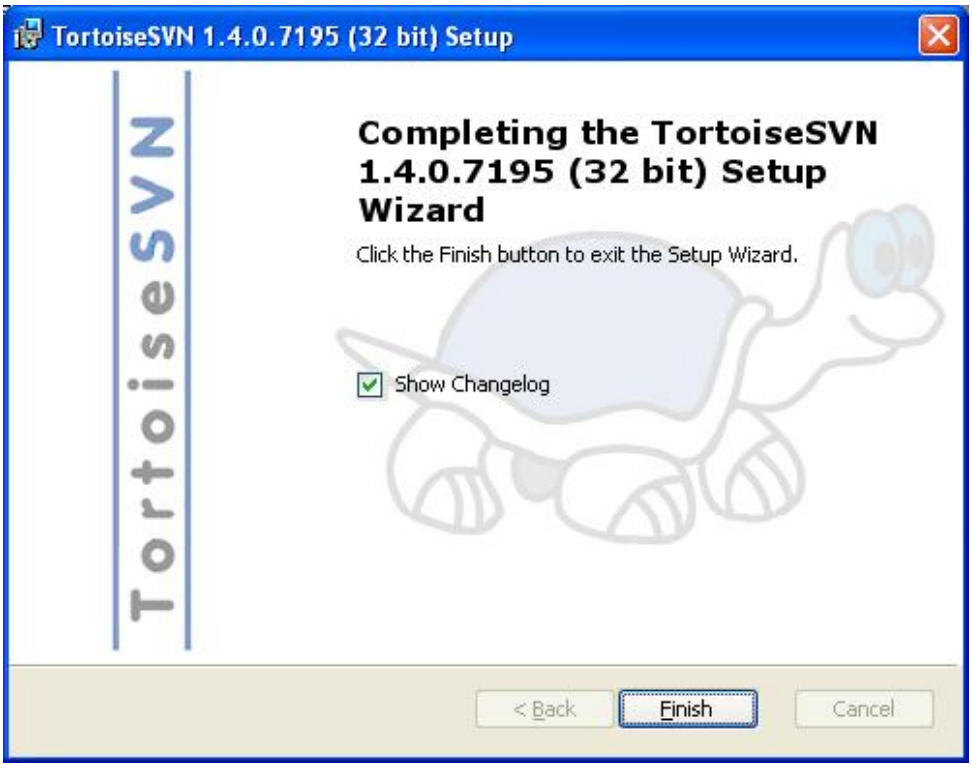
8. En la siguiente pantalla se pregunta por el inicio de la instalación.



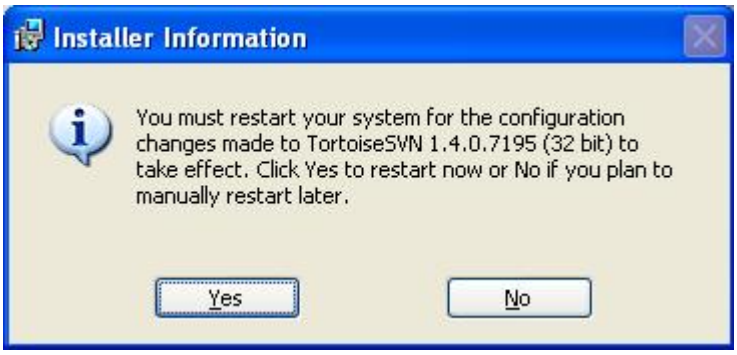
9. La siguiente pantalla muestra el proceso de instalación en el computador



10. La siguiente pantalla muestra que el proceso a culminado con éxito.



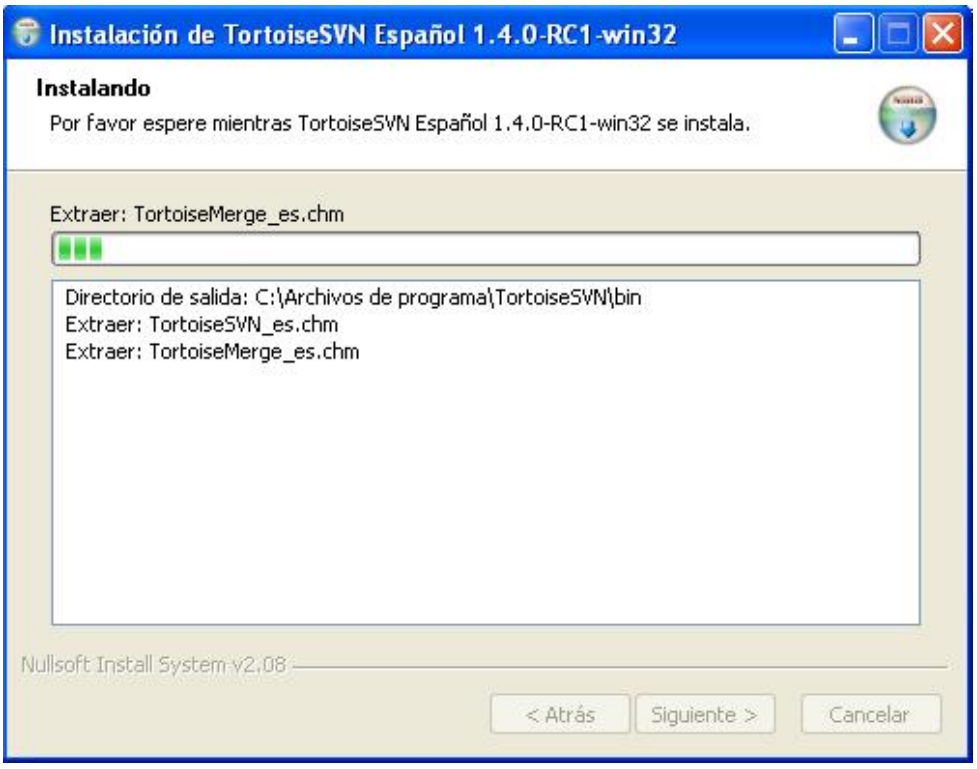
11. La ventana indica que los cambios solo tendrán efecto al reinicio del computador. Se debe aceptar.



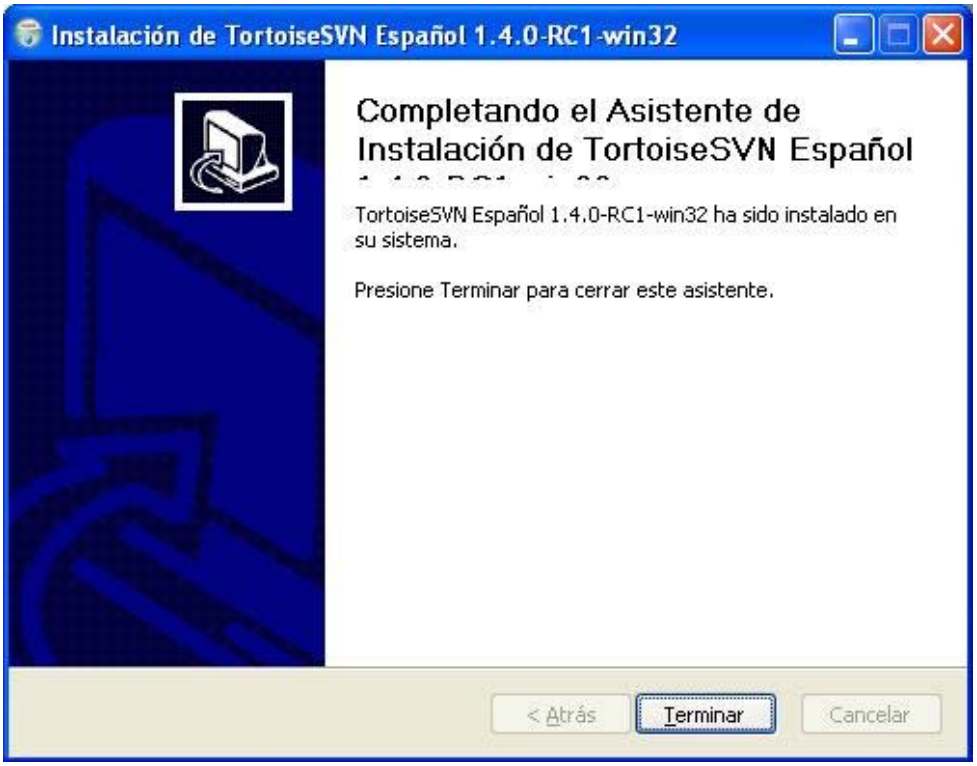
12. Una vez se ha reiniciado el sistema se puede empezar con la instalación del paquete de idioma.

13. La pantalla muestra la bienvenida al paquete y pide iniciar la instalación del paquete.

14. La siguiente ventana muestra el progreso de la instalación.



15. La pantalla final informa que la instalación ha sido completada.



16. Una vez los dos están instalados, en el menú contextual (Clic derecho a cualquier carpeta) de TortoiseSVN se busca Settings y se cambia el idioma a español.

Si se desea también es posible bajar el diccionario para que pueda hacer corrección ortográfica en español dentro de los mensajes de registro que son solicitados al importar un proyecto o al enviar las confirmaciones, como se vera más adelante.

Una vez ha sido configurado el idioma TortoiseSVN esta listo para ser usado de una manera sencilla a través de los menús contextuales.

El anterior es el menú de TortoiseSVN sobre una carpeta cualquiera.

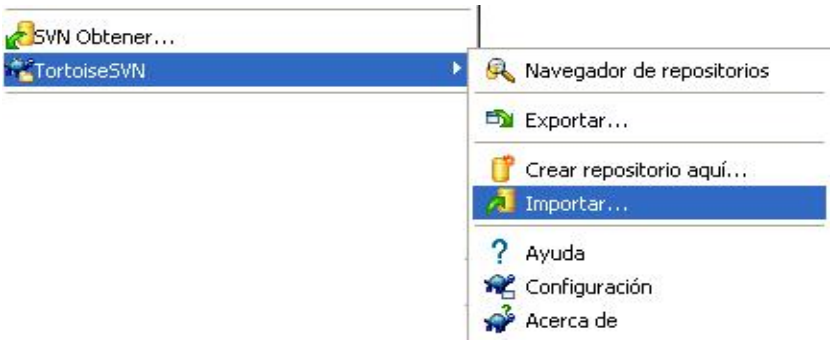
Este es el menú de una carpeta donde se encuentra una copia de trabajo local en el sistema, en el se muestran todas las posibles opciones de uso, así como una ayuda completa sobre la herramienta.

IMPORTAR EL CONTENIDO INICIAL

Inicialmente el repositorio esta completamente vació y muchas veces es necesario introducir en el la información que el proyecto tiene inicialmente. Aunque es recomendable empezar el proyecto de cero y no hacer este procedimiento. Se aclara que esta operación se hace solamente una vez.

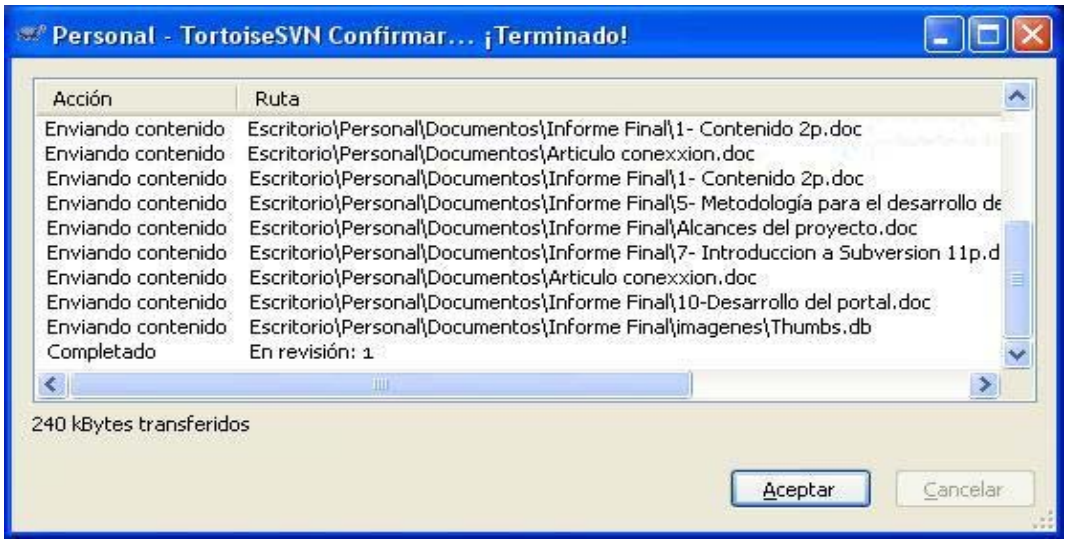
(NOTA: Este término puede ser un poco confuso. Importar significa subir los datos al servidor).

1. Seleccionar Importar del menú ToroiseSVN de la carpeta que se quiere enviar al repositorio.



2. Luego de esto se abrirá la siguiente ventana. En la URL se pone la dirección del repositorio que da el administrador.

3. Para continuar hacer clic en “Aceptar” el proceso se mostrara en una ventana nueva.



Cuando estos pasos se han realizado, la información ya se encuentra en el repositorio central, pero es necesario obtener una copia de esta para trabajar en ella de forma local. Esto se vera en el siguiente capitulo.

CREAR UNA COPIA DE TRABAJO LOCAL

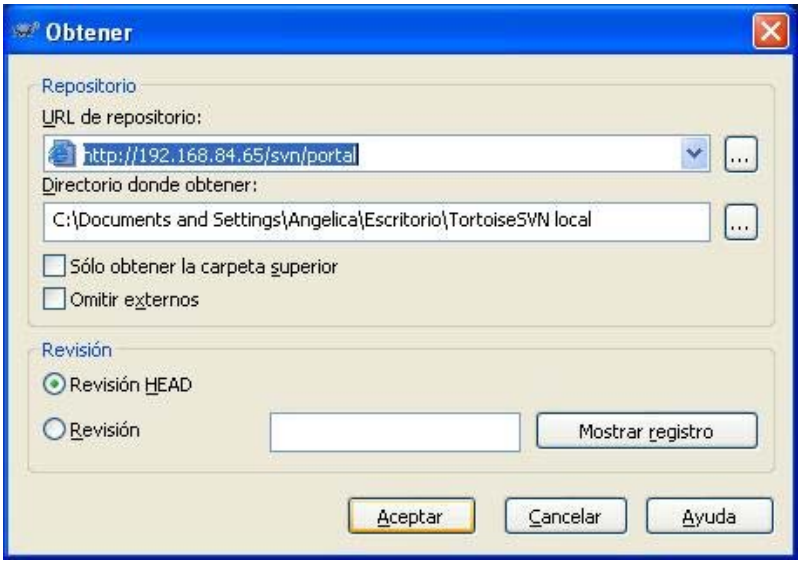
A fin de editar o crear aplicaciones y documentos, será necesario crear una copia local de lo contenido en el repositorio central para trabajar de forma sobre él. Esta copia es completamente independiente del repositorio central, lo que significa que cualquier cambio que se haga sobre ella no tendrá efecto en el repositorio central hasta que ellos no hayan sido enviados. A continuación se encuentran los pasos para crear una copia local en su computador.

Para realizar esta y las demás operaciones es necesario tener una conexión al menos temporal a Internet. Los cambios pueden tener lugar sin conexión alguna.

1. Crear una carpeta para contener la copia local. Para efectos de este manual llamaremos esa carpeta TortoiseSVN local.

2. Seleccionar el menú contextual (clic derecho) y la opción SVN Obtener.

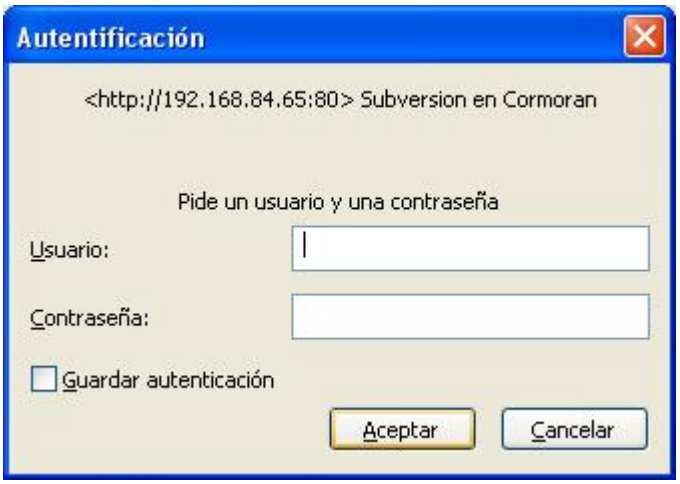
Al realizar esta operación aparece la siguiente ventana:



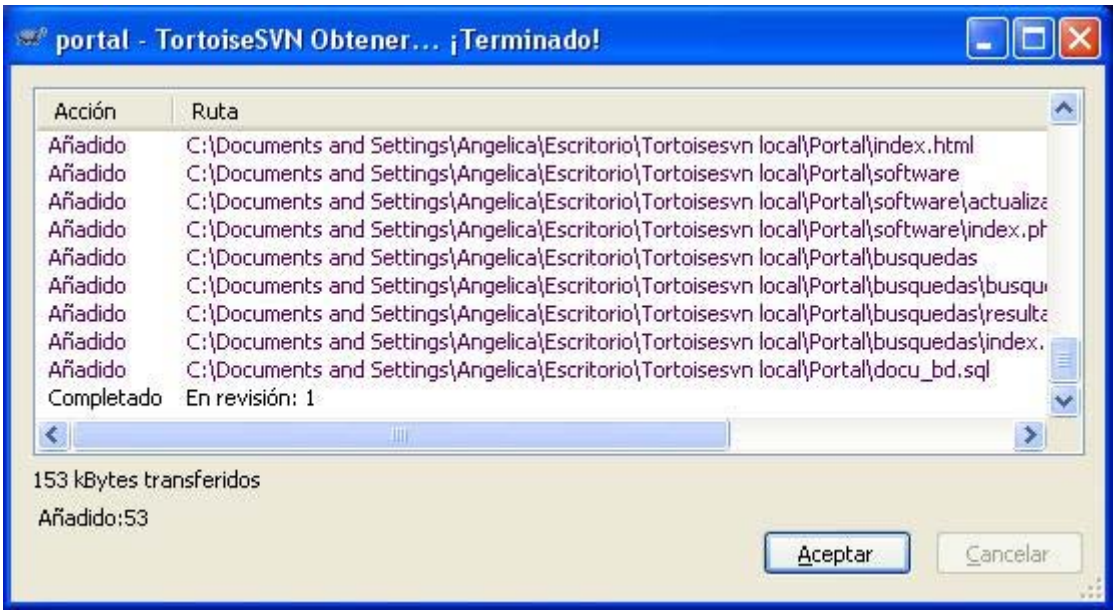
3. En la opción URL se introduce la dirección dada por el administrador. Por defecto esta URL se compone de la dirección de Cormoran "/svn/" el nombre del repositorio central, este será confirmado por el administrador en el momento de la creación del repositorio.

4. En la opción Directorio se busca con [...] la carpeta que se creo en el paso 1. Los demás parámetros se dejan por defecto a menos que se desee obtener una copia local con cambios antiguos.

5. Se solicita la autenticación del usuario y contraseña dados del usuario. Estos son inicialmente dados por el administrador en el momento de la creación de la cuenta de usuario, pero pueden ser cambiados a través de la página web.



6. Al oprimir el botón Aceptar empiezan a descargarse los archivos desde el repositorio central. Estos archivos empezaran a aparecer en una nueva ventana de confirmación.



Cuando los archivos son descargados a la copia local estos tendrán un icono marcado como normal. Sin embargo las marcaciones de los iconos pueden cambiar con el trabajo realizado sobre la copia. Estos iconos son utilizados por TortoiseSVN para hacer entender al usuario de una forma más fácil y agradable el estado de cada fichero. A continuación se explican algunos de ellos, sin embargo, dentro de la configuración estos iconos pueden ser cambiados.

Normal

Este icono es mostrado en todos los archivos cuya copia de trabajo, corresponde a la misma que la guardada dentro del repositorio.

Modificado

Este icono sugiere que el fichero ha sido modificado y que aun no se han confirmado los cambios realizados en el.

En Conflicto

Se muestra cuando después de una actualización, se descubre que los cambios guardados en el repositorio y en la copia de trabajo no son compatibles en algunas líneas.

Solo Lectura

Cuando en un fichero se a determinado la propiedad svn:needs-lock se imprime este icono para recordar que es necesario obtener un bloqueo antes de poder hacer cambios sobre el.

Bloqueado
Si ha obtenido un bloqueo sobre un fichero y su estado es normal, este icono le recordara que debe liberar el bloqueo para que otros usuarios puedan modificar dicho fichero.

Eliminado
Este icono sobre un fichero quiere decir que ha sido borrado del control de versiones, pero sobre una carpeta significa que falta un fichero bajo el control de versiones en el interior.

Añadido
Este icono indica que el fichero ha sido programado para estar bajo el control de versiones en la próxima confirmación.

ACTUALIZAR LA COPIA LOCAL CON LOS CAMBIOS DE OTROS

Si se esta trabajando en grupo, normalmente es necesario incorporar los cambios hechos por otras personas a la copia local. El proceso de obtener estos cambios desde el servidor que contiene el repositorio central, se llama actualizar.

1. Para realizar una actualización correcta, se recomienda seleccionar la carpeta TortoiseSVN local o la carpeta raíz del proyecto, aunque es posible llevar a cabo la actualización individual de cualquier carpeta o archivo bajo el control de versiones.

2. Hacer clic derecho sobre la carpeta o archivo y seleccionar la opción SVN Actualizar.

Esta opción se confirma en una nueva ventana.

CONFIRMAR CAMBIOS

Cuando se han realizado satisfactoriamente los cambios necesarios en la copia de trabajo local, es necesario reenviar la información de vuelta al repositorio central. Esta operación se llama confirmar o commit.

Generalmente los archivos modificados aparecen con un icono como el siguiente, lo que indica que existen cambios sin confirmar en el mismo.

1. Realizar actualización completa (pasos en el capitulo anterior). Esta operación es necesaria para verificar si existen conflictos.

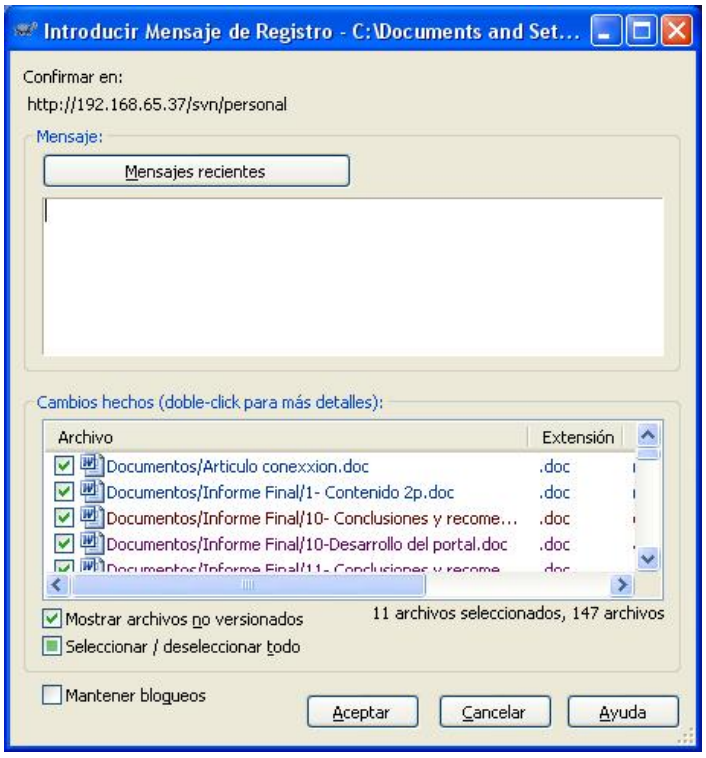
2. De ser necesario realizar la resolución de conflictos, aunque esto generalmente no sucede.

3. Seleccionar el archivo o carpeta a confirmar, se sugiere que esta sea la carpeta raíz para evitar conflictos con otros archivos más adelante.

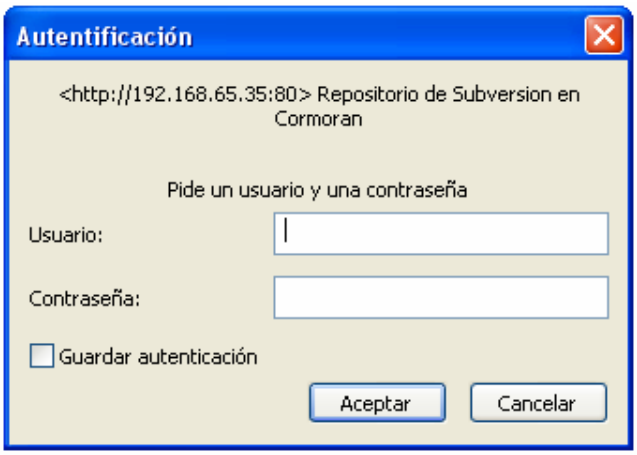
4. Hacer clic derecho y seleccionar la opción SVN Confirmar del menú.

Con esta operación la confirmación de cambios se abre un dialogo con los cambios realizados a los archivos.

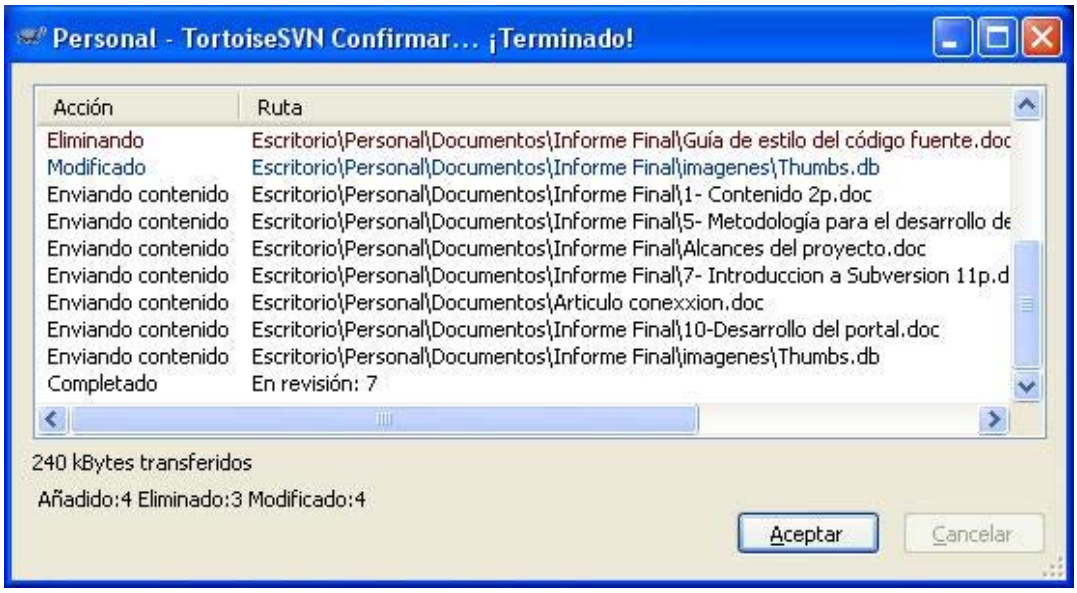
5. Se deja un mensaje con la descripción de los cambios realizados desde la última actualización para tener el historial.



6. Se realiza la autenticación de usuarios.



7. Se abre una ventana que confirma los cambios realizados en el repositorio central.

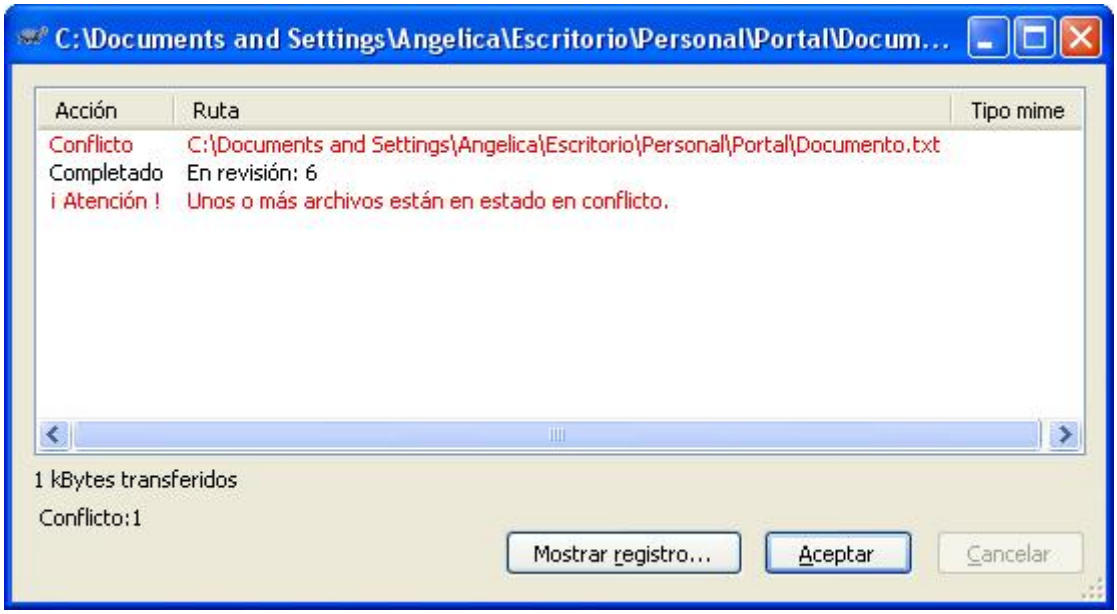


RESOLUCIÓN DE CONFLICTOS

Ocasionalmente se encontrara un conflicto entre los archivos de alguien más y los propios. Esto ocurre cuando dos personas modifican la misma parte de un documento. SubVersion no puede resolver por si solo el conflicto y lo deja para la persona que realiza la actualización. Estos conflictos pueden ser evitados si antes de trabajar en un archivo se hace una actualización, y justo después de terminar se confirman los cambios.

El estado original de un documento seria la siguiente

1. Cuando se realiza la actualización un mensaje de atención muestra que se presento un conflicto.



2. Al terminar esta operación tres nuevos archivos son creados con cada conflicto, estos tienen el mismo nombre del original pero con otras extensiones agregadas.

   Archivo.mine
   Archivo.r*
   Archivo.r*

   Los * son los numeros de revision en los que se presenta el conflicto.

3. El contenido de los archivos es algo parecido a lo siguiente según el caso.

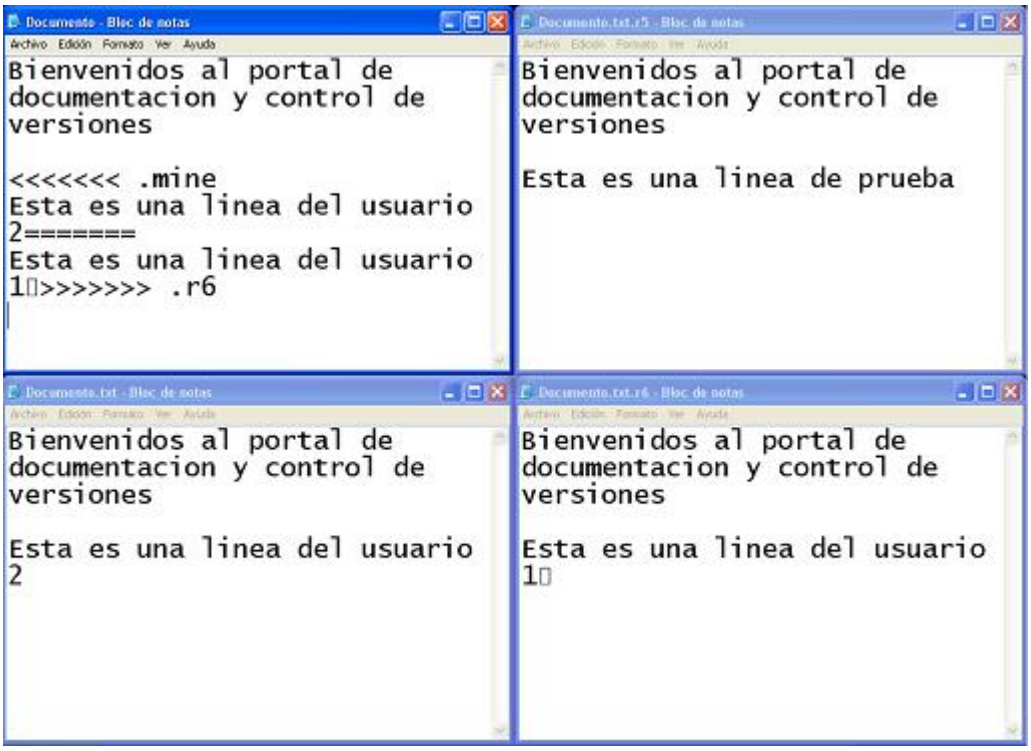
   El archivo original contiene la mezcla de los archivos en conflicto. Cuando se abre el archivo se encuentra una estructura parecida a la siguiente:

   <<<<<<< nombre_del_archivo

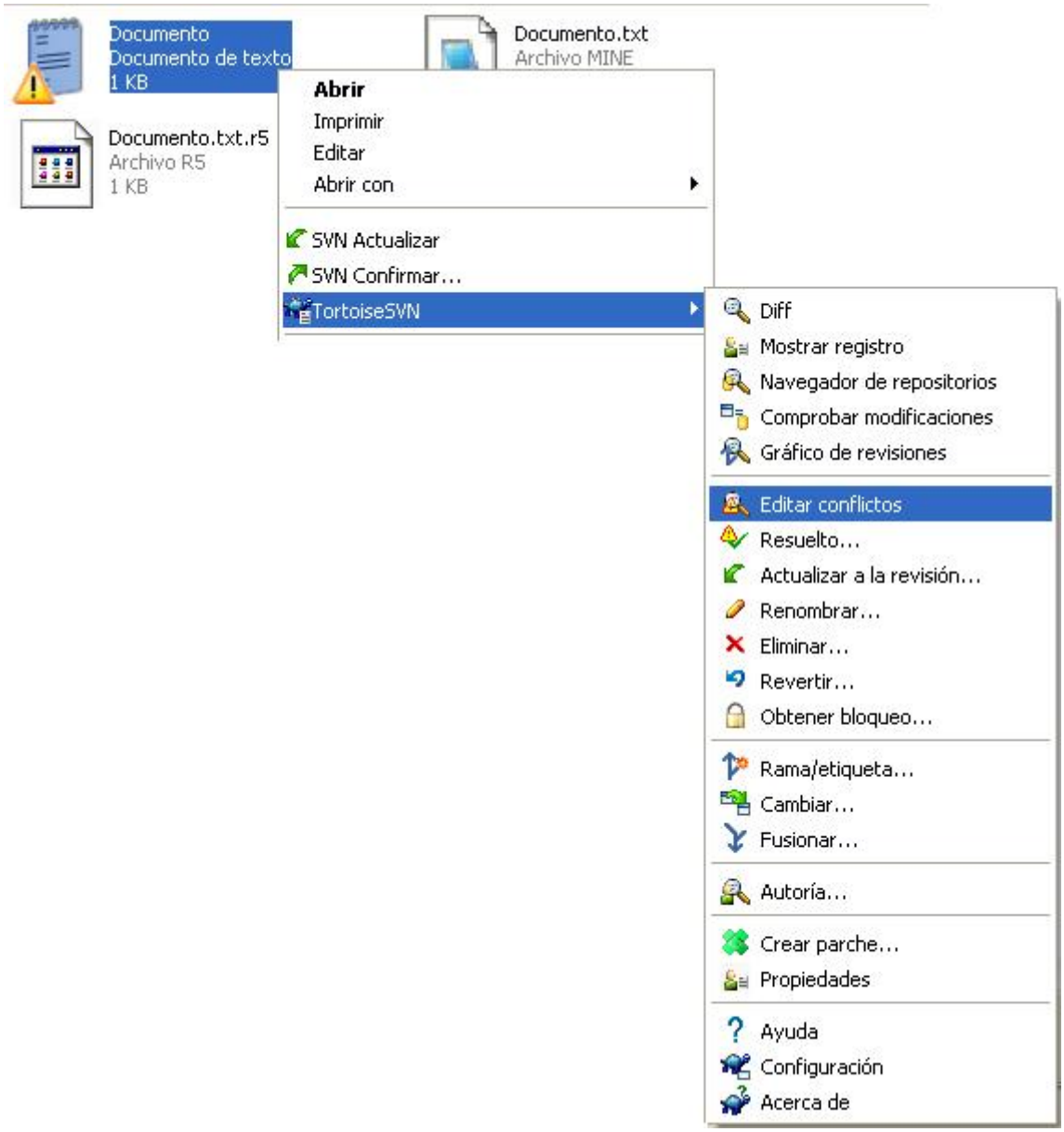
   Cambios locales
   Código mezclado del repositorio
   >>>>>>> Revisión

El archivo con extensión .mine es el archivo que se estaba modificando originalmente.

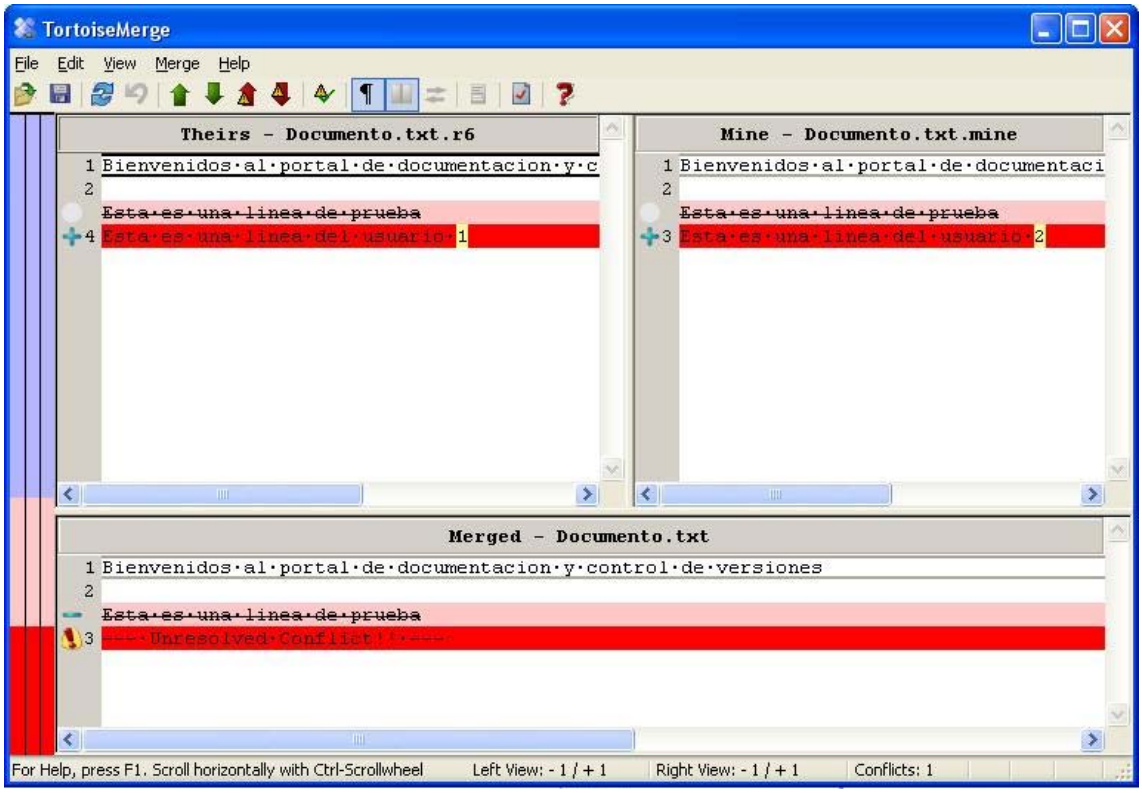
Los archivos con extensión .r* son las versiones anterior y actual.



4. Para solucionar los conflictos es posible modificar los archivos de forma manual o con la ayuda de una herramienta TorotoiseMerge proporcionada por TortoiseSVN. Para esto se hace clic derecho sobre el archivo original y se selecciona Editar conflictos del menú TortoiseSVN.



5. La herramienta se divide en tres partes. Cada una contiene uno de los archivos que se crearon después de la aparición de conflicto, durante la actualización. En la parte superior se encuentran los archivos tal y como se editaron por parte de cada desarrollador. En la parte inferior el archivo que debe ser editado para resolver el conflicto.

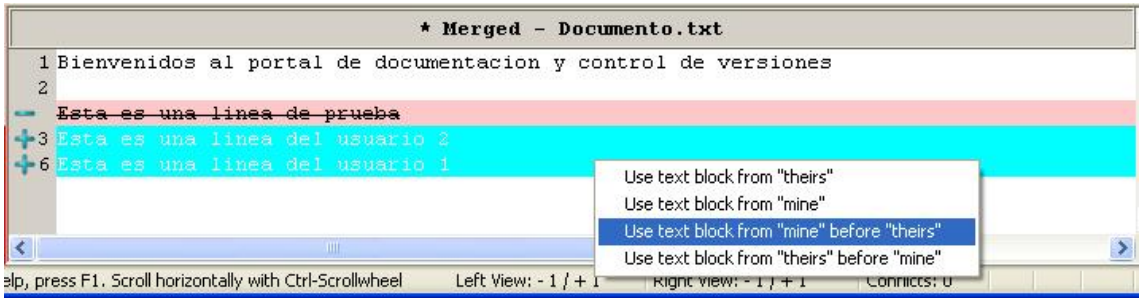


6. Al dar clic derecho sobre la línea rosa que contiene el conflicto se dan varias opciones:

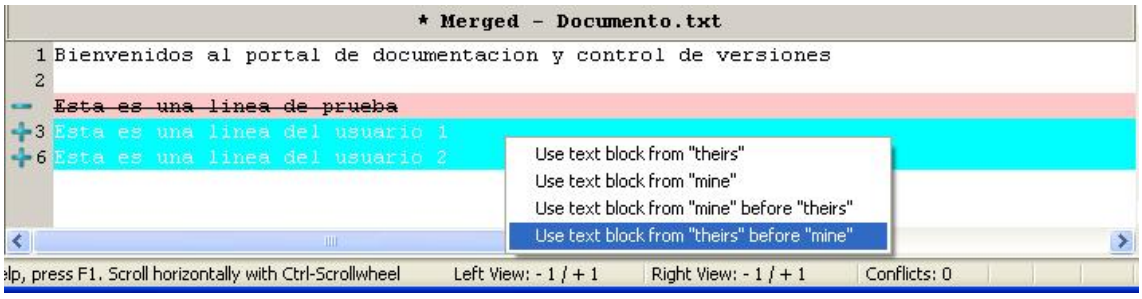
- Use text block from “theirs”: Esto implica dejar el archivo editado en la versión actual y descartar los cambios propios.

a. Use text block from “mine”: Esto implica dejar los cambios propios y descartar los demas.

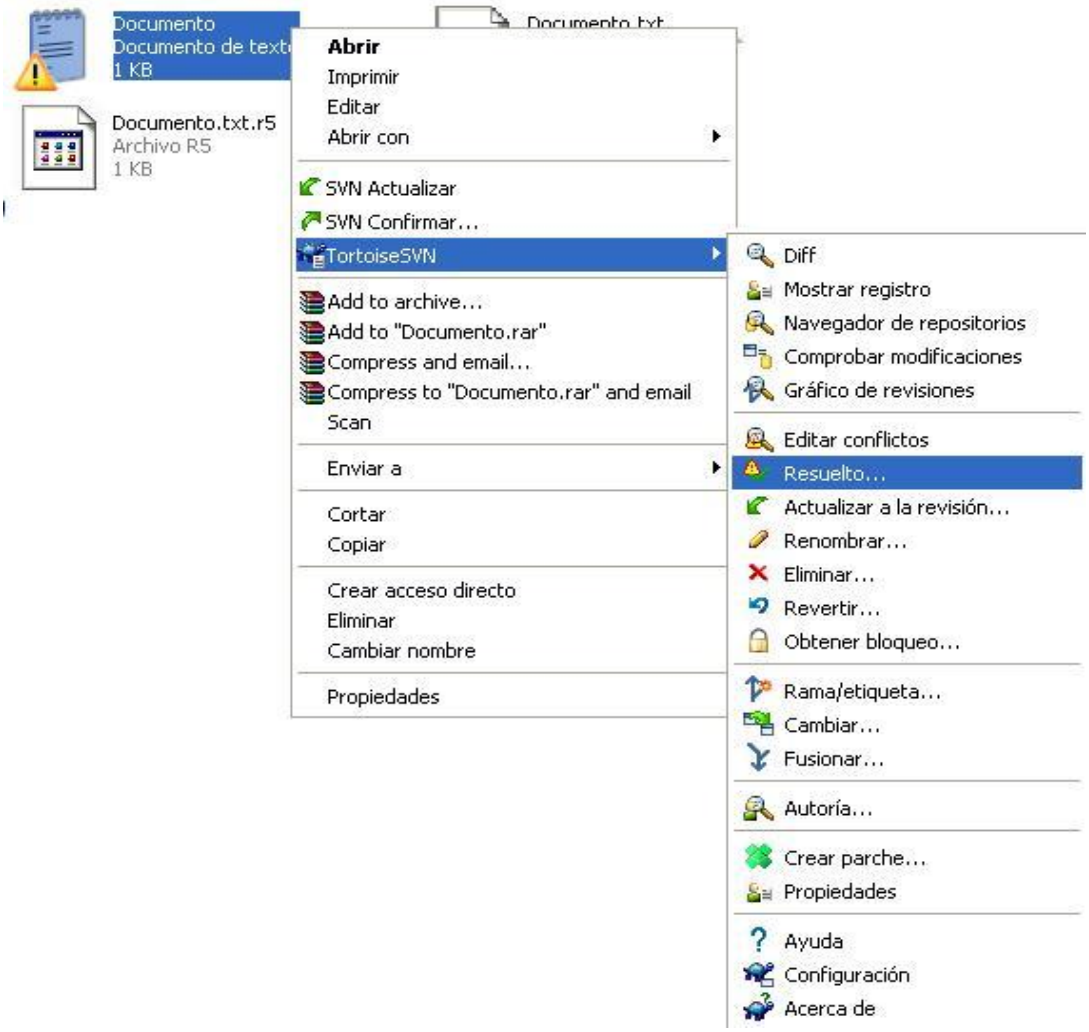
b. Use text block from “mine” before “theirs”: Esto quiere decir que se dejan los cambios propios antes de los de los demás.



c. Use text block from “theirs” before “mine”: Esto quiere decir que se dejan los cambios de los demás antes de los propios.

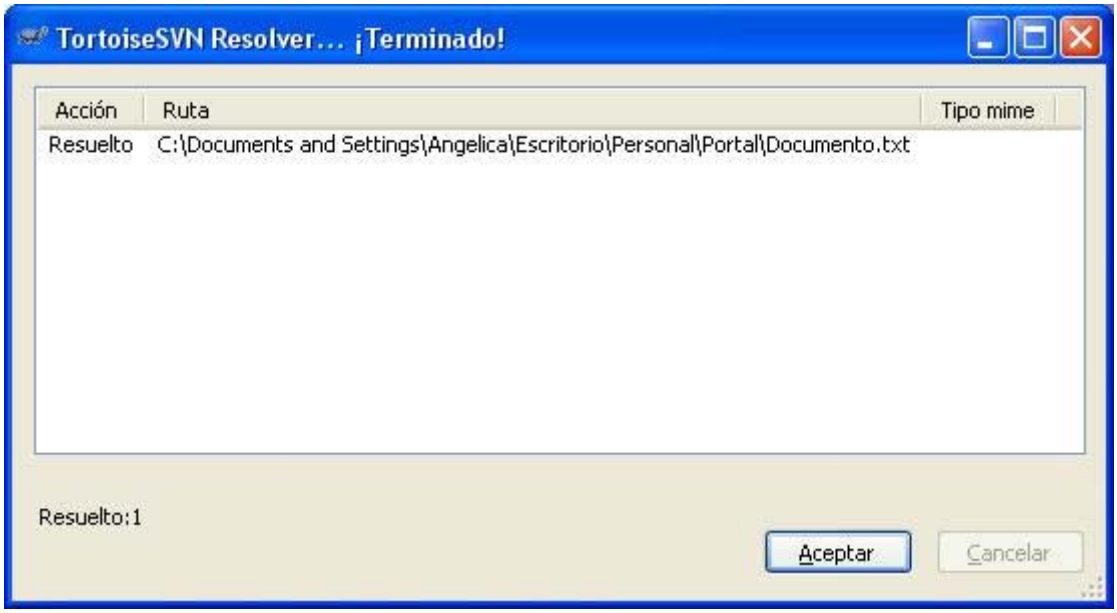


7. Una vez resuelto el documento se debe seleccionar la opción resuelto del menú de tortoiseSVN.



8. Aparece un cuadro de texto con el/los archivo(s) seleccionados para resolver conflictos. Aceptar

9. Una nueva ventana aparece confirmando la operación.



Para que la operación se complete correctamente es necesario hacer la confirmación de los cambios, para que estos queden también almacenados en el repositorio central.

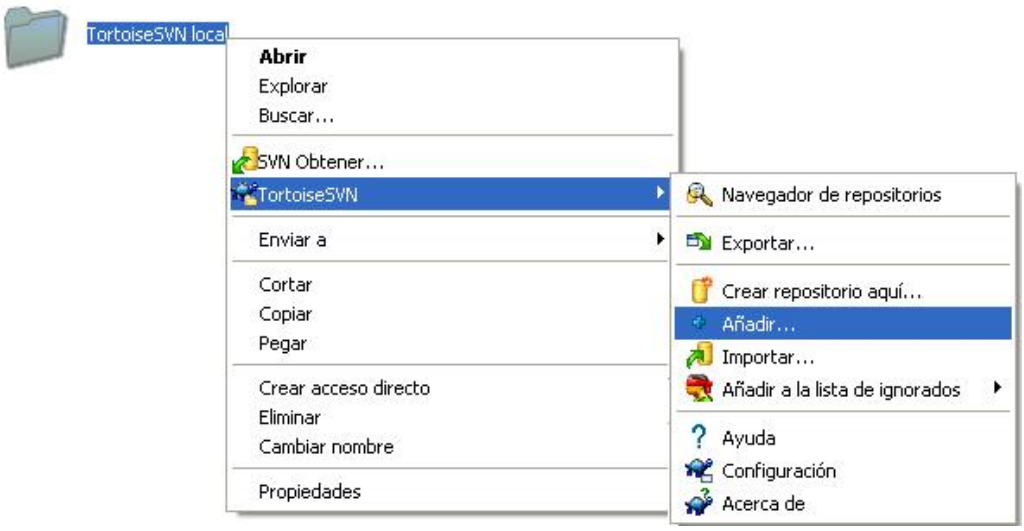
AGREGAR NUEVAS CARPETAS Y ARCHIVOS

Todos los nuevos archivos y directorios necesitan ser añadidos al repositorio central. Sin embargo, antes de que esto pase todos estos archivos deben ser agregados en la copia de trabajo local.

Para agregar una nueva carpeta que contiene uno o más archivos a la copia de trabajo local:

1. Ubique la carpeta o archivo que desea añadir en la copia de trabajo local.

2. Con clic derecho seleccionar la opción Añadir del menú TortoiseSVN.



3. Una confirmación de los archivos a añadir se solicita en una nueva ventana.

4. Al aceptar la información se abre una ventana de confirmación.

5. Al finalizar la carpeta original tendrá un icono con un mas azul que indica que este archivo esta por ser añadido.

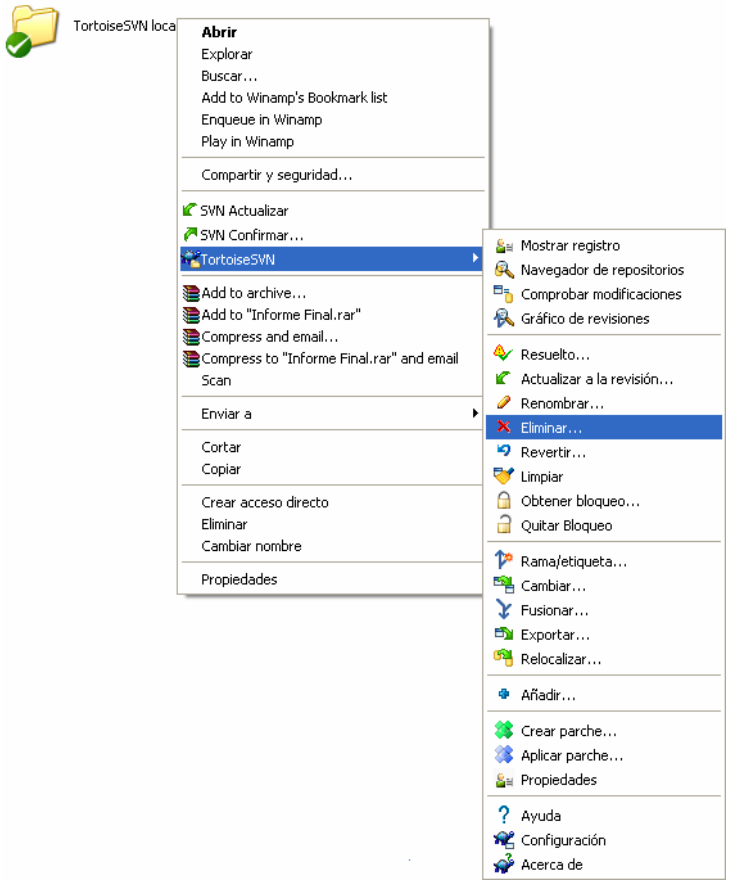
Para que la operación se complete correctamente es necesario hacer la confirmación de los cambios, para que estos queden también almacenados en el repositorio central.

BORRAR Y RENOMBRAR ARCHIVOS Y CARPETAS.

1. Ubique el archivo al que desea efectuarle la operación.

Borrar

2. Con clic derecho seleccionar la opción Eliminar del menú TortoiseSVN.



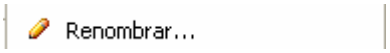
3. Después aparecerá una ventana de confirmación de la operación.

4. Los archivos seleccionados en este momento serán marcados con una X en el icono. En este punto es posible deshacer la acción haciendo clic en el icono Revertir.

5. Para que los cambios sean realizados es necesario hacer una confirmación, con todos los pasos que esto indica.

Renombrar archivos

2. Con clic derecho se puede seleccionar la opción renombrar del menú TortoiseSVN.



3. En un cuadro de dialogo se debe dar el nuevo nombre dado al archivo, con esta operación no se pierde la historia del archivo versionado.

Nota: Si el archivo es renombrado de manera tradicional, al momento de actualizar el repositorio se descargara el archivo que se encuentra en el servidor y el archivo original no se encontrara bajo el control de versiones.

4. Para que el cambio sea llevado a cabo para todos los usuarios del repositorio se debe hacer la confirmación de los cambios.

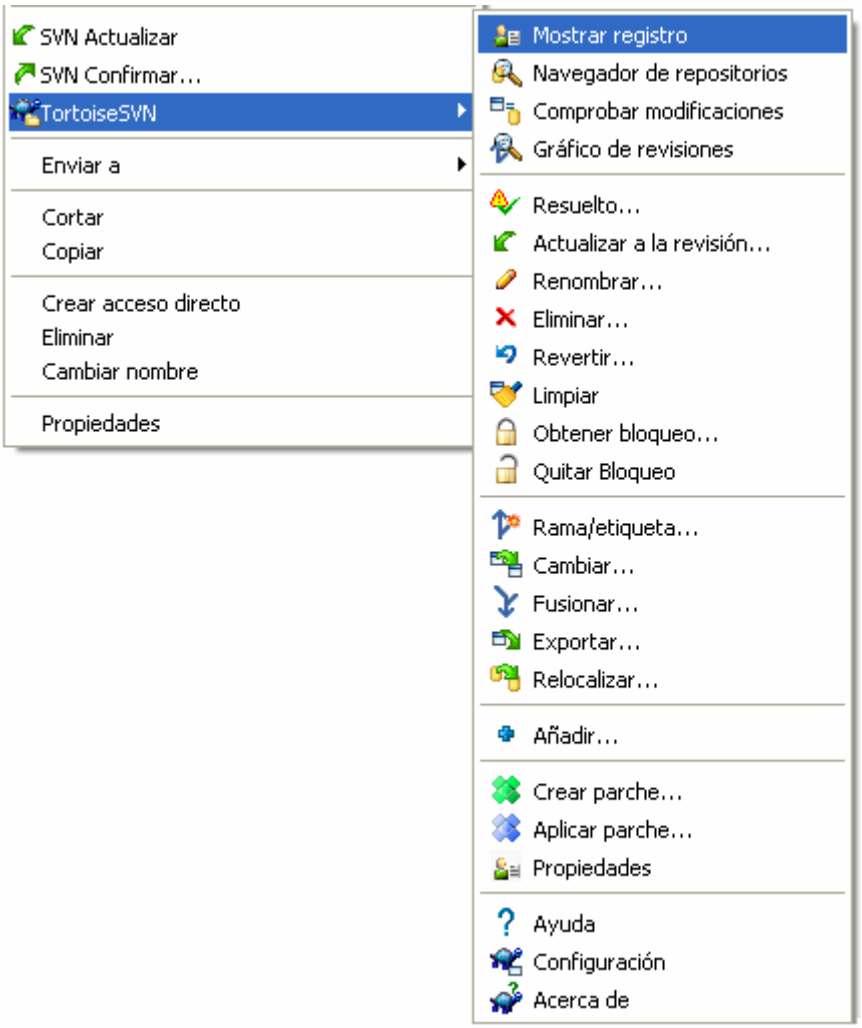
CANCELAR LOS CAMBIOS EN LOS ARCHIVOS

Si usted quiere cancelar todos sus cambios desde la ultima actualización debe seleccionar el archivo y dar clic derecho sobre el y seleccionar la opción Revertir del menú TortoiseSVN.

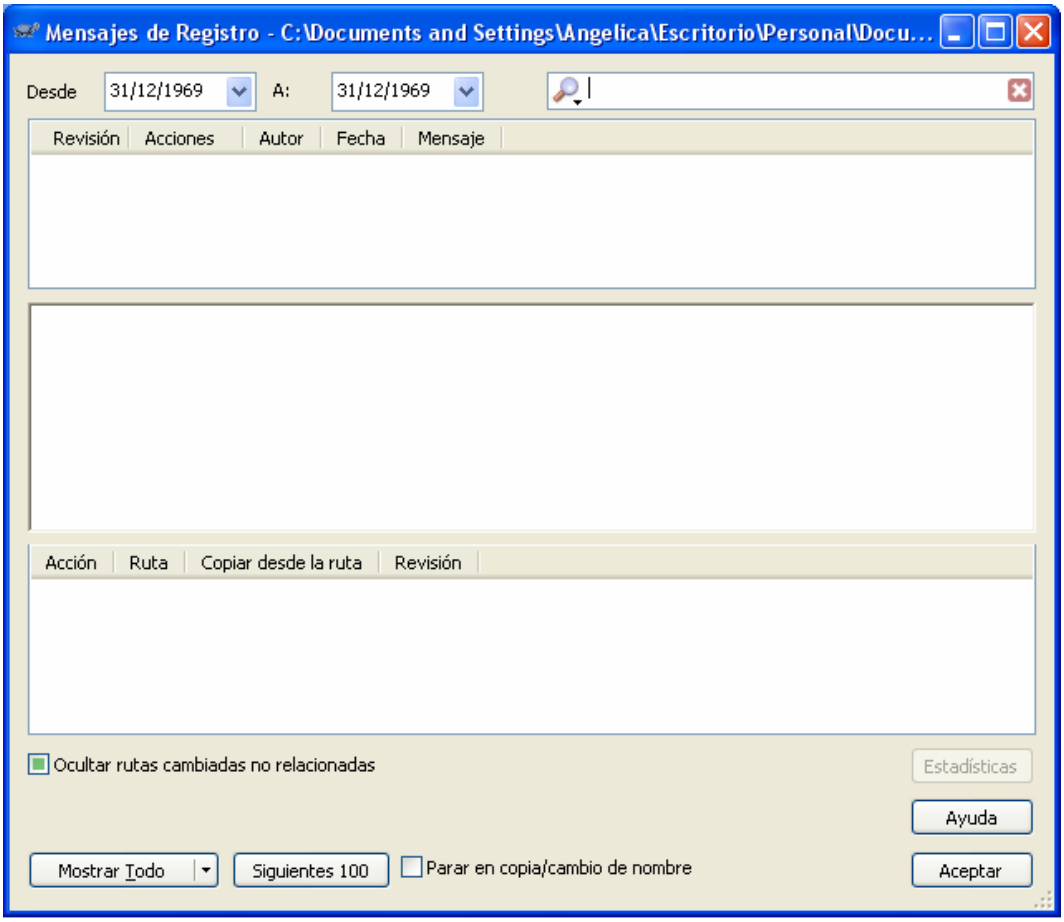
Hay que tener en cuenta que los cambios se perderán y el archivo volverá totalmente al estado anterior.

OBTENER INFORMACIÓN ACERCA DE ARCHIVOS Y DIRECTORIOS

Hay ocasiones en las cuales es necesaria más información acerca de un archivo particular. Esta información se puede obtener en la opción mostrar registros del menú de TortoiseSVN.



Al seleccionar lo anterior aparece una ventana con la información del archivo, que ha sido guardada cada vez que se lleva a cabo una operación sobre el mismo.



SOLUCION A ERRORES FRECUENTES

Error:
Your .svn/tmp directory may be missing or corrupt; run 'svn cleanup' and try again.

Causa:
Existen varias causas, la más común es que alguno de los usuarios hubiera tenido un conflicto relacionado con alguno de los archivos en esta copia de trabajo.

Solución:
Ejecutar el comando Cleanup del menú TortoiseSVN.

Anexo C. MANUAL DE USUARIO

MODOS DE ACCESO

Es posible ingresar con diferentes roles al portal Web, las características principales son las siguientes.

- Usuario Invitado

Inicialmente todos los usuarios ingresan al sistema como usuario invitado. Desde este rol es posible realizar búsquedas simples y avanzadas, ingresar a los proyectos existentes previa autenticación de los mismos y acceder al software utilizado como cliente gráfico de SubVersion y a los manuales desarrollados para estos.

- Usuario Registrado

Cuando el administrador ha registrado un usuario en el sistema, este puede ingresar con su usuario y contraseña, desde cualquiera de las plantillas a las que puede acceder como usuario invitado:

Una vez ha ingresado como usuario registrado su rol cambia, y en el servicio de búsqueda podrá acceder a documentos adicionales, relacionados por el administrador a su rol. Adicionalmente, si pertenece a un proyecto contenido en el control de versiones, cuyo sitio Web es actualizado desde el portal, podrá hacer una copia al sitio de pruebas dispuesto por el administrador del sistema, o directamente al sitio del portal según el tipo de acceso que tenga sobre el repositorio que contiene el proyecto. En este paso se deben tener en cuenta los permisos para cada repositorio, pues estos son totalmente independientes del rol de usuario.

- Usuario Administrador

El usuario Administrador debe introducir su usuario y contraseña de la misma manera que cualquier usuario registrado. Los documentos y proyectos a los que puede acceder dependen de los permisos que tenga sobre ellos, de esta manera, no necesariamente puede acceder a todas las opciones. Al administrador se muestran las siguientes opciones:

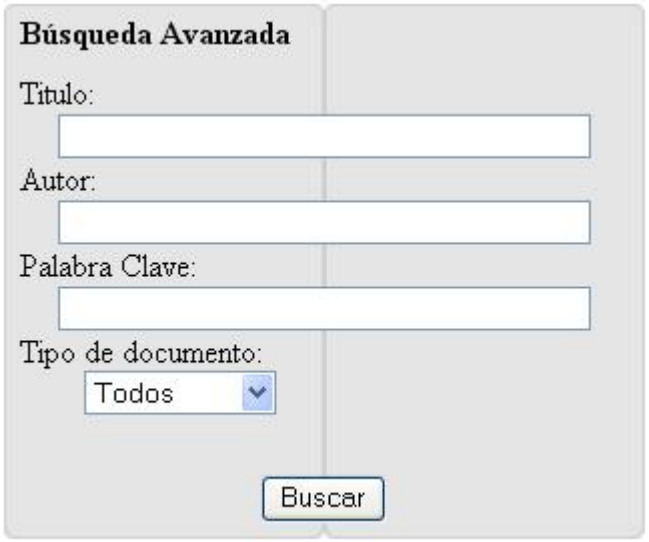
DISTRIBUCIÓN DE LA APLICACIÓN

- Búsquedas

En el servicio de búsquedas se dispone de los documentos propios del servidor de la escuela de ingeniería de sistemas, cormorán. Los resultados de la búsqueda dependen del perfil del usuario que la realice.

En la primera pantalla se encuentra un buscador simple, que sin importar en que campo se encuentre la palabra de búsqueda mostrará todos los resultados.

También se encuentra un vínculo a la búsqueda avanzada, en la cual se pueden realizar las búsquedas de forma más específica.



Cada uno de los documentos en el sistema esta catalogado por titulo, autor, palabras clave y tipo de documento. Los tipos de documentos pueden variar.

- Proyectos

En el menú proyectos se encuentra el listado de los repositorios ubicados bajo el sistema de control de versiones. Esta lista es el vínculo a cada uno de los proyectos, en los que para poder ingresar es necesario tener un usuario registrado en dicho proyecto.

Adicional a esto, en algunos proyectos es posible realizar la actualización a un sitio de pruebas o a un sitio de producción. Esta acción solo puede ser llevada a cabo por un usuario autorizado para dicha operación. El usuario puede realizar las dos operaciones, solo una o ninguna, a pesar de estar registrado en el proyecto.

- Software

En esta sección se encuentra información sobre dos de los clientes gráficos utilizados para el sistema de control de versiones, así como los manuales desarrollados para las mismas. Esta información esta disponible para cualquier usuario en todo momento.

RapidSVN puede ser utilizado en sistemas operativos Linux, Windows y Mac, mientras que TortoiseSVN se integra al núcleo de Windows.

- Administración

El usuario administrador puede realizar las siguientes opciones:

Catalogación de documentos: Introducir datos, asignar permisos de acceso por rol, listar, modificar y eliminar, información sobre documentos de diferentes tipos almacenados en el repositorio asignado para tal fin.

Administración de usuarios: Crear nuevo, listar, modificar, y eliminar usuarios del sistema.

Administración de repositorios (proyectos): Crear nuevo, asignar usuarios, asignar permisos, listar, modificar proyectos, modificar asignación de usuarios, modificar asignación de permisos y eliminar repositorios o proyectos.

Administración de roles de usuario: Crear nuevo, listar, modificar y eliminar roles de usuario. Estas opciones no están disponibles para el rol administrador ni el rol invitado.

Administración de tipos de documentos: Crear nuevo, listar, modificar y eliminar tipos de documentos.

Creación de copias de seguridad: es posible realizar la copia de seguridad en cualquier momento, además de la copia semanal.

- Opciones adicionales

Cerrar Sesión: Vuelve al inicio como usuario invitado.

Cambiar Datos: Es posible cambiar el usuario y la contraseña. (La contraseña debe tener al menos 6 caracteres).

Ir al inicio: Vínculo a la página inicial.

Contáctenos: Información de contacto con el administrador del sistema.

Ayuda: Proporciona al usuario la información necesaria para manipular la herramienta.

Cerrar: Termina la aplicación.

Procedimiento para control de versiones de un proyecto

1. Obtener una cuenta de SubVersion del administrador, para esto se debe tener la autorización del director del proyecto.

2. Solicitar la creación de un repositorio para control de versiones con los permisos para cada uno de los miembros del grupo de desarrollo.

3. Solicitar al administrador, la IP de conexión, la forma de acceder al repositorio y a cada una de las copias.

4. Importar el contenido inicial del proyecto utilizando uno de los clientes gráficos que puede encontrar en el menú Software.

   Al momento de importar el proyecto es mejor eliminar de este fichero toda la información que no quiera tener bajo el control de versiones, y organizarlo en ficheros y carpetas. Aunque después se pueden mover y renombrar es mucho más sencillo hacerlo antes de importarlo.

5. Verificar la información importada ingresando al menú proyectos y al repositorio correspondiente.

6. Obtener una copia de trabajo en el equipo local de cada uno de los miembros del grupo desarrollador.

7. En esta copia se realizan todos los cambios necesarios durante el proceso.

8. Al terminar los cambios, cada uno de los desarrolladores debe sincronizar los cambios con el repositorio, inicialmente se actualiza y si la operación se lleva a cabo correctamente se confirman los cambios.

9. Si desea ver la página Web es necesario dar clic en actualizar pruebas en la misma sección de la página.

10. Cuando el grupo esta totalmente satisfecho con los cambios hechos a la página pueden ser publicados haciendo clic sobre actualizar sitio.

11. Pruebe en el sitio Web que los cambios tuvieron efecto.

No es aconsejable trabajar y publicar en solo sitio pues esto puede resultar en páginas rotas y causar problemas en la administración.