



# **UNIVERSIDAD NACIONAL DE LOJA**

**Área de la Energía, las Industrias y los  
Recursos Naturales no Renovables.**

## **TITULO**

**“CONFIGURACION E IMPLEMENTACION  
DE UN SERVIDOR DE REPOSITORIO DE  
VERSIONES UTILIZANDO SVN EN LINUX,  
PARA LA UNIDAD DE DESARROLLO  
INFORMATICO DEL A.E.I.R.N.N.R DE LA  
UNIVERSIDAD NACIONAL DE LOJA”**

**Tesis previa a la obtención del título  
de Ingeniero de sistemas**

## **AUTOR:**

**Diego Genaro Achupallas España**

## **DIRECTOR:**

**Ing. Germán Patricio Villamarín Coronel. Mg. Sc.**

**Loja – Ecuador**

**2011**



Ing. Germán Patricio Villamarín Coronel.

Docente de la carrera de Sistemas, del Área de la Energía las Industrias y los Recursos Naturales no Renovables.

### **C E R T I F I C A**

Que el trabajo de investigación, con el tema: **“CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMÁTICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA”**, presentado por el egresado: Diego Genaro Achupallas España, ha sido verificado en su totalidad por lo que cumple con los requisitos reglamentarios, autorizando su presentación y defensa correspondiente.

Loja, Enero del 2011.

F\_\_\_\_\_

**Ing. Germán Patricio Villamarín Coronel Mg.Sc.**  
**DIRECTOR DE TESIS**



## AUTORÍA

Los procedimientos de investigación consulta bibliográfica, conceptos, ideas analíticas y de redacción final, vertidos en el presente tesis de Ingeniero en Sistemas, denominada **“CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMATICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA”** son de exclusiva responsabilidad del autor.

F \_\_\_\_\_

Diego Genaro Achupallas España



## **AGRADECIMIENTO**

Al termino del presente trabajo de investigación previo a la obtención del título de Ingeniero en Sistema. Agradezco en primer lugar a Dios, a mis padres que día a día lucharon por darme el estudio y estar aquí presente al término del mismo.

A la Universidad Nacional de Loja, al Área de Energía Industrias y Recursos Naturales No Renovables, quien supo guiarme desde el inicio hasta ser parte del medio profesional competitivo de la sociedad.

A la Ing. Mireya Erreyes, Milton Labanda quien al inicio de la carrera me dieron los conocimientos necesarios para mi superación académica.

A los encargados del departamento de la unidad de desarrollo informático (UDI). En especial al Ing. Patricio Villamarín, quienes me brindaron su apoyo incondicional en el trascurso del desarrollo de la tesis.

A todo el cuerpo docente de la carrera de Ingeniería en sistemas por sus enseñanzas y contribución de mi formación profesional y personal. Y en general a todas las personas que contribuyeron en mi formación académica e intelectual.

Son todos estos seres quienes merecen mi reconocimiento y gratitud por ser apoyo para el logro de este sueño que ahora se hace realidad.





## DEDICATORIA

El presente trabajo investigativo deseo dedicarlo mis padres Carmen España y José Achupallas quienes con su ejemplo, esfuerzo, dedicación, amor y ternura me enseñan el verdadero significado de la vida del porque? y para qué?, y me mostraron que los esfuerzos tienen siempre sus frutos; a mis hermanos, cuñadas, sobrinos y sobrinas quienes representan la compañía y alegría en mi vida, y mis amigos que han estado junto a mí en momentos de alegría y dificultad los cuales me dieron un apoyo moral incondicional.

De manera especial a la comunidad de los desarrolladores de Subversion, quienes me brindaron su información en los foros.

Son todos estos seres quienes merecen mi reconocimiento, dedicatoria y respeto.



## RESUMEN

El presente trabajo de investigación lleva como objetivo principal, incentivar a los estudiantes de la carrera de ingeniería en sistemas, en la parte del desarrollo de programas utilizando Subversion para manejar sus datos, quienes normalmente empleaban su tiempo haciendo pequeños cambios en el software y después deshaciendo esos cambios al día siguiente. Pero la utilidad del software de control de versiones se extiende más allá de los límites del mundo del desarrollo de software. Allá donde pueda encontrarse a gente usando ordenadores para manejar información que cambia a menudo.

Esta herramienta permite acceder al repositorio a través de redes, lo que le permite ser usado por personas que se encuentran en distintos ordenadores que tengan acceso al servidor de repositorio. A un cierto nivel, la capacidad para que varias personas puedan modificar y administrar los datos desde sus respectivas ubicaciones obteniendo un ahorro de tiempo al momento de fusionar sus cambios.



## SUMMARY

This research has as main objective to encourage students in systems engineering career in the software development using Subversion to manage their data, who usually spent their time making small changes to the software and then undoing these changes the next day. But the usefulness of version control software extends far beyond the limits of the software development world. Wherever you can find people using computers to manage information that changes often.

This tool allows access to the repository across networks, which allows it to be used by people who are on different computers that access the repository server. At one level, the ability for various people to modify and manage data from their respective locations to save time getting the time to merge your changes.



## ÍNDICE DE TEMAS

PORTADA.....	I
CERTIFICACIÓN .....	II
AUTORÍA .....	III
AGRADECIMIENTO .....	IV
DEDICATORIA .....	V
RESUMEN.....	VI
SUMMARY .....	VII
INDICE DE TEMAS .....	VIII
INDICE DE FIGURAS .....	XIII
INDICE DE TABLAS .....	XVI
<b>1 INTRODUCCIÓN .....</b>	<b>1</b>
<b>2 METODOLOGÍA .....</b>	<b>3</b>
<b>3 FUNDAMENTACIÓN TEÓRICA .....</b>	<b>5</b>
3.1 Que es GNU( Linux ).....	5
3.1.1 Distribuciones GNU (Linux) Linux .....	6
3.1.2 Instalar Linux (Linux Ubuntu) .....	8
3.2 Instalación.....	11
3.2.1 Instalar Paquetes GNU (Linux Ubuntu) Instalación.....	11
3.2.2 Eliminar Paquetes GNU(Linux Ubuntu).....	11
3.3 Control de Versiones.....	17
3.3.1 Introducción .....	17
3.3.2 Tipos Control de Versiones .....	18
3.3.2.1 CVS.....	18
3.3.2.2 Microsoft Visual SourceSafe .....	18
3.3.2.3 El SourceSafe .....	18



3.3.2.4 Mercurial .....	19
3.3.2.5 Bazaar .....	19
3.3.2.6 Git .....	20
3.3.2.7 Subversion .....	20
3.3.3 Clientes .....	21
3.3.3.1 Uso y reconocimiento .....	22
3.3.4 Características.....	22
3.3.4.1 Características Control de versiones .....	22
3.3.5 Clasificación.....	22
3.3.6 Repositorio.....	24
3.3.7 Vocabulario Común.....	28
3.4 Hardware y software para la implementación del servidor de repositorios svn, en la Unidad de Desarrollo Informático de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.....	28
3.4.1 Introducción .....	28
3.4.2 Servidor.....	30
3.4.3 Servidor http Apache.....	30
3.4.3.1 Módulos de apache .....	31
3.5 Tortoise svn. ....	32
3.5.1.1 Introducción .....	32
3.5.1.2 Características de TortoiseSVN.....	33
3.5.1.3 Integración con el shell de Windows .....	33
3.6 RapidSVN.....	34
3.6.1 Introducción .....	34
3.6.2 Características de RapidSVN.....	35
3.6.2.1 Integración con los sistemas Operativos .....	35
3.6.2.2 Sistema Operativo.....	35
<b>4 EVALUACIÓN DEL OBJETO DE INVESTIGACIÓN.....</b>	<b>35</b>
4.1 Características y Requerimientos de Hardware y Software .....	37



4.1.1 Hardware Mínimo .....	37
4.1.2 Software Mínimo .....	37
<b>5 DESARROLLO DE LA PROPUESTA ALTERNATIVA.....</b>	<b>38</b>
5.1 Análisis de la Aplicación .....	38
5.1.1 Definición del Problema .....	38
5.1.2 Diagrama del servidor de repositorio svn en la Unidad de Desarrollo Informático de la universidad Nacional de Loja.....	39
5.1.2.1 Requerimientos del Servidor de Repositorios.....	40
5.1.3 Configuración del Servidor. ....	41
5.1.4 Configuración de Red .....	41
5.1.5 Configuración de Subversion .....	42
5.1.6 Instalación de Subversion.....	43
5.1.7 Instalación Mediante línea de código .....	43
5.1.7.1 Configuración.....	43
5.1.8 Instalación gráfica del repositorio Subversion .....	47
5.1.9 Instalación del servidor de apache .....	47
5.1.10 Configuración del archivo apache2.conf.....	48
5.1.11 Configuración del archivo apache2.conf sin restricciones de acceso a usuarios.....	56
5.1.12 Configuración del archivo apache2.conf con restricciones de acceso a usuarios.....	59
5.1.13 Creación de los usuarios para los repositorios en el archivo de configuración dav_svn.passwd .....	61
5.1.14 Creación de los usuarios para los repositorios en el archivo de configuración dav_svn.passwd como root .....	61
5.1.15 Configuración de acceso a usuarios por niveles de los repositorio .....	65
5.1.16 Configuración del archivo <i>svn_access_control</i> para dar permiso de acceso para lectura y escritura por grupo a los repositorios .....	69
5.1.17 Configuración del archivo apache2.conf para la habilitación el css para acceder mediante http://. ....	77



5.2 Configuración del archivo apache2.conf para la habilitación el css para acceder mediante http:// .....	82
5.2.1 Requerimientos del Software y Hardware.....	82
5.2.1.1 Requerimientos del Hardware para la instalación SUBVERSION SERVIDOR.....	83
5.2.1.2 Requerimientos del Software para la instalación SUBVERSION SERVIDOR.....	78
5.2.1.3 Instalación de Subversion.....	85
5.2.1.4 Instale Apache2.....	92
5.2.1.5 Configuración de los Repositorios.....	94
5.2.1.6 Comandos en Subversion.....	109
5.2.1.7 Comandos para SVN.....	109
5.2.1.8 Comandos para SVNADMIN.....	111
5.3 Configuración del Repositorio subversión cliente.....	111
5.3.1 Requerimientos del Software y Hardware.....	112
5.3.1.1 Guía de Instalación.....	113
5.3.1.1.1 Iconos Sobreimpresionados.....	118
5.3.1.1.2 Menús Contextuales.....	118
5.3.1.1.3 Arrastrar y Soltar.....	119
5.3.1.1.4 Atajos Comunes.....	120
5.3.1.1.5 Autenticación.....	120
5.3.1.1.6 Maximizando Ventanas.....	121
5.3.1.2 Importando datos en un repositorio.....	121
5.3.1.2.1 Importar.....	121
5.3.1.2.2 Importar en el Sitio.....	122
5.3.1.2.3 Ficheros Especiales.....	123
5.3.1.3 Obteniendo una copia de trabajo.....	123
5.3.1.3.1 Profundidad de Obtención.....	124
5.3.1.4 Confirmando sus cambios en el Repositorio.....	124
5.3.1.4.1 El diálogo de Confirmación.....	124
5.3.1.4.2 Lista de Cambios.....	125
5.3.1.4.3 Excluyendo ítems de la lista de Confirmación.....	125



5.3.1.4.4	Mensajes de Registro de Confirmación.....	126
5.3.1.4.5	Progreso de Confirmación.....	127
5.3.1.5	Actualice su Copia de Trabajo con los cambios de Otros.....	128
5.3.1.6	Resolviendo Conflictos.....	128
5.3.1.6.1	Conflictos Ficheros.....	129
5.3.1.6.2	Conflictos de árbol.....	130
5.3.1.7	Obteniendo Información del Estado.....	130
5.3.1.7.1	Iconos Sobreimpresionados.....	130
5.3.1.7.2	Columnas de TortoiseSVN en el Explorador de Windows.....	131
5.3.1.7.3	Estado Local y Remoto.....	131
5.3.1.7.4	Viendo Diferencias.....	132
5.3.1.8	Lista de Cambios.....	132
5.3.1.9	Diálogo de Registro de Revisiones.....	133
5.3.1.9.1	Invocando el diálogo de Registro de Revisiones.....	134
5.3.1.10	Exportando una Copia de Trabajo de Subversion.....	135
5.3.1.10.1	Exportando Ficheros Sueltos.....	136
5.3.1.10.2	Eliminando una copia de trabajo del control de versiones.....	136
5.3.1.11	Tutorial de RapidSVN.....	137
5.3.1.11.1	Instalación.....	137
5.3.1.11.2	Configuración de las Preferencias.....	138
5.3.1.11.3	Deshacer una edición local (a través de revertir).....	138
5.3.1.11.4	Añadir un directorio (a través de agregar recursiva).....	139
5.3.1.11.5	Añadir un directorio a través de importación.....	139
5.4	Pruebas y Validación de Resultados.....	142
5.4.1	Análisis de Resultados.....	142
6	Valoración Técnica-Económica-Ambiental.....	151
7	Conclusiones.....	154
8	Recomendaciones.....	155
9	Bibliografías y referencias.....	156
10	Anexos.....	158
10.1	Anexo 1 Anteproyecto de Tesis.....	158
10.2	Anexo 2 Glosario de Términos.....	243
10.3	Anexo 3 Modelo de la Encuesta.....	245





10.4	Anexo 4 Certificaciones.....	246
------	------------------------------	-----



## ÍNDICE DE FIGURAS

### **FIGURAS**

Figura 3.1.	Ventana Inicial de Ubuntu 9.04.....	9
Figura 3.2.	Seleccionar el idioma de Ubuntu 9.04.....	9
Figura 3.3.	Elección de la hora de Ubuntu 9.04.....	10
Figura 3.4.	Elección del teclado.....	11
Figura 3.5.	Datos del Usuario .....	12
Figura 3.6.	Instalar o desinstalar programas .....	12
Figura 3.7.	Instalar o desinstalar programas gestor de paquetes Synaptic .....	12
Figura 3.8	Vista de paquetes disponibles para la instalación o desinstalación.....	13
Figura 3.9	Selección de paquetes para la instalación.....	13
Figura 3.10	Vista de paquetes necesarios para inicio de la instalación.....	14
Figura 3.11	Vista de paquetes seleccionados para la instalación.....	14
Figura 3.12	Inicio de la instalación de los paquetes.....	15
Figura 3.13	13 Proceso de descarga de los paquetes.....	15
Figura 3.14	Proceso de instalación de los paquetes.....	16
Figura 3.15	Proceso de Finalización de la instalación de los paquetes.....	16
Figura 3.3.1	Árbol de ficheros del repositorio.....	24
Figura 3.4.1	Arquitectura de subversión.....	29
Figura 5.1	Diagrama de ciclo de vida de Subversion .....	40
Figura 5.2	Requerimientos del Servidor de Repositorios.....	40
Figura 5.3	Configuración del servidor apache2.....	49
Figura 5.4	Configuración del servidor comando vim.....	49
Figura 5.5	Archivo del servidor apache.conf.....	55
Figura 5.6	Configuración del repositorio sin restricciones.....	52
Figura 5.7	Reiniciar el servicio de apache. ....	58
Figura 5.8	Terminación del proceso reiniciar el servicio de apache.....	58
Figura 5.9	Página de inicio del servidor apache.....	59
Figura 5.10	Configuración del repositorio con restricciones.....	60
Figura 5.11	Archivos de configuración del apache2.....	62
Figura 5.12	Configuración del repositorio con restricciones dav_svn.passwd.....	63
Figura 5.13	Vista Archivo apache2 con restricciones svn_access_control.....	73
Figura 5.14	Configuración del repositorio con restricciones svn_access_control....	74
Figura 5.15	Vista protocolo http de apache para acceder al repositorio SVN.....	75
Figura 5.16	Vista desde el protocolo http de apache.....	76



Figura 5.17	Vista desde el protocolo http de apache acceso correcto.....	76
Figura 5.18	Vista desde el protocolo http de apache archivos Subversionados...	77
Figura 5.19	Configuración del repositorio http css.....	78
Figura 5.20	Configuración del repositorio http.....	79
Figura 5.21	Vista http del repositorio utilizando css.....	75
Figura 5.22	Vista desde el protocolo http de apache.....	81
Figura 5.23	Vista desde http.....	81
Figura 5.2.1	Vista Ubuntu 9.04.....	83
Figura 5.2.2	Vista Instalar SVN.....	84
Figura 5.2.3	Vista abrir Gestor de paquetes synaptic.....	84
Figura 5.2.4	Vista Mensaje de Información abrir Gestor de paquetes synaptic.....	85
Figura 5.2.5	Vista buscar paquetes Subversión.....	85
Figura 5.2.6	Vista Seleccionar paquetes Subversion.....	86
Figura 5.2.7	Vista Instalación paquetes Subversion.....	86
Figura 5.2.8	Vista Selección de paquetes Subversion.....	87
Figura 5.2.9	Vista Aplicar paquetes Subversion.....	87
Figura 5.2.10	Vista Aceptar paquetes Subversion.....	88
Figura 5.2.11	Vista Descarga de paquetes Subversion.....	88
Figura 5.2.12	Vista proceso de descarga de paquetes Subversion.....	89
Figura 5.2.13	Vista descarga detallada de los paquetes Subversion.....	89
Figura 5.2.14	Vista Instalando paquetes Subversion.....	90
Figura 5.2.15	Vista proceso de instalación de paquetes Subversion.....	90
Figura 5.2.16	Vista proceso de finalización del proceso de instalación de paquetes Subversion.....	91
Figura 5.2.17	Vista proceso de marcación de paquetes apache2.....	91
Figura 5.2.18	Vista proceso de descarga de paquetes apache2.....	86
Figura 5.2.19	Vista proceso de descarga de paquetes apache2 marcados.....	92
Figura 5.2.20	Vista proceso de aplicar los paquetes para la descarga de apache2...	92
Figura 5.2.21	Vista proceso de aplicar los cambios de los paquetes apache2.....	93
Figura 5.2.22	Vista proceso de descarga de los paquetes de apache2.....	94
Figura 5.2.23	Vista proceso de finalizar los cambios de los paquetes apache2.....	94
Figura 5.2.24	Vista abrir el terminal de Ubuntu.....	90
Figura 5.2.25	Vista terminal de Ubuntu.....	95
Figura 5.2.26	Vista Archivo de configuración de apache2 con Subversion.....	96
Figura 5.2.27	Vista Archivo de configuración de apache2 con Subversion.....	96



Figura 5.2.28	Vista reinicio del apache2.....	97
Figura 5.2.29	Vista finalización del reinicio del apache2.....	98
Figura 5.2.30	Vista archivos del servidor apache2.....	98
Figura 5.2.31	Vista verificación del servidor apache2 mediante http://.....	99
Figura 5.2.32	Vista Habilitamos el css apache2.....	100
Figura 5.2.33	Vista Habilitar el css apache2.....	100
Figura 5.2.34	Vista habilitar css en el servidor.....	101
Figura 5.2.35	Vista Crear repositorio Subversion.....	101
Figura 5.2.36	Vista habilitar permiso del repositorios.....	102
Figura 5.2.37	Crear usuario y clave de acceso.....	103
Figura 5.2.38	Vista actualizar Usuario de svn.....	103
Figura 5.2.39	Vista Actualizar Archivo de configuración para acceso de control.....	104
Figura 5.2.40	Vista Editar Archivo svn_access_control.....	104
Figura 5.2.41	Vista Administrar Archivo acceso_control.....	104
Figura 5.2.42	Vista Archivo acceso_control.....	105
Figura 5.2.43	Vista reiniciar El servidor apache2.....	106
Figura 5.2.44	Vista reiniciar El servidor a2enmod dav.....	106
Figura 5.2.45	Vista reiniciar El servidor apache2 force-reload.....	107
Figura 5.2.46	Vista finalizando el reinicio de los servicios apache2.....	107
Figura 5.2.47	Vista inicio del servidor de apache2.....	107
Figura 5.2.48	Vista de un repositorio Subversion desde el servidor de apache2.....	108
Figura 5.2.49	Vista de un repositorio Subversion desde el servidor de apache2 mediante acceso por usuario. (Petición de la clave de acceso).....	108
Figura 5.2.50	Vista de un repositorio Subversion accediendo al proyecto con css.....	109
Figura 5.3.1	Instalador del Cliente TortoiseSVN.....	113
Figura 5.3.2	Instalación de TortoiseSVN.....	114
Figura 5.3.3	Instalación de Subversion acepta la Licencia .....	114
Figura 5.3.4	Instalación de SubversionSVN aceptar Términos .....	114
Figura 5.3.5	Instalación de SubversionSVN paquetes .....	115
Figura 5.3.6	Instalación de SubversionSVN instalación .....	115
Figura 5.3.7	Finalizando la Instalación de SubversionSVN.....	116
Figura 5.3.8	Paquete de idioma español SubversionSVN.....	116
Figura 5.3.9	Instalación del lenguaje Español.....	117
Figura 5.3.10	Terminación de la instalación del lenguaje Español.....	117
Figura 5.3.11	Explorador mostrando iconos sobreimpresionados.....	118
Figura 5.3.12	Menú contextual para un directorio bajo el control de versiones.....	118



Figura 5.3.13	Menú archivo del explorador para un acceso directo en una carpeta versionada.....	119
Figura 5.3.14	Menú de arrastre con el botón derecho para un directorio bajo el control de versiones.....	119
Figura 5.3.15	Diálogo de autenticación.....	120
Figura 5.3.16	El diálogo Importar.....	122
Figura 5.3.17	El diálogo Obtener.....	123
Figura 5.3.18	El diálogo de Confirmación.....	125
Figura 5.3.19	El corrector ortográfico del diálogo de Confirmación.....	126
Figura 5.3.20	El diálogo Progreso mostrando el progreso de una confirmación.....	127
Figura 5.3.21	Diálogo de progreso mostrando una actualización terminada.....	128
Figura 5.3.22	Comprobar modificaciones.....	131
Figura 5.3.23	Diálogo de confirmación con listas de cambios.....	133
Figura 5.3.24	El diálogo de Registro de revisiones.....	134
Figura 5.3.25	Diálogo de exportar.....	135
Figura 5.3.26	Tutorial de RapidSVN.....	137
Figura 5.3.27	Tutorial de RapidSVN Preferencias.....	138
Figura 5.3.28	Tutorial de RapidSVN.....	138
Figura 5.3.29	Tutorial de RapidSVN Importar.....	139
Figura 5.3.30	Tutorial de RapidSVN copia de trabajo.....	140
Figura 5.3.31	Tutorial de RapidSVN actualizar.....	141
Figura 5.3.32	Tutorial de RapidSVN fusionar.....	142
Figura 5.4.1	Resultado Pregunta 1.....	146
Figura 5.4.2	Resultado Pregunta 2.....	148
Figura 5.4.3	Resultado Pregunta 3.....	149
Figura 5.4.4	Resultado Pregunta 4.....	150



## ÍNDICE DE TABLAS

Tabla 5.1.	Hardware para el servidor del repositorio.....	41
Tabla 5.4.1.	Resultados Pregunta 1.....	146
Tabla 5.4.2.	Resultados pregunta 2.....	147
Tabla 5.4.3	Resultados pregunta 3.....	148
Tabla 5.4.4.	Resultados pregunta 4.....	149
Tabla 6.1.	Recursos Humanos.....	151
Tabla 6.2	Recursos Técnicos Hardware.....	152
Tabla 6.3	Recursos Técnicos Software.....	152
Tabla 6.4	Recursos Materiales .....	153
Tabla 6.5	Total Inversión .....	153



## 1. INTRODUCCIÓN

El presente trabajo de tesis titulado, **“CONFIGURACIÓN E IMPLEMENTACIÓN DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMÁTICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA,**

En su parte estructural la presente investigación consta de: RESUMEN, que es una recapitulación de las partes significativas de la investigación; INTRODUCCIÓN, en donde se expone la importancia de la realización de la implementación de un sistema de control de versiones, el aporte que deja a la Unidad de Desarrollo Informático de la carrera con el presente estudio y una breve síntesis del contenido de la investigación.

El mismo surgió de la necesidad de conocer los métodos, técnicas de desarrollo, para que facilite un mejor control de versiones de los programas en desarrollo, optimizando así tiempo, recursos e incentivando a los estudiantes a que formen parte de la misma.

El desarrollo del trabajo de investigación se fundamenta de acuerdo al objeto de transformación del módulo el cual se denomina: “Capacidad para planificar, organizar, dirigir y controlar recursos informáticos de desarrollo”.

La Universidad Nacional de Loja cuenta con un modelo pedagógico denominado: Sistema Académico Modular por Objetos de Transformación SAMOT, el cual se caracteriza por la integración de las funciones universitarias, de docencia, investigación y vinculación con la colectividad, a través de la investigación. El Área de Energía, Las Industrias y los Recursos Naturales no Renovables, y por ende, la carrera de “Ingeniería en Sistemas”, como partes actuantes de la Universidad Nacional de Loja orientan a sus estudiantes a la investigación científica tecnológica.

Por ello, el interés de realizar este proyecto es contribuir a mejorar el rendimiento en ahorro de tiempo al momento de unir código de varios desarrolladores de software.



La Unidad de Desarrollo Informático del A.E.I.R.N.N.R. de la Universidad Nacional de Loja mediante la implementación y configuración de un repositorio SVN utilizando un servidor en Linux (Ubuntu 9.04).

El sistema operativo Linux (Ubuntu 9.04), nos ofrece varias posibilidades para poder realizar un control de acceso a nuestro repositorio mediante Subversion con la ayuda de apache que se acopla de manera eficiente y en Windows mediante los clientes como TortoiseSVN y RapidSvn en Linux.

El sistema operativo Ubuntu es realmente un Debíán lo cual lo hace basado en paquetes DEBs y es muy fácil de actualizar y mantener. Ubuntu puede ser instalado como estación de trabajo o como servidor.

El Sistema de control de versiones es el arte de manejar cambios históricos de la información. Ha sido desde siempre una herramienta crítica para los programadores, quienes típicamente emplean su tiempo haciendo pequeños cambios al software y luego deshaciendo o comprobando esos cambios al día siguiente. Imagine un equipo de estos programadores trabajando concurrentemente y quizás también simultáneamente en los mismos ficheros.

Para el desarrollo de la presente tesis se aplicó una metodología acorde a los requerimientos básicos, entre ellos el método científico, inductivo – deductivo, analítico e investigativo.

Seguidamente se realizó la REVISIÓN EN LA WEB en la que consta la recopilación bibliográfica de los referentes teóricos sobre el análisis, sus diferentes métodos hasta el informe final, así mismo se expone los antecedentes y generalidades de la Subversion en el estudio; MATERIALES Y MÉTODOS, que resalta los materiales, métodos, técnicas y procedimientos que se utilizaron durante la investigación.

Finalmente se elaboraron las respectivas CONCLUSIONES Y RECOMENDACIONES, donde se describe los aspectos más relevantes obtenidos durante el proceso de análisis e implementación con el objeto de que los directivos tomen las decisiones adecuadas para el mejoramiento de la institución; BIBLIOGRAFÍA, en la que se





expone el material bibliográfico y las fuentes de información utilizados en el desarrollo del presente trabajo y finalmente constan los ANEXOS.

## **2 . METODOLOGÍA**

La Metodología es el conjunto de métodos empleados para el desarrollo de sistemas automatizados.

Para el desarrollo del presente proyecto, se aplicó diferentes métodos investigativos, herramientas técnicas que facilitaron la obtención de la información con un orden secuencial y lógico de sus partes, con el fin de cumplir los objetivos planteados en la investigación.

### **2.1 Metodología Usada**

En el presente capítulo se describirá las metodologías aplicadas para la elaboración del proyecto. La metodología que nos da una visión de cómo se ha desarrollado el proceso investigativo.

La investigación fue realizada mediante tareas que me permitieron obtener los mecanismos suficientes para los cumplimientos de los objetivos y de esta manera justifico el uso de estas herramientas y el beneficio de los estudiantes y el departamento de la Unidad de Desarrollo Informático de la carrera de Ingeniería en Sistemas para la optimización de recursos en la parte del desarrollo de sistemas informáticos.

### **2.2 Métodos Utilizados**

#### **2.2.1 El método Sintético.-**

Este método me permitió en primera instancia buscar en internet (comunidades) relacionadas y acumular gran cantidad de información del tema y así reordenar las ideas de lo investigado, ya que toda información no es verídica hasta que se compruebe su veracidad, porque está fuera del ámbito de la investigación planteada, me permitió conocer como es el funcionamiento real de los repositorios de Subversion utilizando svn, el cual sirvió para presentar de manera resumida y clara los



resultados de todo el proceso investigativo de la Subversion y mediante los resultados obtenidos poder llegar a conclusiones y plantar las respectivas recomendaciones.

### 2.3. Métodos e Instrumentos

Fue el primer acercamiento, con el objeto de conocer y estudiar directamente el procedimiento del desarrollo de programas y del código fuente que respaldan las operaciones que se realizan en el desarrollo del proyecto o proyectos que se realizan en nuestro medio.

- **Observación Indirecta.-** Consiste en tomar datos del sujeto(s) a medida que los hechos se suscitan ante los ojos del observador, quien desde luego podría tener algún entrenamiento a propósito de esa actividad.
- **Observación Directa.-** Se caracteriza por la interrelación que se da entre el investigador y los sujetos de los cuales se habrán de obtener ciertos datos. En ocasiones el investigador se pone en contacto personalmente con el hecho o fenómeno que trata de investigar.
- **El Método Científico** (es un proceso destinado a explicar fenómenos, establecer relaciones entre los hechos y enunciar leyes que expliquen los fenómenos físicos del mundo y permitan obtener, con estos conocimientos, aplicaciones útiles al hombre) el cual me ha permitido desarrollar algunas tareas planificadas. Este método me permitió a través de software libre y hardware vinculado a la Subversion realizar pruebas que me llevaron a definir con exactitud los mecanismos para la configuración final de:
  - Servidor de repositorios svn en Linux (Ubuntu 9.04).
  - Software libre de Subversion svn para uso de los usuarios (Clientes).
  - Permisos a sus respectivos repositorios Subversionados para uso de usuarios (Clientes).
  - Comunicación, entre cliente servidor.

Se lo utilizó como orientador general del proceso investigativo, siguiendo procedimientos lógicos relacionando las actividades internas y externas que se suscitan día a día en la fase del desarrollo de programas utilizando "SUBVERSION" de tal forma que se pueda abordar la realidad desde una perspectiva muy objetiva e imparcial y una vez estudiados y analizados los procesos del desarrollo en la parte



estudiante de la carrera de Ingeniería en Sistemas pude plantear alternativas de solución al problema.

### **Recolección bibliográfica:**

Esta técnica ayudó a recopilar toda la información tesis, así como el Internet para afianzar los conceptos acerca del tema en estudio.

## **3. FUNDAMENTACIÓN TEÓRICA**

En este capítulo nos concentraremos en la parte histórica de GNU Linux y algunas distribuciones disponibles y porque razón preferí a UBUNTU.

### **3.1 Que es GNU( Linux )**

Linux, es un sistema operativo multitarea, se pronuncia de la misma manera en todos los idiomas del mundo, como el idioma Español, y no 'lay-nux', es un sistema operativo (así como lo es Windows, Solaris, Mac OS X); fue creado por Linus Torvalds en 1991 como una alternativa a los sistemas Unix de la época.

En la actualidad Linux ha tenido un gran crecimiento en sus distribuciones tanto en la facilidad de uso y una interface amigable al usuario final.

Algunas características de Linux:

- Su licencia GPL, garantiza que permanecerá LIBRE, lo que significa que los documentos que produzca en este siempre estarán disponibles y no son objeto de políticas corporativas ni decisiones que usted no controla.
- Acceso a los códigos fuentes y derecho a modificación. Esto ayuda la participación de miles de programadores a mejorar y si es necesario modificar el software. Además es muy útil al momento de eliminar errores.
- GNU/Linux es realmente un sistema operativo multiusuario, multitarea que permite que múltiples usuarios trabajen con múltiples aplicaciones.



- Extremadamente estable, robusto, escalable y seguro puede ser actualizado sin necesidad de reiniciar el equipo.
- El entorno es completamente gráfico para su fácil integración de usuarios y requerimientos mínimos en hardware.

### **3.1.1 Distribuciones GNU (Linux) Linux**

Una distribución no es otra cosa, que una recopilación de programas y ficheros, organizados y preparados para su instalación. Estas distribuciones se pueden obtener a través de Internet, o comprando los CDs de las mismas, los cuales contendrán todo lo necesario para instalar un sistema Linux bastante completo y en la mayoría de los casos un programa de instalación que nos ayudara en la tarea de una primera instalación. Casi todos los principales distribuidores de Linux, ofrecen la posibilidad de bajarse sus distribuciones, vía FTP (sin cargo alguno).

Existen muchas y variadas distribuciones creadas por diferentes empresas y organizaciones a unos precios bastantes accesibles (si se compran los CDs, en vez de bajársela vía FTP).

A continuación algunas de las distribuciones más importantes de Linux (aunque no las únicas).

#### **UBUNTU**

Distribución basada en Debian, con lo que esto conlleva y centrada en el usuario final y facilidad de uso. Muy popular y con mucho soporte en la comunidad. El entorno de escritorio por defecto es GNOME.

#### **REDHAT ENTERPRISE**

Esta es una distribución que tiene muy buena calidad, contenidos y soporte a los usuarios por parte de la empresa que la distribuye. Es necesario el pago de una licencia de soporte. Enfocada a empresas.



## **FEDORA**

Esta es una distribución patrocinada por RedHat y soportada por la comunidad. Fácil de instalar y buena calidad.

## **DEBIAN**

Otra distribución con muy buena calidad. El proceso de instalación es quizás un poco más complicado, pero sin mayores problemas. Gran estabilidad con los últimos avances.

Debian es un sistema operativo (S.O.) libre, para su computadora. El sistema operativo es el conjunto de programas básicos y utilidades que hacen que funcione su computadora.

Debian utiliza el núcleo Linux (el corazón del sistema operativo), pero la mayor parte de las herramientas básicas vienen del Proyecto GNU; de ahí el nombre GNU/Linux.

## **OpenSuSE**

Otra de las grandes. Fácil de instalar. Versión libre de la distribución comercial SuSE. Es una de las más conocidas distribuciones Linux existentes a nivel mundial, se basó en sus orígenes en Slackware. Entre las principales virtudes de esta distribución se encuentra el que sea una de las más sencillas de instalar y administrar, ya que cuenta con varios asistentes gráficos para completar diversas tareas en especial por su gran herramienta de instalación y configuración.

## **SuSE LINUX ENTERPRISE**

Otra de las grandes. Muy buena calidad, contenidos y soporte a los usuarios por parte de la empresa que la distribuye, Novell. Es necesario el pago de una licencia de soporte. Enfocada a empresas.

## **SLACKWARE**

Esta distribución es de las primeras que existió. Tuvo un periodo en el cual no se actualizo muy a menudo.

## **GENTOO**

Esta distribución es una de las únicas que incorporaron un concepto totalmente nuevo en Linux. Es un sistema inspirado en BSD-ports. Se puede compilar/optimizar vuestro



sistema completamente desde cero. No es recomendable adentrarse en esta distribución sin una buena conexión a internet, un ordenador medianamente potente (si quiere terminar de compilar en un tiempo prudencial) y cierta experiencia en sistemas Unix.

## **KUBUNTU**

Distribución basada en Ubuntu, con lo que esto conlleva y centrada en el usuario final y facilidad de uso. La gran diferencia con Ubuntu es que el entorno de escritorio por defecto es KDE.

## **MANDRIVA**

Esta distribución fue creada en 1998 con el objetivo de acercar el uso de Linux a todos los usuarios, en un principio se llamó Mandrake Linux. Facilidad de uso para todos los usuarios.

### **Porque instalar Ubuntu?**

Por razones varias, entre ellas posee un livecd, la ventaja de un livecd es que podemos probar sin tener primero que instalar, fácil adquisición es libre, viene en un solo cd lo que hace menos costo de copia a diferencia de debian que viene en 14 cds o 2 dvds, Ubuntu es un debían el cual están basados en paquetes debs, fácil de actualizar e instalar como estación de trabajo o servidor.

### **3.1.2 Instalar Linux (Linux Ubuntu)**

Instalación Distro (Linux Ubuntu) por razones de fácil intuitivo manejo de las instalaciones, no importa que distro encuentres al final todos son iguales al fondo y solo cambia la forma.

La primera, es la imagen (Figura 3.1) del escritorio GNOME utilizado por Ubuntu recién arrancado desde el live-cd, donde tenemos un "bonito" icono, para iniciar la instalación basta con hacer doble click sobre él para que se enlace el programa instalador.



Figura 3.1: Ventana Inicial de Ubuntu 9.04.

1. .- La segunda imagen es la de elección del lenguaje de instalación (Figura 3.2)



Figura 3.2: Seleccionar el idioma de Ubuntu 9.04

2. Indicamos la localización, la hora y la fecha (Figura 3.3)

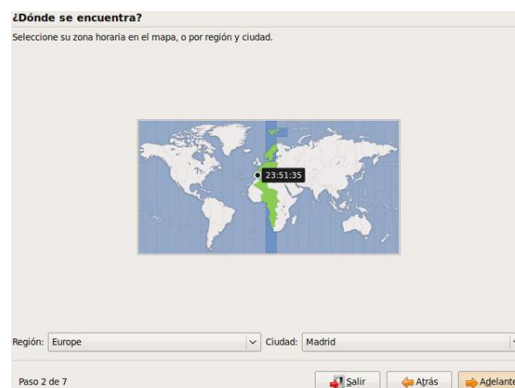


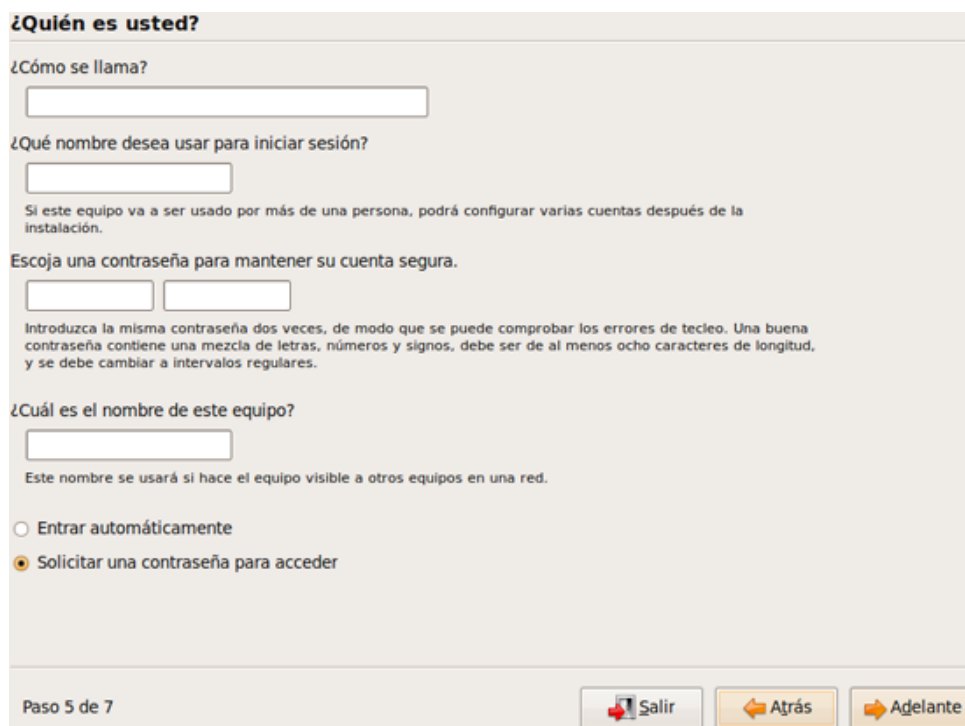
Figura 3.3: Elección de la hora de Ubuntu 9.04

### 3. Distribución de nuestro teclado (Figura. 3.4)



**Figura 3.4: Elección del teclado**

### 4. Proporcionamos los datos identificativos clave y login (Figura. 3.5)



**Figura 3.5: Datos del Usuario**

### 5. En este paso nos brinda tres opciones

- 1) Utilizar el disco completo: la más sencilla de realizar, el instalador se encarga de borrar y particional automáticamente.





2) Utilizar el espacio libre más grande: el instalador configura automáticamente las particiones, pero no usa todo el disco, sólo aquel espacio que no esté particionado y además sea el de mayor tamaño, en el caso de que hay más de uno.

3) Editar todas las particiones: la fase más compleja y para "expertos". En el caso de que escojamos esta opción, todo se realiza a voluntad del usuario, pudiendo escoger o crear entre otras cosas diversas particiones para albergar los diferentes archivos y donde seleccionamos el tamaño de la partición y/ o particiones así como el sistema de ficheros que deseemos para nuestro sistema.

Al final nos pide sacar el cd de instalación y listo tenemos nuestro sistema operativo Ubuntu para usarlo.

## **3.2 Instalación**

### **3.2.1 Instalar Paquetes GNU (Linux Ubuntu)**

Las distribuciones basadas en Debian, como es el caso de Ubuntu, utilizan los paquetes del tipo DEB.

El instalador de aplicaciones en Ubuntu y la mayoría de las distros basadas en Debian hoy en día es Synaptic. Para lanzar el Synaptic, solo debemos dirigirnos al menú del Sistema->Administración->Gestor de Paquetes. Se le presentará la aplicación que vemos en la figuras 3.8, 3.9 y 3.10.

Como la imagen nos muestra es muy fácil agregar y eliminar paquetes esta aplicación nos ayuda a mantener nuestro sistema al día, mediante simplemente buscar, elegir y darle al botón aplicar figuras 3.11, 3.12, 3.13, 3.14, 3.15.

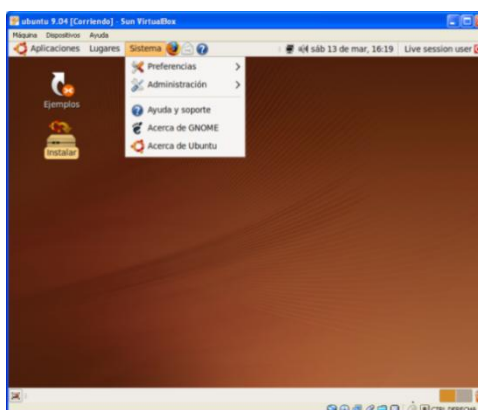
### **3.2.2 Eliminar Paquetes GNU (Linux Ubuntu)**

Para la eliminación de programas en Linux Ubuntu se realiza de la siguiente manera Synaptic (Figura 3.6, 3.7, 3.14, 3.15), desinstalar son tareas sumamente fáciles y con muy poco esfuerzo. Simplemente seleccionamos el paquete que deseamos desinstalar, este paquete lo encontramos utilizando la característica de la herramienta de buscar que nos provee el Synaptic, y cuando lo ubicamos lo

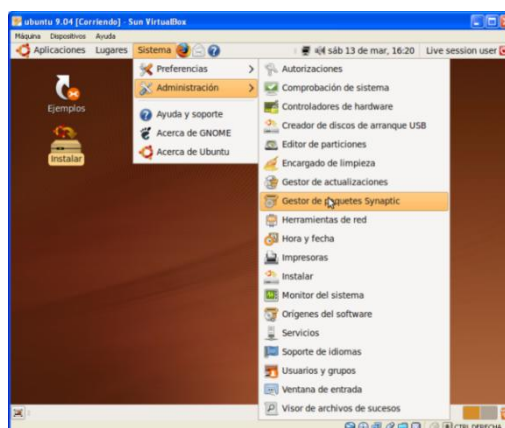
seleccionamos con el mouse y damos un click derecho, elegimos la opción marcar, lo que nos indica que si está instalado, y en el menú conceptual que nos presenta elegimos entre marcar para reinstalar, marcar para eliminar o marcar para eliminar completamente".

Luego de elegir los paquetes que deseamos eliminar, al igual que cuando instalamos un programa damos click sobre el botón de aplicar. Recuerda que puedes elegir entre algunos paquetes para instalar y otros para eliminar, simultáneamente. Cuando eliges un paquete para eliminar, el Synaptic te indicará si esta acción conlleva la eliminación de otros paquetes dependientes. Debes poner mucha atención, ya que eliminar ciertos paquetes y a veces instalar ciertos paquetes puede tornar tu sistema completamente inestable.

Otra consideración importante de instalar y desinstalar es que hay veces problemas de incompatibilidad entre librerías anteriores y que puede que todo el sistema se vuelva inestable.



**Figura 3.6: Instalar o desinstalar programas.**



**Figura 3.7: Instalar o desinstalar programas del gestor de paquetes Synaptic.**

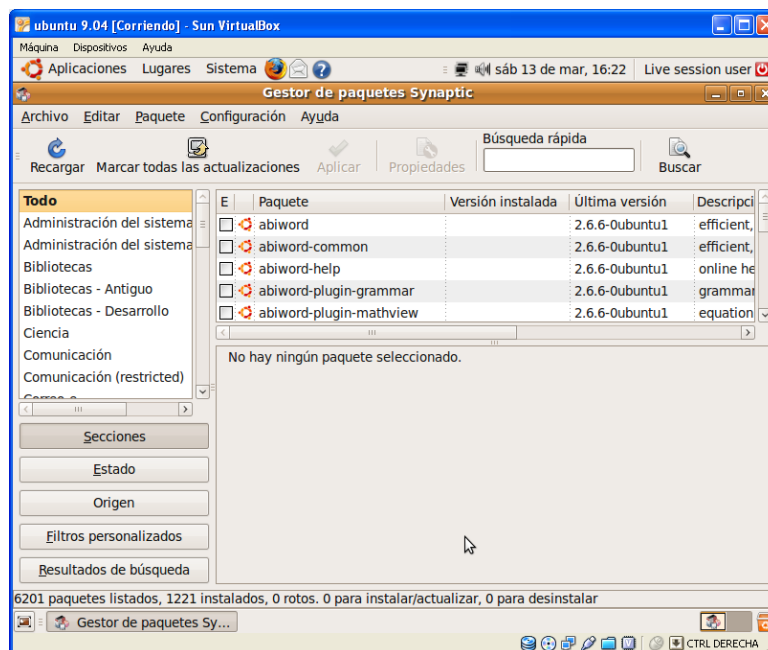


Figura 3.8: Vista de paquetes disponibles para la instalación o desinstalación.

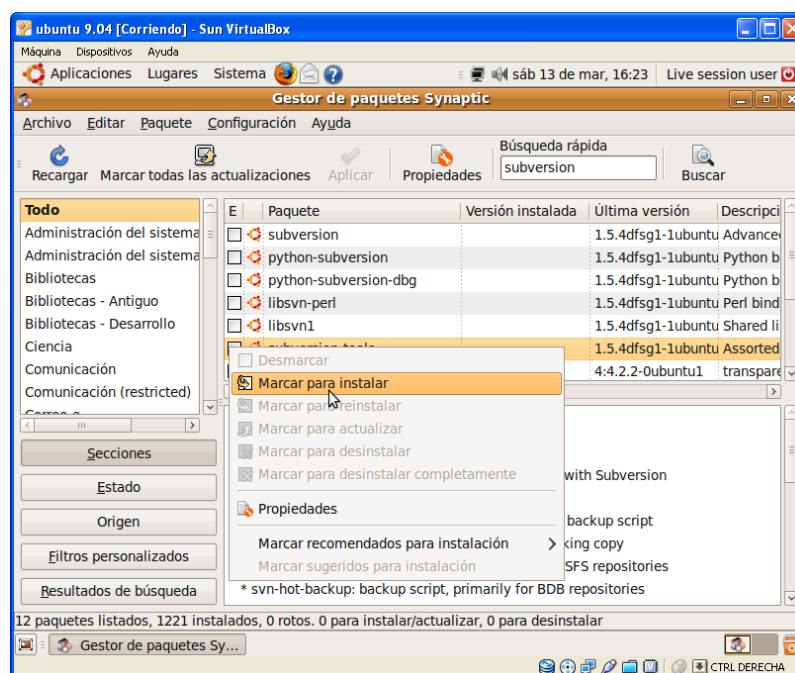


Figura 3.9: Selección de paquetes para la instalación

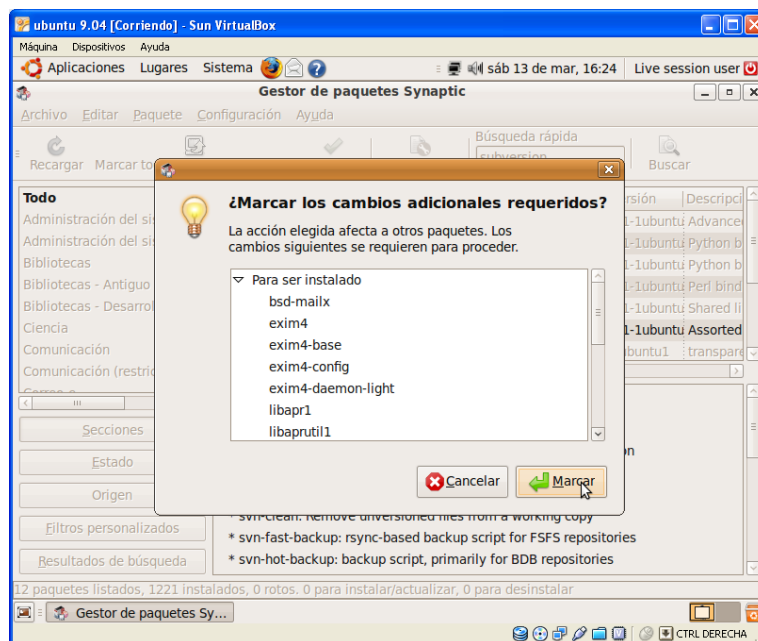


Figura 3.10: Vista de paquetes necesarios para inicio de la instalación.

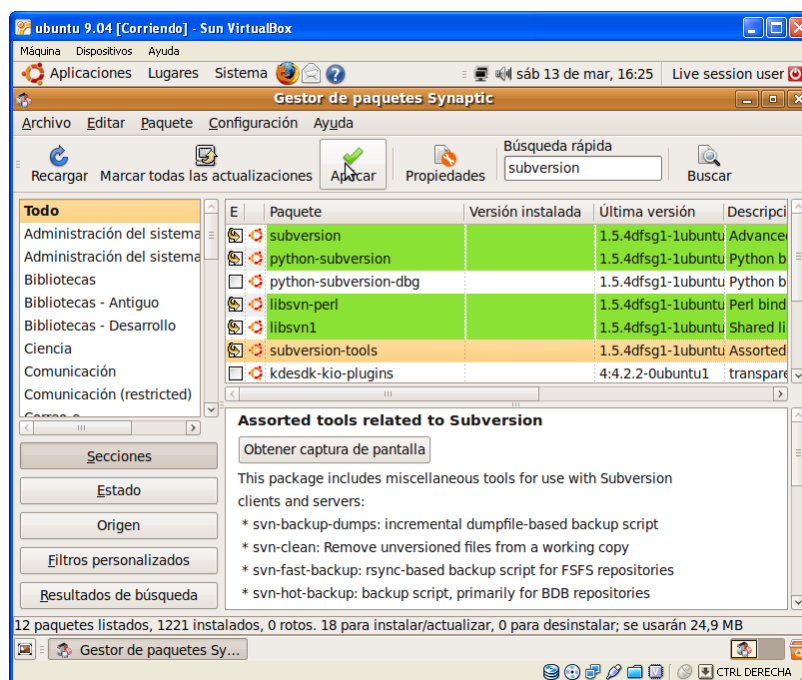


Figura. 3.11: Vista de paquetes seleccionados para la instalación.

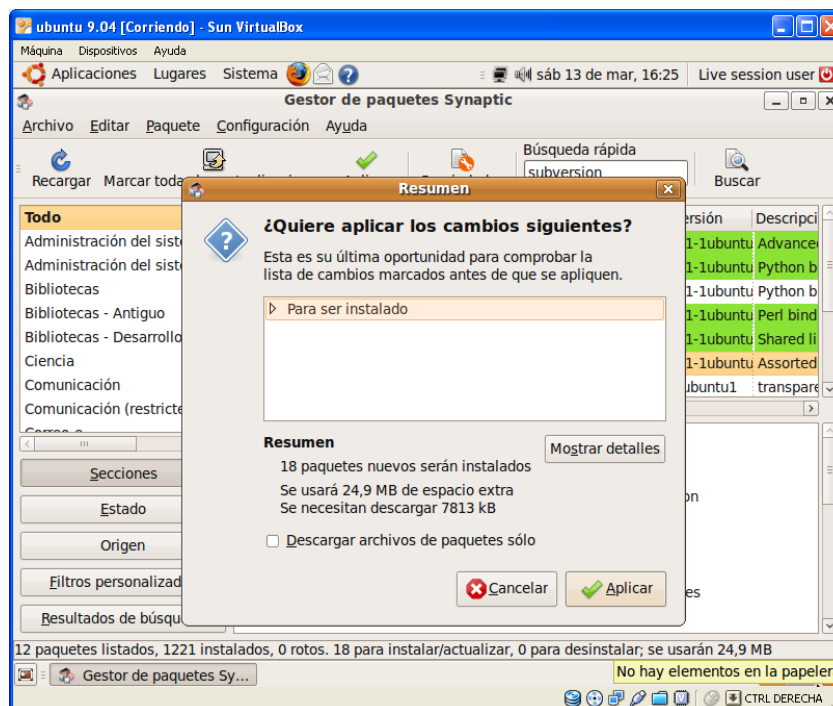


Figura 3.12: Inicio de la instalación de los paquetes.

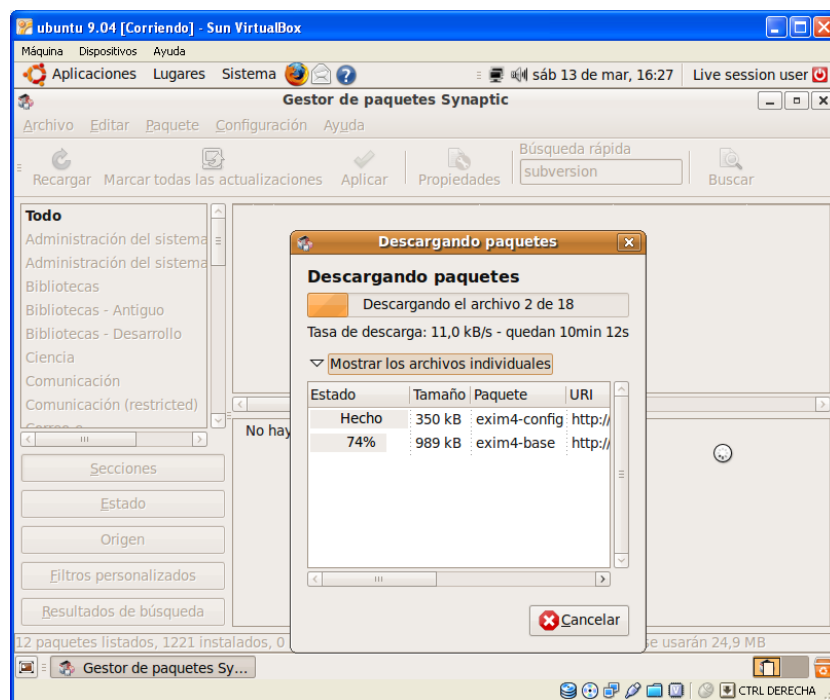


Figura. 3.13: Proceso de descarga de los paquetes.

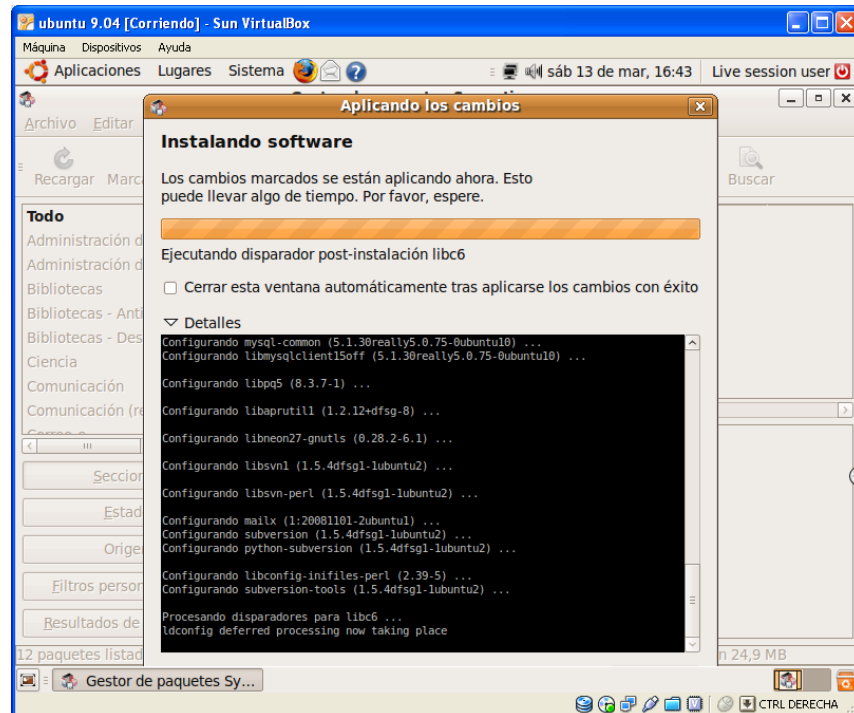


Figura 3.14: Proceso de instalación de los paquetes.

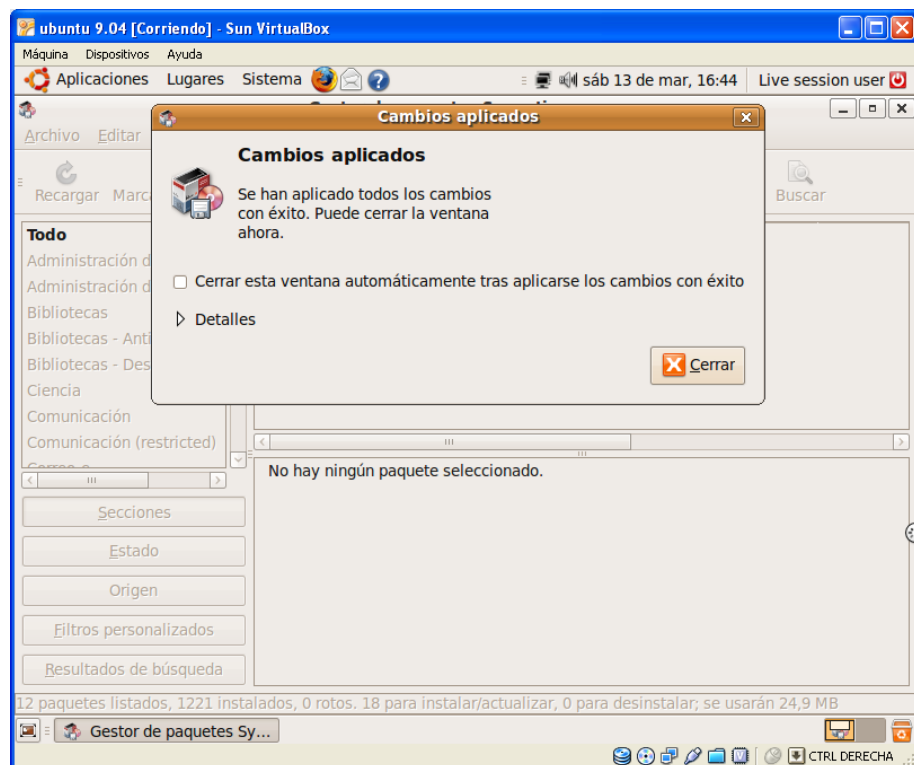


Figura 3.15: Proceso de Finalización de la instalación de los paquetes.



### 3.3 Control de Versiones

#### 3.3.1. Introducción

Una versión, revisión o edición de un producto, es el estado en el que se encuentra en un momento dado en su desarrollo o modificación. Se llama **control de versiones** a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo.

Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas (por ejemplo, para algún cliente específico).

El control de versiones se realiza principalmente en la industria informática para controlar las distintas versiones del **código fuente**. Sin embargo, los mismos conceptos son aplicables a otros ámbitos como documentos, imágenes, sitios web, etcétera.

Aunque un sistema de control de versiones puede realizarse de forma manual, es muy aconsejable disponer de herramientas que faciliten esta gestión (**CVS**, **Subversion**, **SourceSafe**, **Bazaar**, **Mercurial**, **git** etc.).

#### 3.3.2 Tipos Control de Versiones

##### 3.3.2.1 CVS

El Concurrent Versions System (CVS), es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros (código fuente principalmente) que forman un proyecto (de programa) y permite que distintos desarrolladores (potencialmente situados a gran distancia) colaboren. CVS se ha hecho popular en el mundo del software libre. Sus desarrolladores difunden el sistema bajo la licencia GPL.<sup>1</sup>

---

<sup>1</sup> [http://es.wikibooks.org/wiki/Tutorial\\_de\\_uso\\_de\\_CVS](http://es.wikibooks.org/wiki/Tutorial_de_uso_de_CVS)



## Desventajas CVS

Los archivos en el repositorio sobre la plataforma CVS no pueden ser renombrados, estos deben ser agregados con otro nombre y luego eliminados. El protocolo CVS no provee una manera de que los directorios puedan ser eliminados o renombrados, cada archivo en cada subdirectorio debe ser eliminado y re-agregado con el nuevo nombre. Soporte limitado para archivos Unicode con nombres de archivo no ASCII.

### 3.3.2.2 Microsoft Visual SourceSafe

**Microsoft Visual SourceSafe** (también conocido por sus siglas VSS) es una herramienta de Control de versiones que forma parte de Microsoft Visual Studio aunque está siendo sustituida por el Visual Studio Team Foundation Server.

## Desventajas VSS

La principal desventaja de Visual SourceSafe reside en el método de acceso a los archivos compartidos que constituyen su repositorio mediante el protocolo SMB que no impide que éstos sean manipulados de manera externa al producto por cualquier persona que tenga acceso al mismo, provocando corrupción de datos. Este mismo tipo de acceso a archivos compartidos provoca que en equipos de trabajo grandes, el acceso concurrente pueda ser particularmente lento.

### 3.3.2.3 El SourceSafe

Es configurable, permitiendo que un solo programador codifique el código fuente (recomendado) o que lo hagan varios. Las herramientas de gestión de diferencias para reunificar el código fuente modificado por varios programadores no son demasiado buenas comparadas con las de otros gestores de código fuente.





## Desventajas

- El SourceSafe es inestable cuando se suben ficheros binarios de gran tamaño, ya que espera solo ficheros de texto. Así que no vale para almacenar documentación, sólo código fuente.

### 3.3.2.4 Mercurial

Es un control de versiones distribuido (DCVS), y supone un gran paso adelante de otros sistemas como Subversion. Mercurial es descentralizado, rápido y sencillo. Puedes hacer branches sin miedo al merge.

Mercurial también es posible revertir cualquier fichero a cualquier revisión sin necesidad de conexión con el servidor (**distribuido**). Los programadores que se han visto en estas situaciones adversas (como proxies)

## Desventajas

- Requiere algo más de control de accesos
- Organizar estos repositorios, respecto a la organización, es un **problema** que comparten todos los sistemas de **control de versiones distribuidos**

### 3.3.2.5 Bazaar

Bazaar, el proyecto de un sistema de control de versiones **distribuido** patrocinado por Canonical Ltd., anunció así el lanzamiento de su versión 1.0. Según Mark Shuttleworth bazaar tiene ventajas sobre los SCM centralizados como subversion, cvs o source safe. Es una solución parecida a GIT. Bazaar está basado en Python, funciona desde la línea de comando o con una interface gráfica, incluye integración con herramientas como Eclipse y viene con más de 20 plugins.

## Desventajas

- Requiere algo control de actualizaciones
- Organizar estos repositorios, respecto a la organización, es un **problema** que comparten todos los sistemas de **control de versiones distribuidos**



### 3.3.2.6 GIT.

Es un software de control de versiones (**distribuido**) diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número archivos de código fuente.

#### Desventajas

- Requiere algo control de actualizaciones
- Organizar estos repositorios, respecto a la organización, es un **problema** que comparten todos los sistemas de **control de versiones distribuidos**

### 3.3.2.7 Subversion

**Subversion** es un software de sistema de control de versiones diseñado específicamente para reemplazar al popular **CVS**, el cual posee varias deficiencias. Es software libre bajo una licencia de tipo Apache y se le conoce también como **svn** por ser ese el nombre de la herramienta de línea de comandos.<sup>2</sup>

Una característica importante de Subversion es que, a diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.

Subversion puede acceder al repositorio a través de redes, lo que le permite ser usado por personas que se encuentran en distintos ordenadores. A cierto nivel, la capacidad para que varias personas puedan modificar y administrar el mismo conjunto de datos desde sus respectivas ubicaciones fomenta la colaboración. Se puede progresar más rápidamente sin un único conducto por el cual deban pasar todas las modificaciones. Y puesto que el trabajo se encuentra bajo el control de versiones, no hay razón para temer por que la calidad del mismo vaya a verse afectada por la pérdida de ese conducto único si se ha hecho un cambio incorrecto a los datos, simplemente se regresa al cambio anterior.

- **Ventajas**

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.

---

<sup>2</sup> <http://es.wikipedia.org/wiki/Subversion>



- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente.
- Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).
- Puede ser servido mediante Apache, sobre WebDAV. Esto permite que clientes WebDAV utilicen Subversion en forma transparente.
- Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto).
- Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.
- Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor provee a la hora de autenticar archivos (SQL, LDAP, PAM, etc.).

### 3.3.3 Clientes

Existen varias interfaces a Subversion, ya sea programas individuales como interfaces que lo integran en entornos de desarrollo.

- **TortoiseSVN**. Provee integración con el explorador de Windows. Es la interfaz más popular en este sistema operativo.
- **Subclipse**. "Plugin" que integra Subversion al entorno Eclipse.
- **Subversive**. "Plugin" alternativo para Eclipse.
- **Cervisia** Programa para interacción para linux, combinada con Quanta Plus puede llegar a ser muy eficaz.
- **Para mac**, pueden emplearse los interfaces *SvnX*, *RapidSVN* y *Zigversion*.
- **RapidSVN** también corre en Linux.
- **KDESVN**. Provee integración con el escritorio KDE, muy parecido en apariencia, funcionamiento, características a TortoiseSVN.



- **AnkhSVN "Plugin"** para Visual Studio para versiones 2002, 2003, 2005, 2008 y 2010, esta última en modo experimental.

### 3.3.3.1 Uso y reconocimiento

*Subversion es muy conocido en la comunidad de software libre y se utiliza en muchos proyectos, incluyendo la fundación del software de Apache, KDE, GNOME, Free Pascal, FreeBSD, GCC, Python, Django, Ruby, Mono, SourceForge.net, ExtJS y Tigris.org. El servicio Google Code también proporciona almacenamiento Subversion para sus proyectos de software libre. Los sistemas de BountySource lo utilizan exclusivamente. Codeplex ofrece acceso tanto para Subversión como para otros tipos de clientes. Subversión también está siendo adoptado en el mundo corporativo. En un informe 2007 de ForresterResearch, reconocía a Subversion como el único líder en la categoría de sistema de control de versiones.*

### 3.3.4 Características

#### 3.3.4.1 Características control de versiones.

Un sistema de control de versiones debe proporcionar:

- Mecanismo de almacenaje de los elementos que deba gestionar (ej. archivos de texto, imágenes, documentación...)
- Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos)
- Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto)

Aunque no es estrictamente necesario, suele ser muy útil la generación de informes con los cambios introducidos entre dos versiones, informes de estado, marcado con nombre identificativo de la versión de un conjunto de ficheros, etcétera.

### 3.3.5 Clasificación

La principal clasificación que se puede establecer está basada en el almacenamiento del código:



- **Centralizados:** existe un repositorio centralizado de todo el código, del cual es responsable un único usuario (o conjunto de ellos). Se facilitan las tareas administrativas a cambio de reducir la potencia y flexibilidad, pues todas las decisiones fuertes (como crear una nueva rama) necesitan la aprobación del responsable. Algunos ejemplos son **CVS** y **Subversion**.
- **Distribuidos:** se aumenta la capacidad de decisión distribuida. Esto da más flexibilidad pero puede dificultar bastante la sincronización. Funcionamiento.

Todos los sistemas de control de versiones se basan en disponer de un repositorio, que es el conjunto de información gestionada por el sistema. Este repositorio contiene el historial de versiones de todos los elementos gestionados.

Cada uno de los usuarios puede crearse una **copia local** duplicando el contenido del repositorio para permitir su uso. Es posible duplicar la última versión o cualquier versión almacenada en el historial. Este proceso se suele conocer como *checkout* o *desproteger*. Para modificar la copia local existen dos semánticas básicas:

**Exclusivos:** para poder realizar un cambio es necesario marcar en el repositorio el elemento que se desea modificar y el sistema se encargará de impedir que otro usuario pueda modificar dicho elemento.

**Colaborativos:** en el que cada usuario se descarga la copia la modifica y el sistema automáticamente mezcla las diversas modificaciones. El principal problema es la posible aparición de conflictos que deban ser solucionados manualmente o las posibles inconsistencias que surjan al modificar el mismo fichero por varias personas no coordinadas. Además, esta semántica no es apropiada para ficheros binarios.

Tras realizar la modificación es necesario actualizar el repositorio con los cambios realizados. Habitualmente este proceso se denomina **publicar, commit, check in o proteger**.

### 3.3.6 El Repositorio

Subversion es un sistema centralizado para compartir información. En su núcleo está un *repositorio*, que es un almacén central de datos. El repositorio almacena información en forma de un *árbol de ficheros*, una jerarquía típica de ficheros y directorios. Cualquier número de *clientes* se conectan al repositorio, y luego leen o escriben esos ficheros. Al escribir datos, el cliente hace que la información esté disponible para los otros; al leer los datos, el cliente recibe la información de los demás.

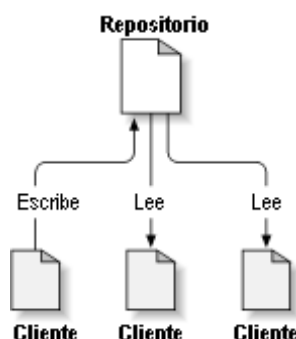


Figura 3.3.1: Árbol de ficheros del repositorio

### 3.3.7 Vocabulario Común

La terminología empleada puede variar de sistema a sistema, pero a continuación se describen algunos términos de uso común<sup>3</sup>.

#### Repositorio

El **repositorio** es el lugar en el que se almacenan los datos actualizados e históricos, a menudo en un servidor. A veces se le denomina depósito. Puede ser un sistema de archivos en un disco duro, un banco de datos, etc.

#### Módulo

Conjunto de directorios y/o archivos dentro del repositorio que pertenecen a un proyecto común.

---

<sup>3</sup> <http://tortoisesvn.tigris.org/>.



### **Rotular ("tag")**

Darle a alguna versión de cada uno de los ficheros del módulo en desarrollo en un momento preciso un nombre común ("etiqueta" o "rótulo") para asegurarse de reencontrar ese estado de desarrollo posteriormente bajo ese nombre. En la práctica se rotula a todos los archivos en un momento determinado. Para eso el módulo se "congela" durante el rotulado para imponer una versión coherente. Pero bajo ciertas circunstancias puede ser necesario utilizar versiones de algunos ficheros que no coinciden temporalmente con las de los otros ficheros del módulo.

### **Revisión ("versión")**

Una **revisión** es una versión determinada de un archivo.

### **Línea base ("Baseline")**

Una revisión aprobada de un documento o fichero fuente, a partir del cual se pueden realizar cambios subsiguientes.

### **Desplegar ("Check-out")<sup>4</sup>**

Un despliegue crea una copia de trabajo local desde el repositorio. Se puede especificar una revisión concreta, y por defecto se suele obtener la última

### **"Publicar" o "Enviar" ("commit", "check-in", "ci", "install", "submit")<sup>5</sup>**

Un **commit** sucede cuando una copia de los cambios hechos a una copia local es escrita o integrada sobre repositorio.

### **Conflicto**

---

<sup>4</sup> <http://picandocodigo.net/downloads/docs/subversion-presentacion-02.pdf>

<sup>5</sup> <http://picandocodigo.net/downloads/docs/subversion-presentacion-02.pdf>



Un conflicto ocurre en las siguientes circunstancias:

- Los usuarios **X** e **Y** despliegan versiones del archivo **A** en que las líneas **n1** hasta **n2** son comunes.
  - El usuario **X** envía cambios entre las líneas **n1 y n2** al archivo **A**.
  - El usuario **Y** no actualiza el archivo **A** tras el envío del usuario **X**.
  - El usuario **Y** realiza cambios entre las líneas **n1 y n2**.
  - El usuario **Y** intenta posteriormente enviar esos cambios al archivo **A**.

El sistema es incapaz de fusionar los cambios. El usuario **Y** debe **resolver** el conflicto combinando los cambios, o eligiendo uno de ellos para descartar el **otro**.

### **Resolver**

El acto de la intervención del usuario para atender un conflicto entre diferentes cambios al mismo documento. Cambio ("**change**", "**diff**")

Un cambio representa una modificación específica a un documento bajo control de versiones. La granularidad de la modificación considerada un cambio, varía entre diferentes sistemas de control de versiones.

### **Lista de cambios ("**changelist**", "**change set**", "**patch**")<sup>6</sup>**

En muchos sistemas de control de versiones con commits multi-cambio atómicos, una lista de cambios identifica el conjunto de cambios hechos en un único commit.

Esto también puede representar una vista secuencial del código fuente, permitiendo que el fuente sea examinado a partir de cualquier identificador de lista de cambios particular.

---

<sup>6</sup> [http://es.wikipedia.org/wiki/Control\\_de\\_versiones](http://es.wikipedia.org/wiki/Control_de_versiones)





### **Exportación ("export")<sup>7</sup>**

Una exportación es similar a un **check-out**, salvo porque crea un árbol de directorios limpio sin los metadatos de control de versiones presentes en la copia de trabajo.

Se utiliza a menudo de forma previa a la publicación de los contenidos, es decir cuando se quiere liberar al proyecto de la subversion.

### **Importación ("import")**

Una importación es la acción de copia un árbol de directorios local (que no es en ese momento una copia de trabajo) en el repositorio por primera vez.

### **Integración o fusión ("merge")**

Una **integración** o **fusión** une dos conjuntos de cambios sobre un fichero o un conjunto de ficheros en una revisión unificada de dicho fichero o ficheros.

### **Actualización ("sync" ó "update")**

Una **actualización** integra los cambios que han sido hechos en el repositorio (por ejemplo por otras personas) en la **copia de trabajo** local.

### **Copia de trabajo**

La **copia de trabajo** es la copia local de los ficheros de un repositorio, en un momento del tiempo o revisión específicos. Todo el trabajo realizado sobre los ficheros en un repositorio se realiza inicialmente sobre una copia de trabajo.

### **Congelar**

Significa permitir los últimos cambios (commits) para solucionar las fallas a resolver en una entrega (release) y suspender cualquier otro cambio antes de una entrega, con el fin de obtener una versión consistente. Si no se congela el repositorio, un desarrollador podría comenzar a resolver una falla cuya resolución no está prevista y cuya solución dé lugar a efectos colaterales imprevistos.

---

<sup>7</sup> [http://es.wikipedia.org/wiki/Control\\_de\\_versiones](http://es.wikipedia.org/wiki/Control_de_versiones)



### **3.4 Hardware y software para la implementación del servidor de repositorios svn, en la Unidad de Desarrollo Informático de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja**

#### **3.4.1 Introducción**

En la actualidad empresas, instituciones y comunidades de desarrollo han optado por la migración total o parcial de los repositorios de Subversion, con el propósito de reducir costos y recursos, para estos fines es necesario contar con hardware y software apropiado que permita la implementación de esta tecnología.

En este capítulo se abordará las características de software para el repositorio de subversión svn, hardware necesario, refiriéndonos al servidor del repositorio svn.

#### **Software seleccionado para la configuración e implementación del servidor svn**

Luego del análisis realizado de las diferentes herramientas para las Subversiones, se cree conveniente utilizar subversión para la implementación del repositorio svn en el servidor de la carrera de ingeniería en sistemas de la Universidad Nacional de Loja.

Subversion es un software completo de conectividad entre distintas tecnologías, de desarrollo, actúa en Linux y Windows provee todas las configuraciones que se espera para la autenticación de usuarios por medio del protocolo WebDAV y con ayuda del apache pueden ser accedidos utilizando las configuraciones del mismo.

## Arquitectura de Subversion

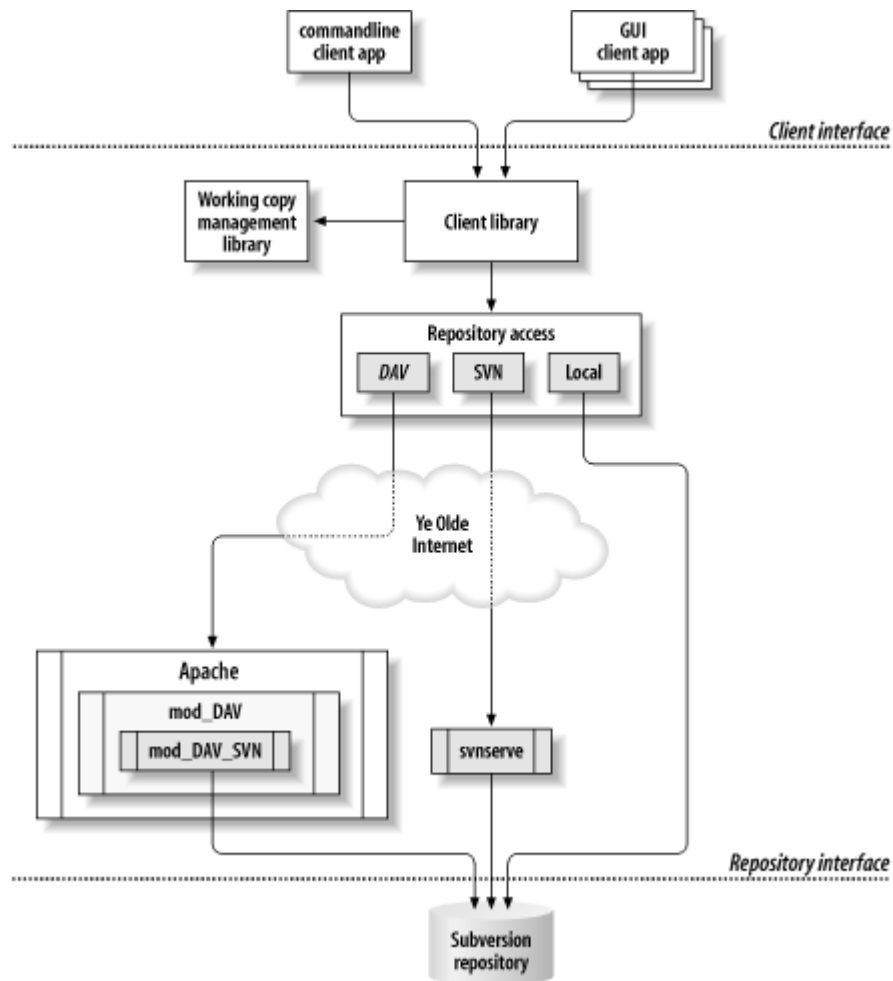


Figura 3.4.1: Arquitectura de subversión<sup>8</sup>

<sup>8</sup> <http://svnbook.spears.at/nightly/es/svn-ch-1-sect-4.html>



### 3.4.2 Servidor

Es una aplicación informática o programa que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes. Algunos servicios habituales son los servicios de archivos, que permiten a los usuarios almacenar y acceder a los archivos de una computadora y los servicios de aplicaciones, que realizan tareas en beneficio directo del usuario final. Este es el significado original del término. Es posible que un ordenador cumpla simultáneamente las funciones de cliente y de servidor.

### 3.4.3 Servidor http Apache

Es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTP d 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que Behelendorf eligió ese nombre porque quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de internet. Además, Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, a patchy server (un servidor "parcheado")<sup>9</sup>.

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

---

<sup>9</sup> <http://es.wikipedia.org/wiki/Apache>



### 3.4.3.1 Módulos de apache

La arquitectura del servidor Apache es muy modular. El servidor consta de una sección y diversos módulos que aportan mucha de la funcionalidad que podría considerarse básica para un servidor web. Algunos de estos módulos son:

- mod\_ssl - Comunicaciones Seguras vía TLS.
- mod\_rewrite.- Reescritura de direcciones (generalmente utilizado para transformar páginas dinámicas como php en páginas estáticas html para así engañar a los navegantes o a los motores de búsqueda en cuanto a cómo fueron desarrolladas estas páginas).
- mod\_dav - Soporte del protocolo WebDAV (RFC 2518).
- mod\_deflate - Compresión transparente con el algoritmo deflate del contenido enviado al cliente.
- mod\_auth\_ldap - Permite autenticar usuarios contra un servidor LDAP.
- mod\_proxy\_ajp - Conector para enlazar con el servidor JakartaTomcat de páginas dinámicas en Java (servlets y JSP).

El servidor de base puede ser extendido con la inclusión de módulos externos entre los cuales se encuentran:

- mod\_perl - Páginas dinámicas en Perl.
- mod\_php - Páginas dinámicas en PHP.
- mod\_python - Páginas dinámicas en Python.
- mod\_rexx - Páginas dinámicas en REXX y Object REXX.
- mod\_ruby - Páginas dinámicas en Ruby.
- mod\_aspdotnet - Páginas dinámicas en .NET de Microsoft (Módulo retirado).
- mod\_mono - Páginas dinámicas en Mono.
- mod\_security - Filtrado a nivel de aplicación, para seguridad.



## 3.5 TortoiseSvn.

### 3.5.1 Introducción

El control de versiones es el arte de manejar cambios en la información. Ha sido desde siempre una herramienta crítica para los programadores, quienes típicamente emplean su tiempo haciendo pequeños cambios al software y luego deshaciendo o comprobando esos cambios al día siguiente. Imagine un equipo de estos programadores trabajando concurrentemente.<sup>10</sup>

#### Concepto

TortoiseSVN es un cliente gratuito de código abierto para el sistema de control de versiones *Subversion*. Esto es, TortoiseSVN maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un *repositorio* central. El repositorio es prácticamente lo mismo que un servidor de ficheros ordinario, salvo que recuerda todos los cambios que se hayan hecho a sus ficheros y directorios. Esto permite que pueda recuperar versiones antiguas de sus ficheros y examinar la historia de cuándo y cómo cambiaron sus datos, y quién hizo el cambio. Esta es la razón por la que mucha gente piensa que Subversion, y los sistemas de control de versiones en general, son una especie de “máquinas del tiempo”.

Algunos sistemas de control de versiones también son sistemas de manejo de configuración del software (SCM). Estos sistemas están diseñados específicamente para manejar árboles de código fuente, y tienen muchas características que son específicas para el desarrollo de software - tales como el entendimiento nativo de los lenguajes de programación, o proporcionan herramientas para compilar software. Subversion, sin embargo, no es uno de estos sistemas; es un sistema general que puede ser utilizado para manejar *cualquier* colección de ficheros, incluyendo código fuente

---

10 <https://forja.rediris.es/docman/view.../TortoiseSVN-1.4.1-es.pdf>



### 3.5.2 Características de TortoiseSVN

#### 3.5.2 .1 Integración con el shell de Windows

El cliente de subversión TortoiseSVN se integra perfectamente en la **shell**<sup>11</sup> de Windows (por ejemplo, el explorador). Esto significa que puede seguir trabajando con las herramientas que ya conoce y que no tiene que cambiar a una aplicación diferente cada vez que necesite las funciones del control de versiones.

Por este motivo, puede que en otras aplicaciones la integración no sea tan completa y que, por ejemplo, los iconos sobreimpresionados en las carpetas no se muestren.

**Iconos sobreimpresionados.-** El estado de cada carpeta y fichero versionado se indica por pequeños iconos sobreimpresionados. De esta forma, 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 Subversion están disponibles desde el menú contextual del explorador. TortoiseSVN añade su propio submenú allí.

**Confirmaciones atómicas.-** Una confirmación o bien entra en el repositorio completamente, o no entra en absoluto. Esto permite a los desarrolladores construir y confirmar cambios como unidades lógicas.

**Metadatos versionados.-** Cada fichero y directorio tiene un conjunto invisible de “propiedades” adjuntos. Puede inventarse y almacenar cualquier par de clave/valor que desee. Las propiedades se versionan en el tiempo, igual que el contenido de los ficheros.

**Elección de capas de red.-** Subversion tiene una noción abstracta del acceso al repositorio, haciendo que la gente pueda implementar nuevos mecanismos de red fácilmente. El “avanzado” servidor de red de Subversion es un módulo para el servidor web Apache, que es una variante de HTTP llamada **WebDAV/DeltaV**.

---

<sup>11</sup> [http://es.wikipedia.org/wiki/DOS\\_Shell](http://es.wikipedia.org/wiki/DOS_Shell)



Esto dota a Subversion una gran ventaja en estabilidad e interoperabilidad, y proporciona varias características importantes gratis: autenticación, autorización, compresión de la transmisión y navegación del repositorio.

**Manejo de datos consistente.-** Subversion expresa las diferencias entre ficheros usando un algoritmo de diferenciación binario, que funciona exactamente igual tanto en ficheros de texto (legibles por los humanos) como en ficheros binarios (que no son legibles por nosotros). Ambos tipos de ficheros se almacenan igualmente comprimidos en el repositorio, y las diferencias se transmiten en ambas direcciones por la red.

**Etiquetado y creación de ramas eficiente.-** El costo de crear una rama o una etiqueta no necesita ser proporcional al tamaño del proyecto. Subversion crea ramas y etiquetas simplemente copiando el proyecto, utilizando un mecanismo similar a los vínculos. Por tanto estas operaciones llevan un tiempo pequeño y constante, y muy poco espacio en el repositorio.

**Extensibilidad.-** Subversion no tiene lastre histórico; está implementado como una colección de librerías C compartidas con APIS bien definidas. Esto hace que Subversion sea extremadamente multiplataforma y se pueda utilizar por otras aplicaciones y lenguajes.

## 3.6 RapidSVN

### 3.6.1 Introducción

El cliente de Subversion es una aplicación multiplataforma, es una plataforma visual para el sistema Subversion escrito en C++.

Se utiliza un nuevo marco wxWidgets y también se incluye un cliente de Subversion C++ API en el proyecto.

El objetivo de este proyecto era conseguir un producto de fácil manejo para los usuarios principiantes pero lo suficientemente potente y con herramientas interesantes para los usuarios avanzados.

El programa funciona en cualquier plataforma y se puede ejecutar en Linux, Windows, Mac OS / X, Solaris, etc. Para facilitar más el trabajo los desarrolladores.





### 3.6.2 Características de RapidSVN

- Es multiplataforma, funciona en cualquier sistema operativo.
- Es con licencia libre.

#### 3.6.2.1 Integración con los sistemas Operativos

El cliente de subversión RapidSVN se integra perfectamente en la shell de cualquier sistema operativo. Esto significa que puede seguir trabajando con las herramientas que ya conoce y que no tiene que cambiar a una aplicación diferente cada vez que necesite las funciones del control de versiones, solo se debe descargar el cliente para el sistema operativo que se encuentre trabajando.

#### 3.6.2.2 Sistema Operativo

Subversion.- fue originalmente desarrollado para trabajar bajo el entorno de Linux, en la actualidad puede ser usado en otros sistemas operativos.

Linux es la denominación de un sistema operativo y el nombre de un núcleo. Es uno de los paradigmas del desarrollo de software libre, donde el código fuente está disponible públicamente y cualquier persona puede libremente usarlo, modificarlo y redistribuirlo.

Varias distribuciones como **RedHat, Mandrake, Fedora, Debian, Slackware, Ubuntu** fueron usadas con Subversion. Para el desarrollo del proyecto utilizaré la distribución de **Linux Ubuntu 9.4.**

## 4. EVALUACIÓN DEL OBJETO DE INVESTIGACIÓN

La Carrera de Ingeniería en Sistemas del Área de Energía y Recursos Naturales no Renovables de la Universidad Nacional de Loja, forma profesionales con un elevado conocimiento en la planificación, análisis, diseño y elaboración de soluciones informáticas a medida y así ser capaces de dar respuesta a las necesidades de nuestro entorno dentro de la sociedad. Es así que, como egresado de ésta carrera, con el objetivo de llevar a la práctica todos los conocimientos adquiridos durante la carrera universitaria, propuse implementar un servidor de repositorio de versiones SVN



en Linux para ayudar a optimizar tiempo y recursos al momento de fusionar código fuente de los programas a desarrollar en la carrera de Ingeniería en sistema de la Universidad Nacional de Loja.

Para dar cumplimiento a los objetivos planteados en el proyecto se realizaron una serie de actividades, tal es el caso que para recopilar la información de los datos y procesos que se utilizan para el desarrollo de aplicaciones en la carrera de Ingeniería en Sistemas y la forma de unir el código fuente independientemente al lenguaje de programación, la información se obtuvo en el departamento de la UDI (Unidad de desarrollo informático) de la carrera de Ingeniería en sistema. Así mismo la verificación de una buena conexión de la red a los diferentes lugares del *Área de Energía Industria y Recursos Naturales no Renovables de la Universidad Nacional de Loja*.

En lo que se refiere a la instalación y configuración del servidor de repositorio svn en Linux, se lo realizó igualmente con la ayuda del departamento de la Unidad de desarrollo Informático, donde el procedimiento usado para el efecto es la instalación del sistema operativo Linux y los requerimientos del Hardware para el buen funcionamiento de la aplicación. Para esto se investigó de los diferentes distribuciones de Linux cual se adaptaría de mejor manera para la implementar la Subversion y a su vez sea fácil uso y manejo para los usuarios finales lo cual permitirá familiarizar a los nuevos estudiantes del Área a utilizar Software libre.

Otro punto importante es crear los repositorios de las tesis en desarrollo de la carrera de Ingeniería en Sistema con sus respectivos niveles de acceso a los usuarios , que permita agilizar las tareas y el acceso a la información (datos) a los diferentes programadores (Clientes), en este caso se optó por la instalación y configuración de Subversion y apache2.0 el cual nos permite registrar los usuarios creación de los repositorios, datos de la consulta del de los cambios efectuados por cada usuario final .

El sistema implementado permite que la información del o los programadores sea de fácil y rápido acceso para lo o los repositorios creados en el servidor de la carrera, que diariamente, semanalmente o mensualmente en bien sus cambios al servidor y así optimizar recursos y tiempo al momento de funcionar.



## **Características y Requerimientos de Hardware y Software**

El presente proyecto ayuda a la Automatización de los procesos de desarrollo de proyectos de programación, a fin de que exista un eficaz manejo de información, con lo cual agilice y mejore los procesos de fusión de código fuente y por ende una correcta toma de decisiones, su informatización logra que dichos procesos se elaboren de una manera eficaz logrando con ello el ahorro de tiempo y dinero, permitiéndose una mejor organización.

El uso del repositorio de versiones Subversion mencionada procura la rapidez en cuanto a la presentación de resultados, debido a que cubre con todas las expectativas del desarrollo del software previstas en un inicio, por el conocimiento adquirido con lo que respecta al manejo de todos sus componentes.

Todos los objetivos planteados se han cumplido y se ven reflejados en el servidor, actualmente el servidor tiene la IP 172.16.44.119. El cual se encuentra alojado en el departamento de desarrollo Informático de la Carrera de Ingeniería en Sistemas del Área de Energía y Recursos Naturales no Renovables de la Universidad Nacional de Loja, consiguiéndose un sistema totalmente eficiente.

### **Hardware Mínimo**

Subversion es intensivo en el uso del procesador, ya que usa el propio procesador del computador para hacer el procesamiento de los datos.

Para la configuración de un servidor Subversiones con las características descritas a continuación. Se debe tener un PC con procesador compatible con placa Intel, no inferior a un Pentium IV 300Mhz con 512 MB en memoria RAM es lo suficiente y disco duro de 80 GB (dependiendo del uso a realizar a los repositorios). Así mismo no precisa de una placa de vídeo sofisticada o periféricos; puertos seriales, paralelos y USB pueden ser completamente deshabilitados. Pero una buena tarjeta de red es esencial.

### **Software Mínimo**

Sistema operativo Linux distribución Ubuntu 9.04 o superior, servidor apache 2 o superior, Subversion versión. TortoiseSVN 1.2.0, RapidSVN, sistema operativo para cliente independiente (Soporte svn).



## 5 DESARROLLO DE LA PROPUESTA ALTERNATIVA.

### 5.1 Análisis de la Aplicación

#### 5.1.1 Definición del problema

El Área de Energía Industrias Recursos Naturales no Renovables de la Universidad Nacional de Loja, cuenta con diferentes carreras, que mediante actividades específicas contribuyen a la prestación de servicios educativos, para el aseguramiento de la Calidad profesional del estudiante para el buen desenvolvimiento en la sociedad en que nos rodea.

En esta institución no existe un sistema informático que ayude agilizar las labores del departamento de desarrollo de proyectos para tener una buena **calidad**, como son la de fusionar código fuente, tener un historial de los cambios realizados por cada programador, además de la molestia del tiempo que contrae en realizar manualmente ya que el proceso no termina ahí, sino que aún debe realizar otros cambios para sacar su producto final.

Otro problema que existe es el de los estudiantes de la carrera de Ingeniería en Sistemas, es que las investigaciones que realizan cada módulo necesitan desarrollar un proyecto el cual necesitan dividirse las tareas para la resolución del mismo ya que se maneja de forma aislada.

Es así que para solucionar los problemas mencionados anteriormente esta investigación pretende realizar la “**CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMATICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA**”, que permita el ahorro de tiempo, problemas o errores al momento de funcionar el código fuente de los programadores y ser utilizarlos y sea utilizado los cambios en línea de una manera que se pueda lograr un trabajo más rápido, y eficiente.

Para dar cumplimiento a los objetivos planteados del proyecto propongo la siguiente solución que permitirá el uso del servicio de repositorios svn en Linux en la



red de datos de la Unidad de Desarrollo Informático del A.E.I.R.N.N.R de la Universidad Nacional de Loja a través de Software Subversion y los clientes svn.

Aquí se describe la configuración de software y hardware necesario para la instalación de Subversion y la conexión con el servidor de la Unidad de Desarrollo Informático del A.E.I.R.N.N.R. De la Universidad Nacional de Loja actualmente en la carrera de Ingeniería en Sistemas.

En cuanto a hardware se detallará el lugar donde se ubica el servidor de repositorios de versiones y las seguridades requeridas para un óptimo funcionamiento, así mismo describimos los pasos necesarios para la configuración del repositorio general (padre) y los repositorios hijos.

En la parte de software, se describe los paquetes para la instalación de Subversion y apache en el servidor de repositorios, las configuraciones necesarias en los archivos de configuración, la utilización de la interfaz Figuras para los clientes, y acceso mediante http.

#### **5.1.2 Diagrama del servidor de repositorio svn en la Unidad de Desarrollo Informático de la Universidad Nacional de Loja**

El servidor queda ubicado en el departamento de la Unidad de Desarrollo Informático de la carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja, debido a que es el punto central de la red interna del área y el punto de llegada de la red externa de la Universidad Nacional de Loja. Además los servidores que posee el A.E.I.R.N.N.R. se encuentran ubicados en este departamento.

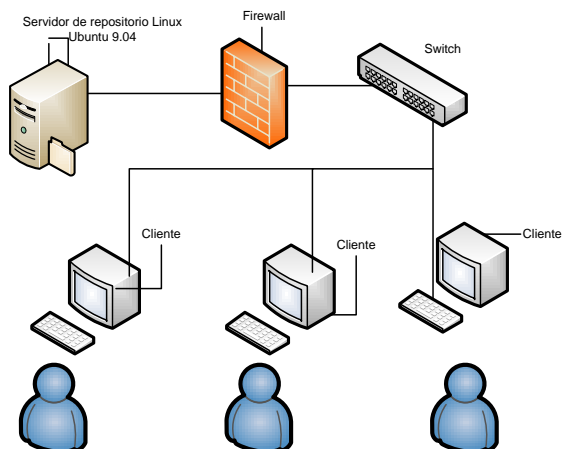
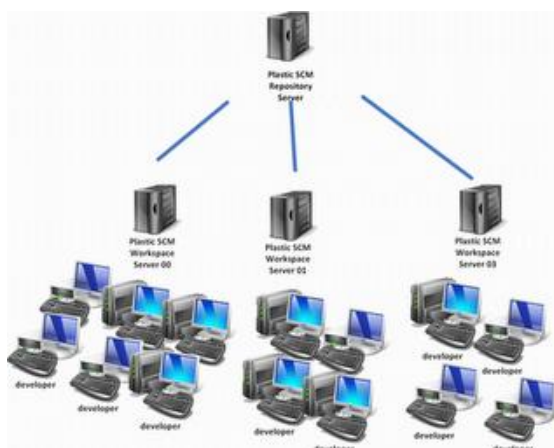


Grafico 4.2.1 DIAGRAMA DEL SERVIDOR DE REPOSITORIO SVN EN LA UNIDAD DE DESARROLLO INFORMÁTICO DE LA UNIVERSIDAD NACIONAL DE LOJA

**Figura 5.1: Diagrama de ciclo de vida de Subversion**



**Figura 5.2: Requerimientos del Servidor de Repositorios**

### 5.1.2.1 Requerimientos del Servidor de Repositorios

Para la configuración de un servidor Subversiones con las características descritas a continuación.



Hardware	Descripción
Procesador	Pentium IV 2.4 Mhz
Memoria RAM	512 Mb
Disco Duro	80 Gb
Tarjeta de Red	ADVANTEK/CNET 10/100
Motherboard	Intel
Tabla 5.1: Hardware para el servidor del repositorios	

### 5.1.3 Configuración del Servidor

A continuación detallare paso a paso las configuraciones para el servidor de repositorio svn en Linux Ubuntu 9.04 y clientes en Windows.

### 5.1.4 Configuración de Red

En el servidor por confiabilidad se utiliza una **dirección IP estática**. Para configurar la dirección de red, se ingresa al directorio:

- **Configuración de la dirección DHCP**

```
Sudo vi /etc/network/interfaces
```

```
#the primary network interface
```

```
auto eth0
```

```
iface eth0 inetdhcp
```



Luego guardamos y reiniciamos el servidor de red

```
sudo /etc/init.d/networking restart
```

- **Configuración manual**

```
sudo vi /etc/network/interface
```

```
auto eth0
```

```
iface eth0 inet static
```

```
address 172.16.44.119
```

```
netmask 255.255.240
```

```
network 172.16.44.0
```

```
broadcast 172.16.44.255
```

```
gateway 172.16.44.1
```

Luego guardamos y reiniciamos el servidor de red `sudo /etc/init.d/networking restart`

- **Modo Grafico**

Iniciamos el sistema, administración, elegimos la opción **red**, el sistema pide la **clave de root** para habilitar el permiso, abre una ventana en donde editamos la tarjeta de red y agregamos nuestra dirección IP, mascara de subred, los Dns.

### 5.1.5 Configuración de Subversion

En este capítulo mostraremos cómo configurar un servidor Subversion en un sistema GNU/Linux (más concretamente para Ubuntu aunque la mayor parte es válida para cualquier distribución).





## 5.1.6 Instalación de Subversion

### 5.1.7 Instalación mediante líneas de Código

- 1) Descargar las e instalar los paquetes de svn subversion-tools
  - Ingresar a la consola de servidor y digitamos el comando  
sudo apt-get install subversion libapache2-svn para instalar subversion.
- 2) Instalación de Apache y sus dependencias
  - Ingresar a la consola de servidor y digitamos el comando  
sudo apt-get install subversion libapache2-svn

#### 5.1.7.1 Configuración:

- *Crear carpetas para el repositorio padre mediante el comando.*
  - sudo mkdir /opt/svn/energía/tesis
- Le otorgamos permiso a la carpeta padre, en este caso el repositorio svn principal.
  - sudo chmod 777 -R / /opt/svn/energía/tesis
- **Configuramos el archivo de configuración de apache2**
  - sudo vim /etc/apache2/apache2.conf
  - Al final del archivo de configuración apache2.conf le agregamos las siguientes líneas.

< Location /svn/tesis>

DAV	Svn
SVNParentPath	/opt/svn/energia/tesis
AuthzSVNAccessFile	/etc/apache2/.svn_access_control
Require	valid-user
AuthType Basic	
AuthName	"REPOSITORIO SUBVERSION UNL AEIRNNR ACCESO RESTRINGIDO"



AuthUserFile                      /etc/apache2/dav\_svn.passwd

</Location>

Esta definición solo permite acceso a usuarios validos (Require valid-user), si se quieren permitir usuarios anónimos o colocar otro tipo de restricciones a los accesos deben consultar la documentación de Apache. Fíjense en los 3 valores a modificar

**SVNParentPath:** debe apuntar al directorio padre de los repositorios.

**AuthzSVNAccessFile:** define los accesos de los usuarios a los repositorios.

**AuthUserFile:** archivo con la definición de usuarios y los passwords.

#### o Reiniciamos el servidor de apcahe2

```
sudo /etc/init.d/apache2 restart
```

Si creamos otros repositorios más adelante no será necesario reiniciar Apache, sólo ésta primera vez ya que hemos cambiado uno de sus archivos de configuración

#### o Habilitamos el css (opcional)

- Opcionalmente podemos dar un mejor aspecto al front del repositorio, esto es mediante la habilitación del css, que tiene subversión por defecto, debemos mover los achivos de svnindex.css y svnindex.xsl a la carpeta /var/www
- ```
sudo cp /var/www/apache2-default/svnindex.css /var/www
```
- ```
sudo cp /var/www/apache2-default/svnindex.xsl /var/www
```

Modificamos el archivo de configuración apache2.conf. Digitando en la consola del servidor

```
sudo vim /etc/apache2/apache2.conf
```

Modificamos el archivo de configuración apache2.conf del repositorio padre agregando  
SVNIndexXSLT /svnindex.xsl

< Location /svn/tesis>



```
DAV                                Svn

SVNParentPath                      /opt/svn/energia/tesis

AuthzSVNAccessFile                 /etc/apache2/.svn_access_control

Require                            valid-user

SVNIndexXSLT                       /svnindex.xsl

AuthType Basic

AuthName                           "REPOSITORIO    SUBVERSION
                                UNL    AEIRNNR    ACCESO
                                RESTRINGIDO"

AuthUserFile                       /etc/apache2/dav_svn.passwd

</Location>
```

○ **Crear nuestro repositorio svn**

- `sudo svnadmin create /opt/svn/tesis/mi_repositorio`
- `sudo chmod 777 -R`  
`/opt/svn/energía/tesis/mi_repositorio`
- `sudo chown -R www-data`  
`/opt/svn/energía/tesis/mi_repositorio`

Con Apache 2 instalado tiene más sentido que el propietario de tus repositorios sea el usuario **www-data**

○ **Asignar propiedad y permisos al repositorio y sus subarchivos**

- `sudo htpasswd2 -c`  
`/etc/apache2/dav_svn.passwd us1`
- `sudo htpasswd2`  
`/etc/apache2/dav_svn.passwd us2`

Para el primer usuario que autoricemos debemos especificar el modificador **-c** para que el archivo almacén de contraseñas se cree por



primera vez. Después podremos añadir tantos usuarios como queramos sin especificar **-c**, tal y como muestra la segunda línea.

new password:

re-type new password:

updating password for user user1

Si se quiere limpiar el archivo y definir usuarios desde cero se debe ejecutar el comando (se recomienda hacer esto la primera vez)

- **Asignamos permiso por grupo a los repositorios**

La definición de los accesos se hace en el archivo definido por la entrada **AuthzSVNAccessFile**, en nuestro caso es **/etc/apache2/svn\_access\_control**. En este archivo se definen grupos de accesos y permisos de acceso a cada repositorio.

- **Crear el archivo de accesos por grupo.**

```
sudo vim/etc/apache2/svn_access_control
```

```
# Se definen los grupos de trabajo
```

```
[groups]
```

```
svn=user1, user2
```

```
guest=user3, user4
```

```
deveryone=@devteam, @guest
```

```
# Acceso de lectura para todo el mundo
```

```
[/]
```

```
* = r
```

```
# guest pueden leer pero solo devteam actualizar en
```

```
# el repositorio repo
```

```
[repo:/]
```

```
@devteam=rw
```

```
@guest=r
```

```
# Todo el mundo puede leer el repo test pero solo
```

```
# user5 lo puede modificar
```

```
[test:/]
```

```
user5=rw
```



```
@everyone=r
# Solo el grupo devteam tiene acceso total al
# repositorio bcol. Guest no tienen ningun acceso
[bcol:/]
@devteam = rw
@guest =
```

- Para que Apache tome los cambios se deben habilitar los módulos de svn y reiniciar el servicio

```
sudo a2enmod dav
```

```
sudo a2enmod dav_svn
```

```
sudo /etc/init.d/apache2 force-reload
```

### 5.1.8 Instalación gráfica del repositorio Subversion

Para la instalación de Subversion en Ubuntu se realiza de la siguiente manera abrimos la **Shell**. Y escribimos `sudo apt-getinstall subversion subversion-tools`, o podemos ir Sistema, **gestor de paquetes synaptic de Ubuntu**, buscar **subversion-tools** e instalarlo, esto tardara dependiendo el ancho de banda del proveedor de internet.

#### Componentes

- **svn**: programa cliente (línea de comando).
- **svnversion**: programa que reporta el estado de la copia de subversion.
- **svnlook**: herramienta para inspeccionar el repositorio de subversion.
- **svnadmin**: herramienta para crear, reparar, etc. el repositorio de subversion.
- **mod\_dav\_svn**: modulo para apache, que permite acceso remoto.

### 5.1.9 Instalación del servidor del apache

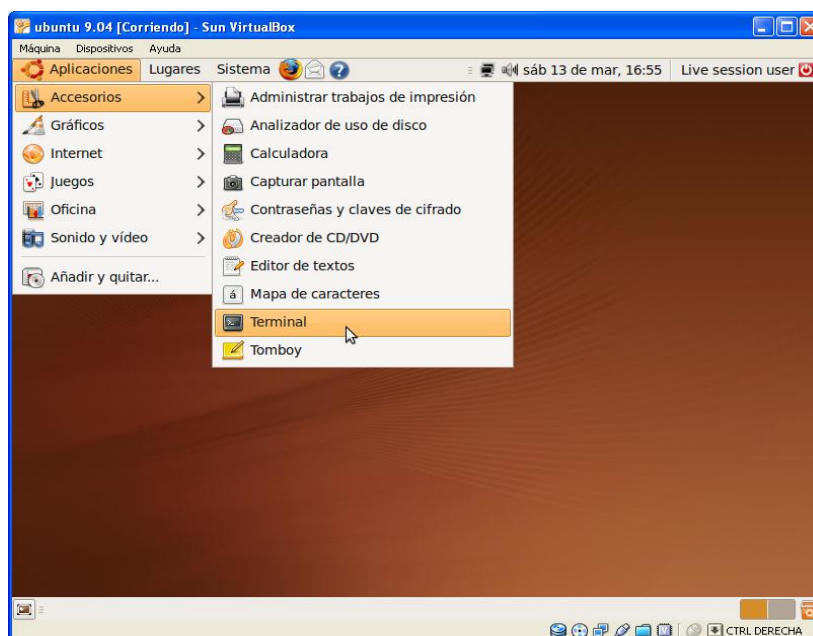
Para la instalación del servidor del **apache** en Ubuntu se realiza de la siguiente manera abrimos la **Shell**, escribimos `sudo apt-getinstall subversion libapache2-svn`, o podemos ir Sistema, **gestor de paquetes synaptic de Ubuntu**, buscar

**libapache2-svn** e instalarlo, esto tardara dependiendo el ancho de banda del proveedor de internet.

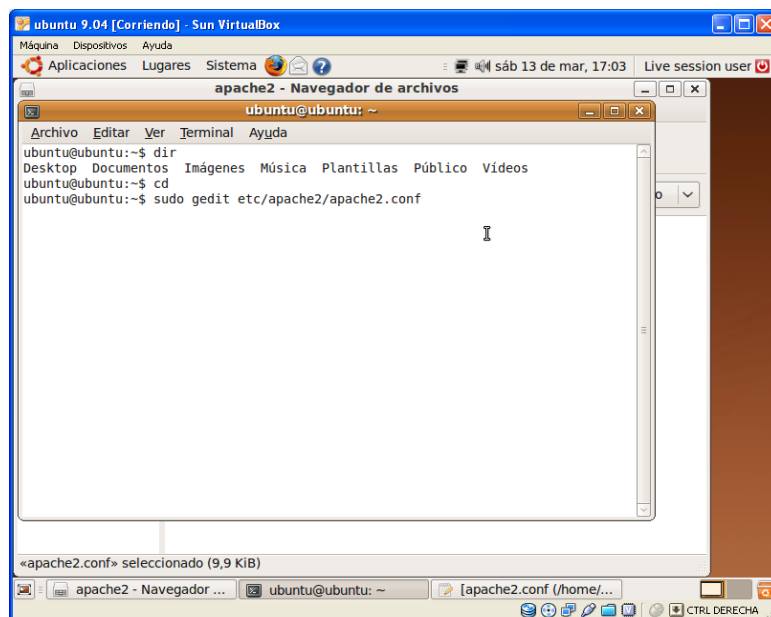
#### 5.1.10 Configuración del archivo **apache2.conf**

Luego de instalar la Subversion y el servidor de apache proseguimos a configurar el archivo de **apache2.conf**.

Abrimos la Shell, podemos utilizar **vim**, **nano**, **gedit** o el editor que más guste, en la Shell digitamos `sudo vim /etc/apache2/apache2.conf`, es la ruta donde se encuentra el archivo de configuración de nuestro servidor de apache2.



**Figura 5.3: Configuración del servidor apache2**



**Figura 5.4: Configuración del servidor comando gedit**

Archivo de configuración del Apache2 usando el comando **gedit**

### **Inicio del archivo de configuración apache2.conf**

```
#
# Based upon the NCSA server configuration files originally by Rob McCool.
#
# This is the main Apache server configuration file. It contains the
# configuration directives that give the server its instructions.
# See http://httpd.apache.org/docs/2.2/ for detailed information about
# the directives.
#
# Do NOT simply read the instructions in here without understanding
# what they do. They're here only as hints or reminders. If you are unsure
# consult the online docs. You have been warned.
#
# The configuration directives are grouped into three basic sections:
# 1. Directives that control the operation of the Apache server process as a
#    whole (the 'global environment').
# 2. Directives that define the parameters of the 'main' or 'default' server,
#    which responds to requests that aren't handled by a virtual host.
#    These directives also provide default values for the settings
#    of all virtual hosts.
# 3. Settings for virtual hosts, which allow Web requests to be sent to
#    different IP addresses or hostnames and have them handled by the
#    same Apache server process.
#
# Configuration and logfile names: If the filenames you specify for many
# of the server's control files begin with "/" (or "drive:/" for Win32), the
# server will use that explicit path. If the filenames do *not* begin
# with "/", the value of ServerRoot is prepended -- so "/var/log/apache2/foo.log"
# with ServerRoot set to "" will be interpreted by the
```



```
# server as "/var/log/apache2/foo.log".
#

### Section 1: Global Environment
#
# The directives in this section affect the overall operation of Apache,
# such as the number of concurrent requests it can handle or where it
# can find its configuration files.
#

#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# NOTE! If you intend to place this on an NFS (or otherwise network)
# mounted filesystem then please read the LockFile documentation (available
# at <URL:http://httpd.apache.org/docs-2.1/mod/mpm_common.html#lockfile>);
# you will save yourself a lot of trouble.
#
# Do NOT add a slash at the end of the directory path.
#
ServerRoot "/etc/apache2"

#
# The accept serialization lock file MUST BE STORED ON A LOCAL DISK.
#
#<IfModule !mpm_winnt.c>
#<IfModule !mpm_netware.c>
LockFile /var/lock/apache2/accept.lock
#</IfModule>
#</IfModule>

#
# PidFile: The file in which the server should record its process
# identification number when it starts.
# This needs to be set in /etc/apache2/envvars
#
PidFile ${APACHE_PID_FILE}

#
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 300

#
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On

#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
```





```
#
MaxKeepAliveRequests 100

#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 15

##
## Server-Pool Size Regulation (MPM specific)
##

# prefork MPM
# StartServers: number of server processes to start
# MinSpareServers: minimum number of server processes which are kept spare
# MaxSpareServers: maximum number of server processes which are kept spare
# MaxClients: maximum number of server processes allowed to start
# MaxRequestsPerChild: maximum number of requests a server process serves
<IfModule mpm_prefork_module>
    StartServers      5
    MinSpareServers   5
    MaxSpareServers   10
    MaxClients        150
    MaxRequestsPerChild 0
</IfModule>

# worker MPM
# StartServers: initial number of server processes to start
# MaxClients: maximum number of simultaneous client connections
# MinSpareThreads: minimum number of worker threads which are kept spare
# MaxSpareThreads: maximum number of worker threads which are kept spare
# ThreadsPerChild: constant number of worker threads in each server process
# MaxRequestsPerChild: maximum number of requests a server process serves
<IfModule mpm_worker_module>
    StartServers      2
    MaxClients        150
    MinSpareThreads   25
    MaxSpareThreads   75
    ThreadsPerChild   25
    MaxRequestsPerChild 0
</IfModule>

# These need to be set in /etc/apache2/envvars
User ${APACHE_RUN_USER}
Group ${APACHE_RUN_GROUP}

#
# AccessFileName: The name of the file to look for in each directory
# for additional configuration directives. See also the AllowOverride
# directive.
#

AccessFileName .htaccess
```



```
#
# The following lines prevent .htaccess and .htpasswd files from being
# viewed by Web clients.
#
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>

#
# DefaultType is the default MIME type the server will use for a document
# if it cannot otherwise determine one, such as from filename extensions.
# If your server contains mostly text or HTML documents, "text/plain" is
# a good value.  If most of your content is binary, such as applications
# or images, you may want to use "application/octet-stream" instead to
# keep browsers from trying to display binary files as though they are
# text.
#
DefaultType text/plain

#
# HostnameLookups: Log the names of clients or just their IP addresses
# e.g., www.apache.org (on) or 204.62.129.132 (off).
# The default is off because it'd be overall better for the net if people
# had to knowingly turn this feature on, since enabling it means that
# each client request will result in AT LEAST one lookup request to the
# nameserver.
#
HostnameLookups Off

# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
# container, error messages relating to that virtual host will be
# logged here.  If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog /var/log/apache2/error.log

#
# LogLevel: Control the number of messages logged to the error_log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel warn

# Include module configuration:
Include /etc/apache2/mods-enabled/*.load
Include /etc/apache2/mods-enabled/*.conf

# Include all the user configurations:
Include /etc/apache2/httpd.conf
```



```
# Include ports listing
Include /etc/apache2/ports.conf

#
# The following directives define some format nicknames for use with
# a CustomLog directive (see below).
# If you are behind a reverse proxy, you might want to change %h into %{X-Forwarded-For}i
#
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
vhost_combined
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

#
# Define an access log for VirtualHosts that don't define their own logfile
CustomLog /var/log/apache2/other_vhosts_access.log vhost_combined

#
# Customizable error responses come in three flavors:
# 1) plain text 2) local redirects 3) external redirects
#
# Some examples:
#ErrorDocument 500 "The server made a boo boo."
#ErrorDocument 404 /missing.html
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://www.example.com/subscription_info.html
#

#
# Putting this all together, we can internationalize error responses.
#
# We use Alias to redirect any /error/HTTP_<error>.html.var response to
# our collection of by-error message multi-language collections. We use
# includes to substitute the appropriate text.
#
# You can modify the messages' appearance without changing any of the
# default HTTP_<error>.html.var files by adding the line:
#
# Alias /error/include/ "/your/include/path/"
#
# which allows you to create your own set of files by starting with the
# /usr/share/apache2/error/include/ files and copying them to /your/include/path/,
# even on a per-VirtualHost basis. The default include files will display
# your Apache version number and your ServerAdmin email address regardless
# of the setting of ServerSignature.
#
# The internationalized error documents require mod_alias, mod_include
# and mod_negotiation. To activate them, uncomment the following 30 lines.

# Alias /error/ "/usr/share/apache2/error/"
#
# <Directory "/usr/share/apache2/error">
```



```
# AllowOverride None
# Options IncludesNoExec
# AddOutputFilter Includes html
# AddHandler type-map var
# Order allow,deny
# Allow from all
# LanguagePriority en cs de es fr it nl sv pt-br ro
# ForceLanguagePriority Prefer Fallback
# </Directory>
#
# ErrorDocument 400 /error/HTTP_BAD_REQUEST.html.var
# ErrorDocument 401 /error/HTTP_UNAUTHORIZED.html.var
# ErrorDocument 403 /error/HTTP_FORBIDDEN.html.var
# ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
# ErrorDocument 405 /error/HTTP_METHOD_NOT_ALLOWED.html.var
# ErrorDocument 408 /error/HTTP_REQUEST_TIME_OUT.html.var
# ErrorDocument 410 /error/HTTP_GONE.html.var
# ErrorDocument 411 /error/HTTP_LENGTH_REQUIRED.html.var
# ErrorDocument 412 /error/HTTP_PRECONDITION_FAILED.html.var
# ErrorDocument 413 /error/HTTP_REQUEST_ENTITY_TOO_LARGE.html.var
# ErrorDocument 414 /error/HTTP_REQUEST_URI_TOO_LARGE.html.var
# ErrorDocument 415 /error/HTTP_UNSUPPORTED_MEDIA_TYPE.html.var
# ErrorDocument 500 /error/HTTP_INTERNAL_SERVER_ERROR.html.var
# ErrorDocument 501 /error/HTTP_NOT_IMPLEMENTED.html.var
# ErrorDocument 502 /error/HTTP_BAD_GATEWAY.html.var
# ErrorDocument 503 /error/HTTP_SERVICE_UNAVAILABLE.html.var
# ErrorDocument 506 /error/HTTP_VARIANT_ALSO_VARIES.html.var
```

```
# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.
```

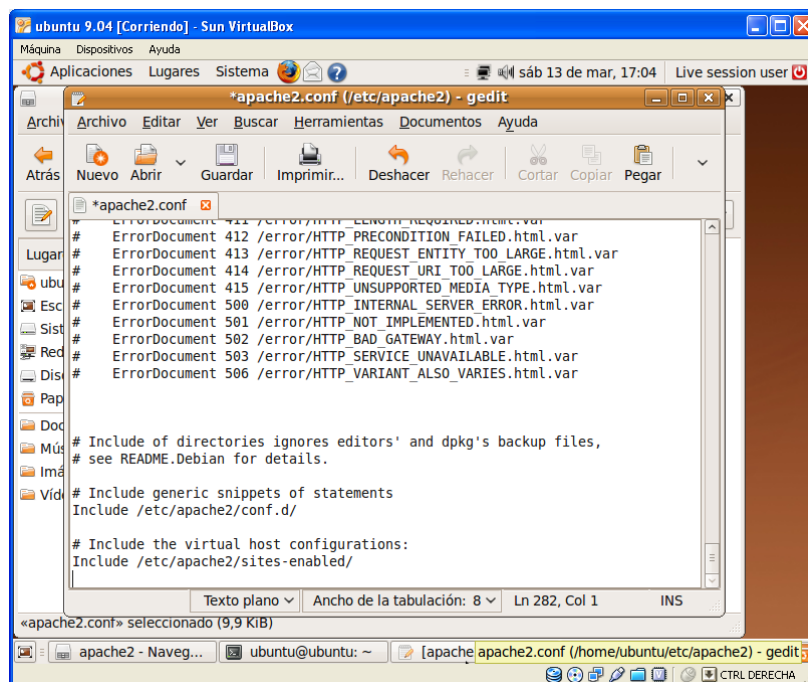
```
# Include generic snippets of statements
Include /etc/apache2/conf.d/
```

```
# Include the virtual host configurations:
Include /etc/apache2/sites-enabled/
```

```
# Configuración del repositorio
```

```
<Location /svn>
DAV svn
SVNParentPath /opt/svn/unlrepo
</Location>
```

**Fin del archivo de configuración del apache2.conf**



**Figura 5.5: Archivo del servidor apache.conf**

Subversion depende también de la tercera parte en las siguientes colecciones: Libapr y libapr-util (Requerido para cliente y servidor)

El Apache Portable Runtime (APR) ofrece una biblioteca de abstracción de la explotación de servicios de nivel de sistema como archivo de I/O, la gestión de memoria, y así sucesivamente.

SQLite (Requerido para cliente y servidor).- Subversion utiliza SQLite para gestionar algunas de las bases de datos internas.

Libz (Requerido para cliente y servidor).- Subversion utiliza zlib para comprimir las diferencias binarias. Estas se utilizan en todas partes: en la red, en el repositorio, y en la copia de trabajo del cliente.

Libserf (opcional para el cliente) Las bibliotecas de Serf, permite que el cliente de Subversion pueda enviar peticiones HTTP. Esto es necesario si desea que su cliente pueda acceder a un repositorio servido por el servidor de Apache HTTP.

OpenSSL (opcional para el cliente y servidor) OpenSSL permite a su cliente el acceso encriptado. Para usar SSL con el servidor de Subversion WebDAV, Apache, necesita elaborarse con OpenSSL.



Existen dos maneras diferentes de implementar el repositorio. Una aplicación almacena los datos en un bloque de sistema de archivos, y las aplicaciones de otros datos en una base de datos Berkeley DB (conocido como BDB).

Cuando se crea un repositorio, (Windows) usted tiene la opción de especificar el lugar de almacenamiento de back-end. Berkeley<sup>12</sup> DB back-end, sólo será disponible si las bibliotecas de BDB se encuentran inicializadas.

#### 5.1.11 Configuración del archivo **apache2.conf** sin restricciones de acceso a usuarios

Para la realización de la configuración de repositorios sin claves de acceso por los usuarios o grupos de usuario editamos el archivo de configuración **apache2** que se encuentra ubicado en la dirección: **etc/apache2/**.

Se puede elegir cualquier editor de Linux en mi caso elegí el vim.

Abrimos la Shell y digitamos `sudo vim etc/apache2/apche2.conf`.

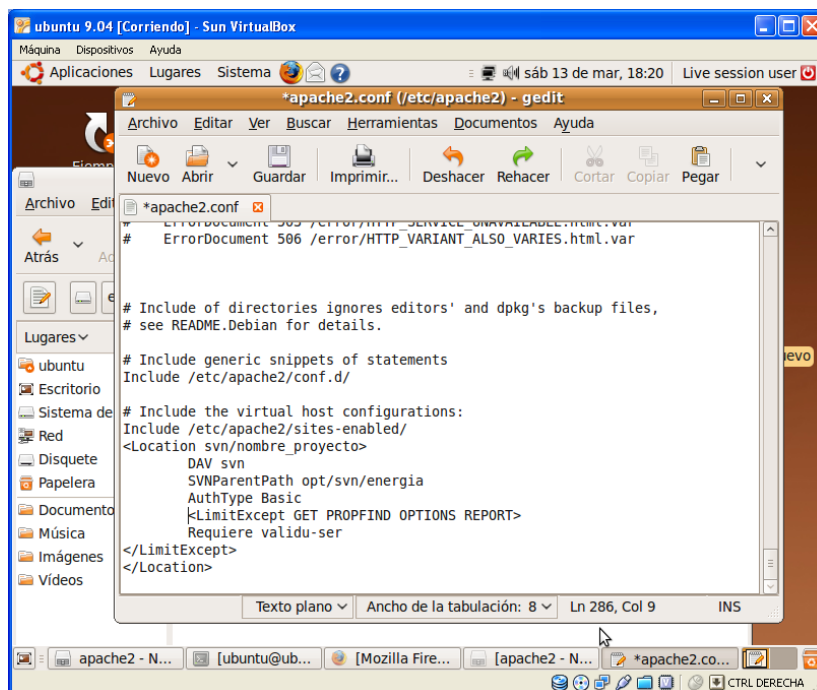
Para realizar cambios en este archivo de configuración debemos ingresar como **root**.

Digitando en la Shell: **su root** y nos pide la clave del mismo.

Luego nos abrirá el archivo de configuración y al final del archivo escribimos:

---

<sup>12</sup> <http://www.ventanazul.com/articulos/porque-usar-fsfs-y-no-berkeley-con-subversion>



**Figura 5.6: Configuración del repositorio sin restricciones**

#### # Configuración del repositorio

<Location /svn>

```
DAV svn
#DIRECCION DEL LUGAR DEL REPOSITORIO
SVNParentPath /opt/svn/DIRECTORIO
```

</Location>

Luego guardamos y reiniciamos el servidor de apache para que los cambios se realicen. Mediante la siguiente línea de código:

```
sudo /etc/init.d/apache2 restart
```

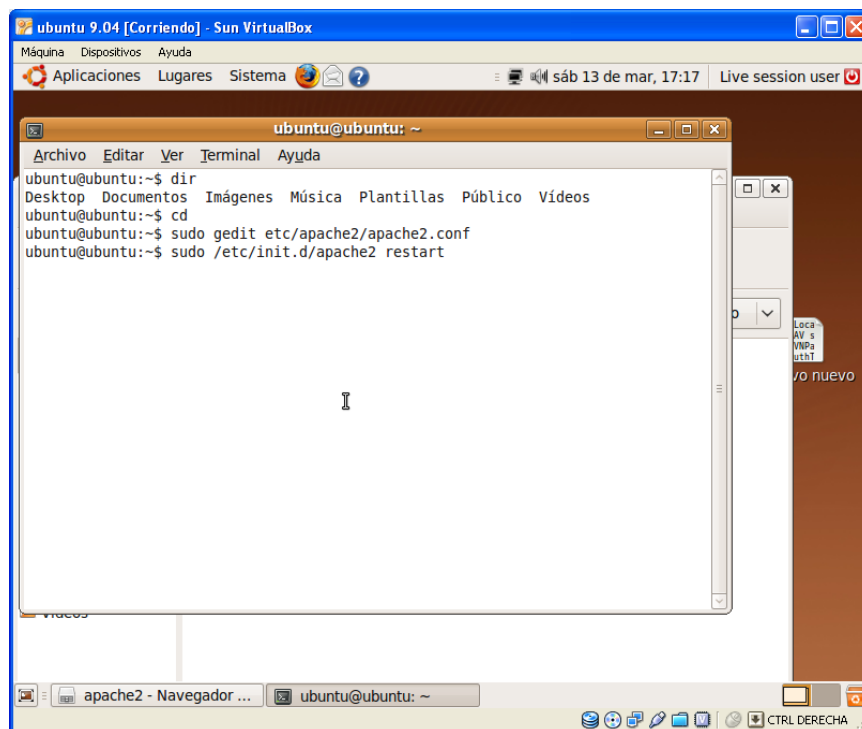


Figura 5.7: Reiniciar el servicio de apache

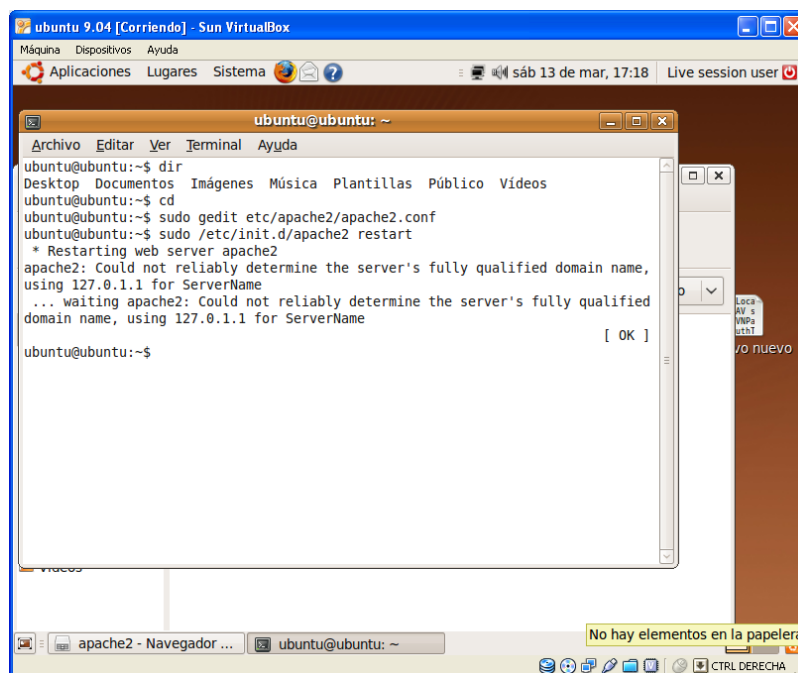


Figura 5.8: Terminación del proceso reiniciar el servicio de apache

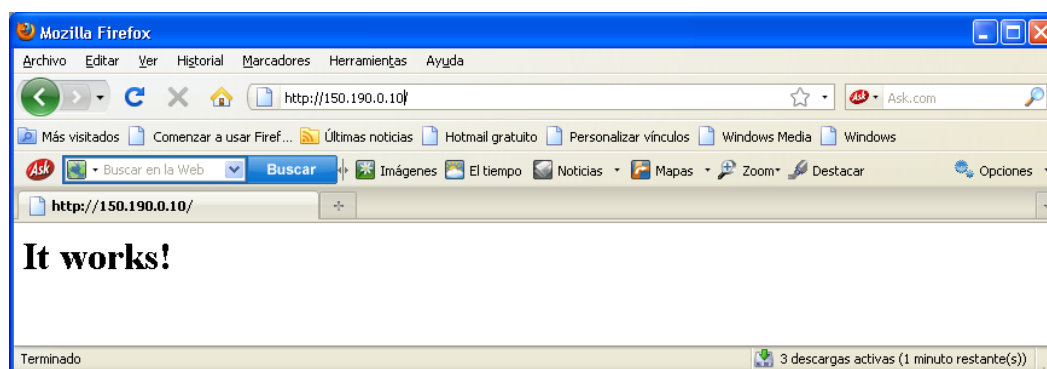


**<Location /svn/proyecto>** indica la ruta de acceso a nuestro repositorio a los clientes o el acceso de protocolo http.

**SVNParentPath:** es el directorio padre de los repositorios.

**</Location>** fin de la línea de configuración.

Para comprobar si está bien la configuración abrimos el navegador que tengamos y escribimos la **http://ip de la máquina**. Y nos debe salir la siguiente vista.



**Figura 5.9: Página de inicio del servidor apache**

#### **Nota:**

Si muestra una página como la Figura 5.9 debe revisar la configuración nuevamente.

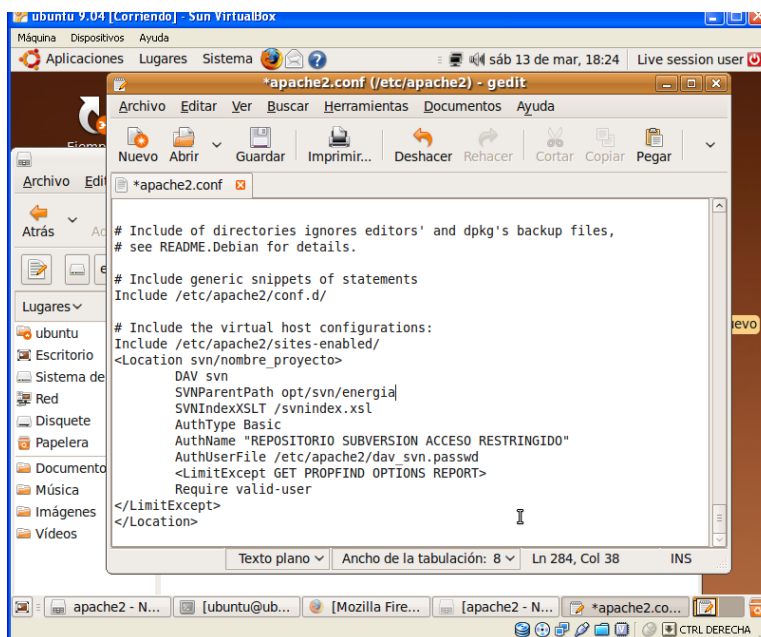
#### **5.1.12 Configuración del archivo apache2.conf con restricciones de acceso a usuarios.**

La configuración de repositorios con restricciones de acceso por los usuarios o grupos de usuario editamos el archivo de **configuración apache2** que se encuentra ubicado en la dirección de **etc/apache2/**.

Para realizar cambios en este archivo de configuración ingresamos como root digitando en la Shell su root y nos pide la clave del mismo.

Podemos elegir cualquier editor de Linux en mi caso elegí el vim.

Abrimos la Shell y digitamos `sudo vim etc/apache2/apche2.conf`.



**Figura 5.10. Configuración del repositorio con restricciones**

```
<Location /svn/proyecto>
```

```
DAV svn
```

```
SVNParentPath /opt/svn/energia/tesis
```

```
AuthType Basic
```

```
AuthName "REPOSITORIO SUBVERSION UNL AEIRNNR ACCESO RESTRINGIDO"
```

```
AuthUserFile /etc/apache2/dav_svn.passwd
```

```
<LimitExcept GET PROPFIND OPTIONS REPORT >
```

```
Require valid-user
```

```
< /LimitExcept>
```

```
</Location>
```



Para comprobar si está bien la configuración abrimos el navegador que tengamos y

Escribimos la **http://ip de la máquina**. Y nos debe salir la siguiente vista.

Esta definición solo permite el acceso a usuarios validos (Require valid-user), si se

Si se quieren permitir usuarios anónimos o colocar otro tipo de restricciones a los accesos, deben consultar la documentación de Apache. Fíjense en los 3 valores a modificar.

**AuthName:** Mensaje que muestra al usuario final el momento de acceder al repositorio.

**SVNParentPath:** debe apuntar al directorio padre de los repositorios.

**AuthUserFile:** archivo con la definición de usuarios y los **passwords**.

#### 5.1.13 Creación de los usuarios para los repositorios en el archivo de configuración dav\_svn.passwd

Para la configuración de repositorios con restricciones de acceso por los usuarios o grupos de usuario editamos el archivo de configuración **apache2** que se encuentra ubicado en la dirección de **etc/apache2/**.

Para realizar cambios en este archivo de configuración debemos ingresar como **root** digitando en la **Shell su root** y nos pide la clave del mismo.

Podemos elegir cualquier editor de Linux en mi caso elegí el **vim**.

#### 5.1.14 Creación de los usuarios para los repositorios en el archivo de configuración dav\_svn.passwd como root

Para que los usuarios puedan acceder al repositorio se debe abrir la **shell** y digitar la siguiente línea de comando:

Debemos ingresar como **root**:

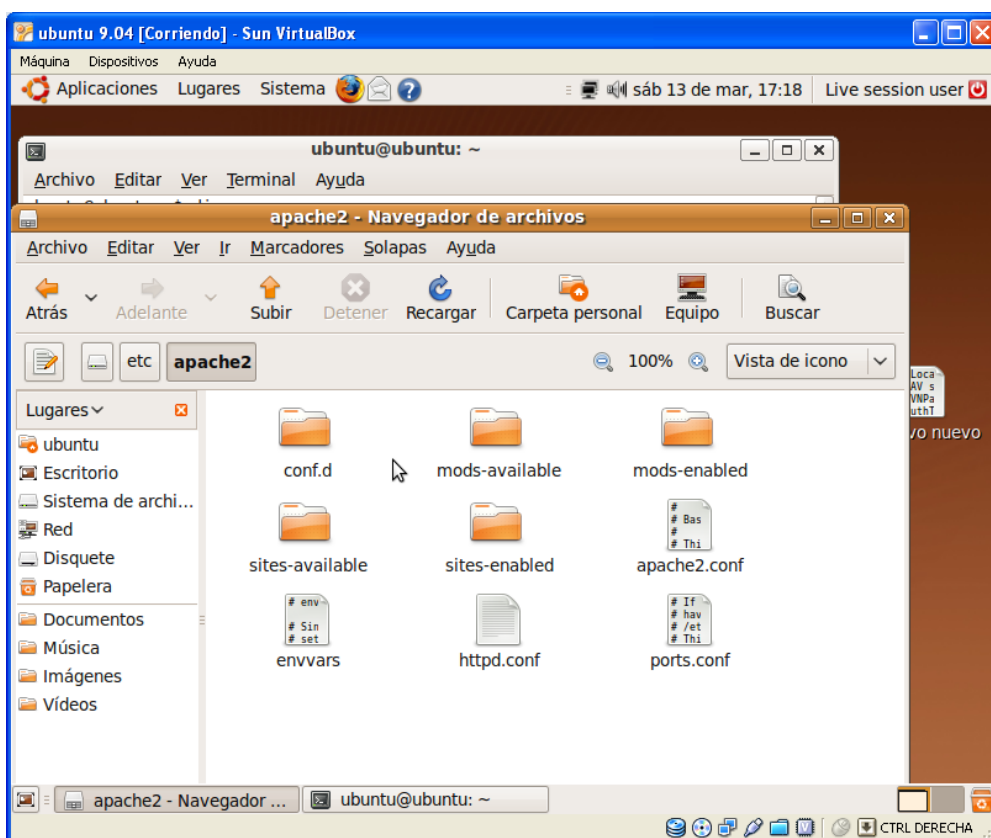
**sudo su root** y nos pide la clave del **root**

```
sudo htpasswd2-c /etc/apache2/dav_svn.passwd us1
```

Seguidamente nos pide la clave y reiterar nuevamente la clave.

Para el primer usuario que autoricemos debemos especificar el modificador **-c** para crear los usuarios y contraseñas por primera vez. Después podremos añadir tantos usuarios como queramos sin especificar **-c**, tal y como muestra la siguiente línea.

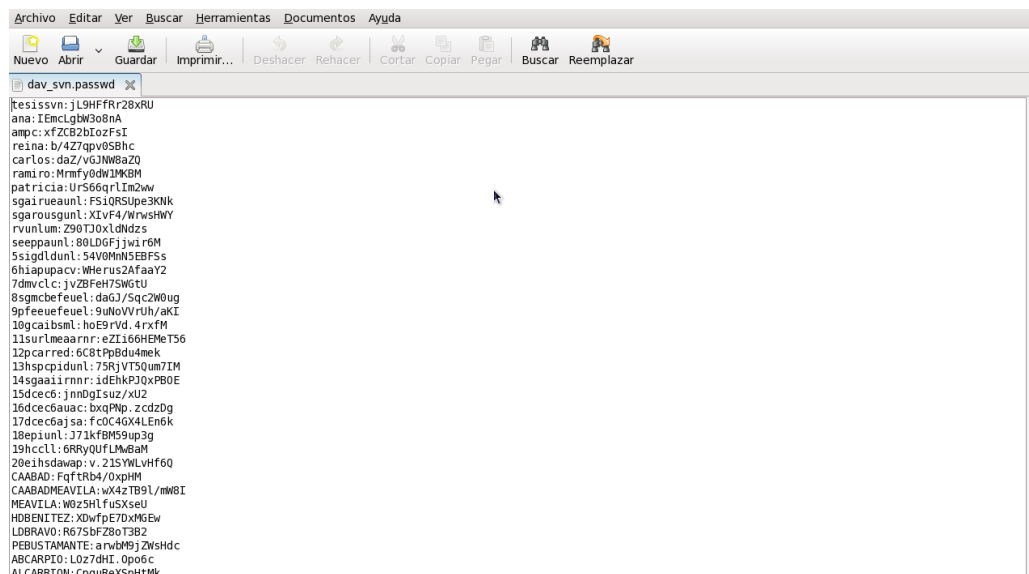
```
sudo htpasswd2 /etc/apache2/dav_svn.passwd us1 ver. ( Figura 5.10)
```



**Figura 5.11: Archivos de configuración del apache2**

Los archivos principales de configuración de la Subversion es:

- **apache2.conf**
- **dav\_svn.passwd**



**Figura 5.12: Configuración del repositorio con restricciones  
dav\_svn.passwd**

Inicio del configuración del archivo de dav\_svn.passwd

```
tesissvn:jL9HFfRr28xRU
ampc:xfZCB2bIozFsI
ramiro:Mrmfy0dW1MKBM
patricia:UrS66qrLIIm2ww
sgairueaunl:FSiQRSUpe3KNk
sgarousgunl:XIVF4/WrwsHWY
rvunlum:Z90TJOxldNdzs
seeppaunl:80LDGFjjwir6M
5siglddunl:54V0MnN5EBFSs
6hiapupacv:WHerus2AfaaY2
7dmvcl: jvZBFH7SWGtU
8sgmcbefeuel:daGJ/Sqc2W0ug
9pfeeuefeuel:9uNoVVrUh/aKI
10gcaibsm: hoE9rVd.4rxfM
11surlmeaarnr:eZi166HEMeT56
12pcarred:6C8tPpBdu4mek
13hspcpidunl:75RjVT5Qum7IM
14sgaaiirnnr: idEhkPJQxPBOE
15dcec6: jnnDgIsuz/xU2
16dcec6auac: bxqPNp.zcdzDg
17dcec6ajsa: fcOC4GX4LEn6k
18epiunl: J71kfBM59up3g
19hccll: 6RRyQUfLMwBaM
20eihsdawap: v.21SYWLvHf6Q
CAABAD: FqftRb4/OxpHM
CAABADMEAVILA: wX4zTB9l/mW8I
MEAVILA: W0z5HlfuSXseU
HDBENITEZ: XDwfpE7DxMGEw
LDBRAVO: R67SbfZ8oT3B2
PEBUSTAMANTE: arwbMbjZWshdc
ABCARPIO: L0z7dHt. Opo6c
ALCARRION: CoouRexSHtMk
```



LDBRAVO:R67SbFZ8oT3B2  
PEBUSTAMANTE:arwbM9jZWshdc  
ABCARPIO:LOz7dHI.Opo6c  
ALCARRION:CpquReXSpHtMk  
FJCASTILLO:Q2e.0LTVAcsw  
RCCONDOY:oloI6QiGFr6Co  
GRCORREA:TODRvgg2onGzs  
MLCUENCA:82qRLXlL1c/Kg  
HCCUESTA:X0i9iiY5RJQLU  
ELESPINOSA:MKilFt4kW/zPI  
EGESPINOZA:LitBlOrf90FpE  
LAFLORES:VogvC513.9bfQ  
LAGONZALEZ:q.KaU5QgixcZc  
GSGONZALEZ:wJUaWhLxU023Y  
REGUAMAN:090mkFvB/KRM2  
JMINIGUEZ:c76LmMDX7Qt5E  
LHJARAMILLO:47DjHNzFz./zM  
DFJIMENEZ:BSZkHZh6H2wG6  
JJLOAYZA:RdqV6FTTrMltXo  
REMACAS:5Q8TrP.1CXNDY  
LAMAZA:Ta1jP9.6jiglc  
VVMERINO:IEKM6RIpyNOWo  
SMMIRANDA:eqr/9UkBd13CQ  
DAMOROCHO:ddibmHRLkyghE  
GMNARVAEZ:QDgwXmxf5MSII  
KEORTEGA:lqTo2V10HhO2c  
WPPACHECO:eGr75nL4qSV/o  
JMPADILLA:iZJRV28iWMsUk  
RFROJAS:aJGGzwH2uUae  
RNVACACELA:gBy1Esuh5WFIQ  
TCVALAREZO:o4p6V6pp1rvuU  
JGVIRACUCHA:jalOOW/OmYHy6  
LXALVARADO:BfvaKWgPxgcBM  
AEARMIJOS:Us4sX3MqhJ47w  
IJBRICENO:bsb42H4QDWZJo  
MKCANDO:8hHgRK9.vorsE  
LICAPA:hW6oGrkIA60uc  
JMCARPIO:BGixGpzzLH/5c  
LECARTUCHE:MJ9Fpn.q8Sq6g  
EMCASTILLO:c9cZ9U5mliyZU  
JJCEVALLOS:yzRJjIgr3/0x.  
JLGUALAN:nZoEFQqPyR6hU  
DOGUALOTUNA:85MihuVDFfhto  
CAINIGUEZ:.lwQQB1wYhzks  
NEJARAMILLO:wy1GX7sl.6PC.  
MGJIMENEZ:AQYW3wDiyGw5U  
AELIMA:nCI.xg41bgiZQ  
IALOPEZ:pUszg8FffQRHk  
APLUNA:ELY.CkGOY0mMc  
JGMALDONADO:Fkgs0R2.3TW5c  
LCMATO:MBLbSeQPH6frU  
JLMEJIA:YC84G8PsMlca2  
KLNAMICELA:PJS/xqHI.UkZI  
JMNARVAEZ:aW8xF2W6O2WZE



PGORTEGA: lau.4/Q2Rlvyc  
CEPALACIOS: b0J6mY6tAG7vU  
CPPEREIRA: 6MP2v5upPyf9Q  
JAPONCE: zYiV7iUZs/Qjo  
EFPULLAGUARI: pxK12QC6Hw2wA  
JCRAMIREZ: AN4Zk.xUE8ygU  
GMRODRIGUEZ: o4KBE.wGqZ7BM  
DJROMERO: hoSB3CzTc.Qkk  
YPROMAN: VaSbJpu/VcoN6  
MCSALINAS: K6Y90E86qZ.UE  
MASILVA: 05ohZPAamc40c  
LJSINCHE: J0OLeankliDJY  
LASOTO: 7T8OufO4TmuEg  
VATIGRE: Ns4ErTxALCun.  
MMZAMBRANO: eEoA0l1K1VXCk  
patricio: AGdI/ixZd3Ai2  
hernan: WyO4FPQ0iQIIY  
milton: PZWxLCgNmCnGo  
marco: Ko9Vjxwe9ZLmI

#### **Fin del configuración del archivo de dav svn.passwd**

### **5.1.15 Configuración de acceso a usuarios por niveles de los repositorios**

Para la configuración de accesos a los repositorios, a los diferentes usuarios se explicara paso a paso el archivo de configuración del apache y la creación del archivo **svn\_access\_control**, el cual nos permitirá el control de acceso.

#### **Inicio del archivo de configuración svn\_access\_control**

#se defines los grupos  
[groups]

svn=ampc, patricio  
desarrollador=tesissvn, ana, carlos

todos=@svn, @desarrollador  
estudiantes=reina

# estudiantes de la carrera de ingeniería en sistemas quinto modulo a y b  
# año 2009

estudenstB=LXALVARADO, AEARMIJOS, IJBRICENO, MKCANDO, LICAPA,  
JMCARPIO, LECARTUCHE, EMCASTILLO, JJCEVALLOS, JLGUALAN,  
DOGUALOTUNA, CAINIGUEZ, NEJARAMILLO, MGJIMENEZ, AELIMA, IALOPEZ,  
APLUNA, JGMALDONADO, LCMATO, JLMEJIA, KLNAMICELA, JMNARVAEZ,  
PGORTEGA, CEPALACIOS, CPPEREIRA, JAPONCE, EFPULLAGUARI, JCRAMIREZ,



GMRODRIGUEZ, DJROMERO, YPROMAN, MCSALINAS, MASILVA, LJSINCHE,  
LASOTO, VATIGRE, MMZAMBRANO, marco, milton, hernan

estudenstA=CAABAD, MEAVILA, HDBENITEZ, LDBRAVO, PEBUSTAMANTE,  
ABCARPIO, ALCARRION, FJCASTILLO, RCCONDOY, GRCORREA, MLCUENCA,  
HCCUESTA, ELESPINOSA, EGESPINOZA, LAFLORES, LAGONZALEZ, GSGONZALEZ,  
REGUAMAN, JMINIGUEZ, LHJARAMILLO, DFJIMENEZ, JJLOAYZA, REMACAS,  
LAMAZA, VVMERINO, SMMIRANDA, DAMOROCHO, GMNARVAEZ, KEORTEGA,  
WPPACHECO, JMPADILLA, RFROJAS, RNVACACELA, TCVALAREZO,  
JGVIRACUCHA, marco, milton, hernan

#tesis

#acceso de lectura para todo el mundo

[/]

\*.\*=r

[/]

@svn=rw

#estudiantes puede leer pero solo desarrollador puede modificar

[rvunlum:/]

@estudiantes=r

@desarrollador=rw

[programas:/]

@svn=rw

# pruebas

[repo1:/]

@todos=r

# programas

[repo\_programas:/]

@svn=rw

[repo\_avg:/]

@svn=rw

#tesis

[rvunlum:/]

rvunlum=rw

@estudiantes=r

[sgairueaunl:/]

sgairueaunl=rw

@estudiantes=r

[sgarousgunl:/]

sgarousgunl=rw

@estudiantes=r

[seeppaunl:/]





seeppaunl=rw  
@estudiantes=r

[5sigdldunl:/]  
5sigdldunl=rw  
@estudiantes=r

[6hiapupacv:/]  
6hiapupacv=rw  
@estudiantes=r

[6hiapupacv:/]  
6hiapupacv=rw  
@estudiantes=r

[7dmvclc:/]  
7dmvclc=rw  
@estudiantes=r

[8sgmcbefeuel:/]  
8sgmcbefeuel=rw  
@estudiantes=r

[9pfeeuefeuel:/]  
9pfeeuefeuel=rw  
@estudiantes=r

[10gcaibsm1:/]  
10gcaibsm1=rw  
@estudiantes=r

[11surlmeaarnr:/]  
11surlmeaarnr=rw  
@estudiantes=r

[12pcarred:/]  
12pcarred=rw  
@estudiantes=r

[13hspcpidunl:/]  
13hspcpidunl=rw  
@estudiantes=r

[14sgaaiirnnr:/]  
14sgaaiirnnr=rw  
@estudiantes=r

[15dcec6:/]  
15dcec6=rw  
@estudiantes=r

[16dcec6auac:/]  
16dcec6auac=rw  
@estudiantes=r



[17dcec6ajsa:/]  
17dcec6ajsa=rw  
@estudiantes=r

[18epiunl:/]  
18epiunl=rw  
@estudiantes=r

[19hccll:/]  
19hccll=rw  
@estudiantes=r

[20eihsdawap:/]  
20eihsdawap=rw  
@estudiantes=r

[21SistemaFinaciero:/]  
@estudiantes=r

[22factividadydiseño:/]  
@estudiantes=r

[23desarrollodeunSistema:/]  
@estudiantes=r

[24Reestructuracióndelportal:/]  
@estudiantes=r

[25sistemadegestión:/]  
@estudiantes=r

[26sistemainformático:/]  
@estudiantes=r

[27informacióndelsistema:/]  
@estudiantes=r

[28aplicaciónweb:/]  
@estudiantes=r

[29sistemadeplanificación:/]  
@estudiantes=r

[30desarrollaplicaciónweb:/]  
@estudiantes=r

[31herramientacotización:/]  
@estudiantes=r

[32diseñoherramientagrafica:/]  
@estudiantes=r

[33diseñosistema:/]



@estudiantes=r

[34softwaredetutorias:/]

@estudiantes=r

[35aplicacionweb:/]

@estudiantes=r

[36controlgarantias:/]

@estudiantes=r

.....

[105siga:/]

@estudiantes=r

[128svagb:/]

@estudiantes=r

[129maubu:/]

@estudiantes=r

[130gasparin:/]

@estudiantes=r

[131sdtgpl:/]

@estudiantes=r

[quintoA:/]

@estudenstA=rw

marco, milton, hernan=rw

[quintoB:/]

@estudenstB=rw

marco, milton, hernan=rw

#### **5.1.16 Configuración del archivo *svn\_access\_control* para dar permiso de acceso para lectura y escritura por grupo a los repositorios**

La definición de los accesos por grupo a un repositorio se realiza mediante la línea ***AuthzSVNAccessFile***, en nuestro caso es ***/etc/apache2/svn\_access\_contro***, en este archivo se definen grupos de accesos y permisos de acceso a cada repositorio.



Para crear el archivo de configuración de grupos de accesos se realiza de la siguiente forma:

Ingresamos a la Shell.

Realizamos los cambios como súper usuario (**root**).

**sudo su root** y nos pide la clave para ingresar como root.

Luego digitamos **sudo vim /etc/apache2/svn\_access\_control**.

Nos abre un editor y digitamos la siguiente configuración.

# Se definen los grupos de trabajo cabecera [groups]:

[groups]

# Se crea grupos de usuarios:

svn=user1, user2

guest=user3, user4

devteam=user5,user6

deveryone=@ svn, @guest

# Acceso de lectura para todo el mundo:

[/]

\* = r

# Acceso de escritura para todo el mundo:

\*=w

# Acceso de lectura y escritura a un grupo de usuario:

@svn=rw

# Acceso de lectura y escritura a todos los repositorios para un usuario:

user1=rw

# Acceso a un solo repositorio de lectura o escritura:

# guest pueden leer pero solo devteam actualiza en el repositorio repo

[nombre\_repo:/]

# el grupo de usuario devteam tiene la opción de lectura y escritura

@devteam=rw

# el grupo de usuario guest tiene la opción de lectura al repositorio:

@guest=r



```
# Todo el mundo puede leer el repo test pero solo
# user5 lo puede modificar
[test:/]
user5=rw
@everyone=r
# Solo el grupo devteam tiene acceso total al
# repositorio bcol. Guest no tienen ningun acceso
[bcol:/]
@devteam = rw
```

**[group]** se define el inicio de la configuración de grupos

**svn** se crea los grupos de usuario

**svn =** se adiciona los usuario al grupo svn

**[/]** todos los grupos o usuarios que se encuentren debajo tienen la opción de acceder a todos los repositorios creados en este directorio padre.

**[nombre\_repo:/]** todos los que están debajo de esta línea tienen acceso solo a este repositorio y sus contenidos.

**[nombre\_repo/directorio:/]** todos los que están debajo de esta línea tienen acceso solo a este repositorio y sus contenidos.

**@grupo=** se adiciona si es un grupo de usuario creado al inicio se le puede dar permiso de lectura o escritura ejemplo:

**[nombre\_repo:/]**

**@grupo1=rw**

**@grupo2=r**

**@grupo3=w**



**user1=** Se le puede asignar a un usuario un repositorio, no necesita estar creado en el archivo siempre y cuando sea creado por el servidor de apache con las sentencias `sudo htpasswd2 /etc/apache2/dav_svn.passwd` **us1**, ejemplo:

[nombre\_repo:/]

user1=rw

user2=r

user3=w

Luego guardamos el archivo y reiniciamos el servidor de apache. Para que Apache tome los cambios se deben habilitar los módulos de svn y reiniciar el servicio con los siguientes comandos:

```
sudo a2enmod dav
```

```
sudo a2enmod dav_svn
```

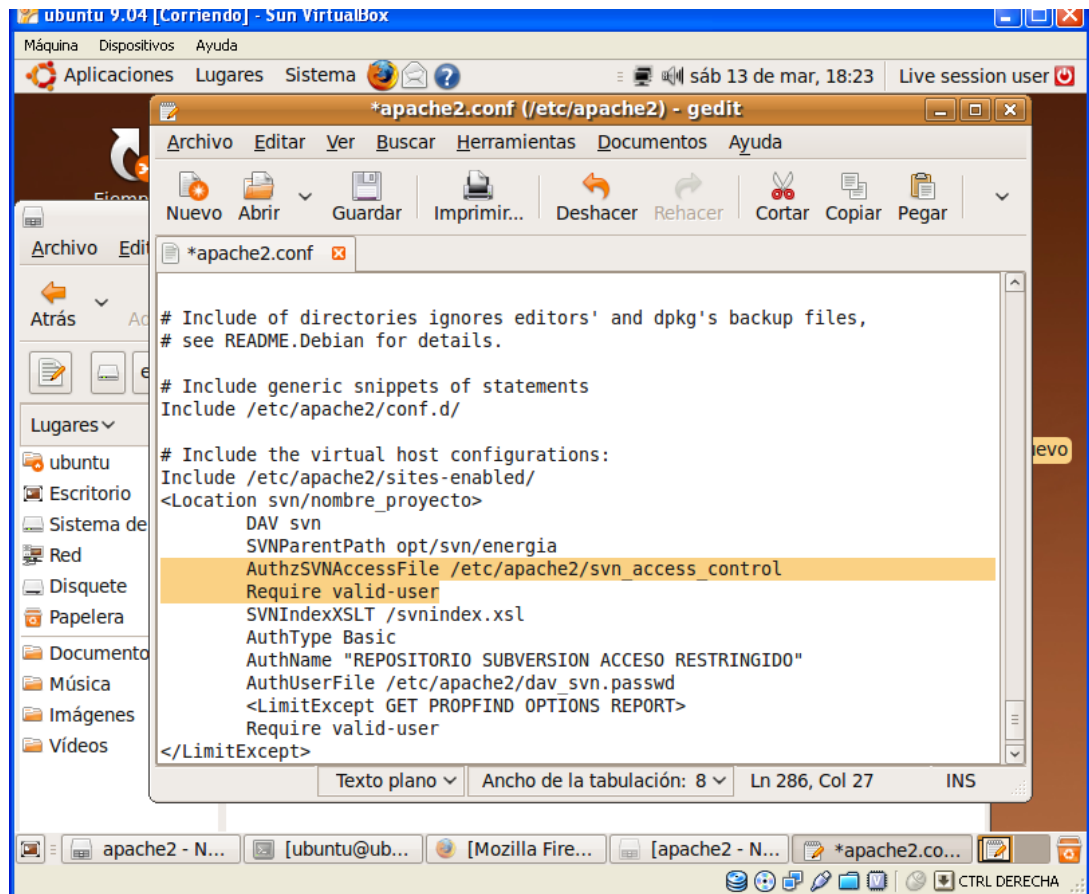
```
sudo /etc/init.d/apache2 force-reload
```

**Nota:** se debe reiniciar el archivo **svn\_access\_controada** vez que modifiquemos el archivo.

Para que subversion tome los cambios se debe modificar el archivo de **apache2.conf**.

Para realizar cambios en este archivo de configuración debemos ingresar como **root** mediante el comando **su root** y nos pide la clave del mismo.

Podemos elegir cualquier editor de Linux en mi caso elegí el **vim** abrimos la **Shell** y digitamos **sudo vim/etc/apache2/apche2.conf** ver (Figura 5.13)



**Figura 5.13: Vista Archivo apache2 con restricciones  
svn\_access\_control**

<Location /svn/proyecto>

DAV svn

SVNParentPath /opt/svn/energia/tesis

AuthzSVNAccessFile /etc/apache2/.svn\_access\_control

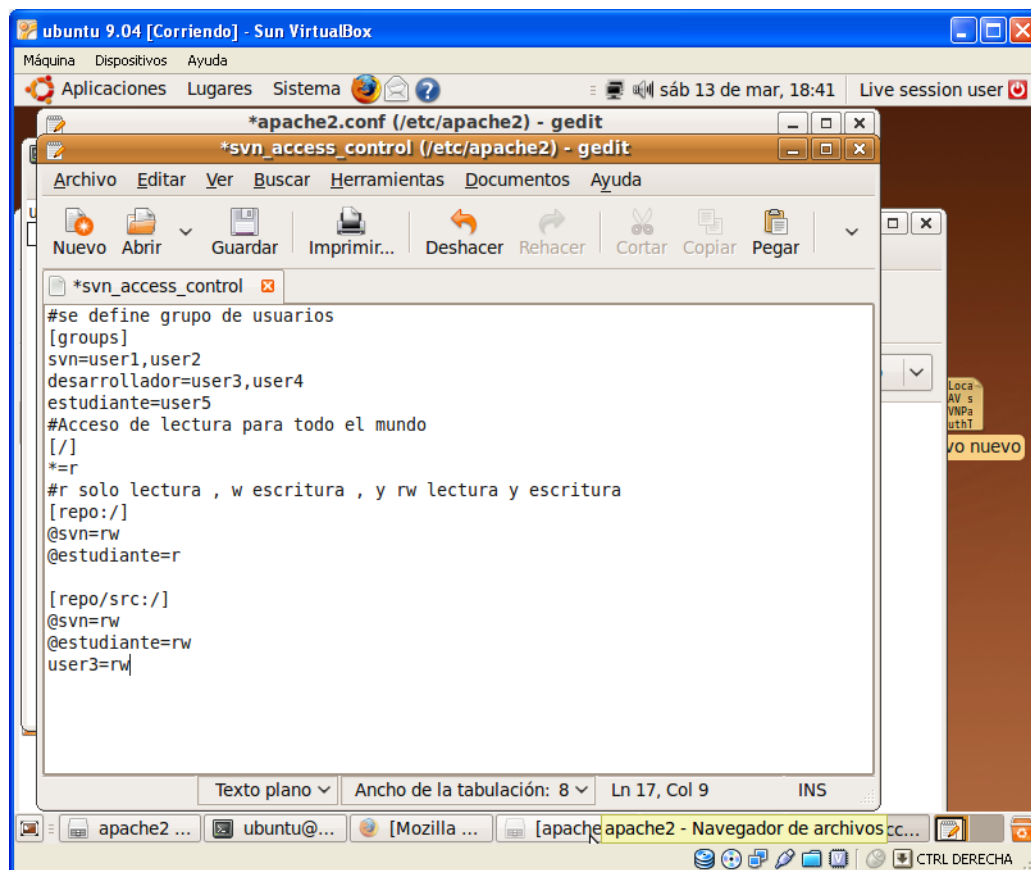
Require valid-user

AuthType Basic

AuthName "REPOSITORIO SUBVERSION UNL AEIRNNR ACCESO RESTRINGIDO"

AuthUserFile /etc/apache2/dav\_svn.passwd

</Location>



**Figura 5.14: Configuración del repositorio con restricciones  
svn\_access\_control**

Guardamos y **reiniciamos** el servidor de apache para que los cambios se realicen. Mediante las siguientes líneas de código `sudo /etc/init.d/apache2 restart`.

Para comprobar si está bien la configuración abrimos el navegador que tengamos y escribimos la **http://ip de la máquina**. Y nos debe salir la siguiente vista.

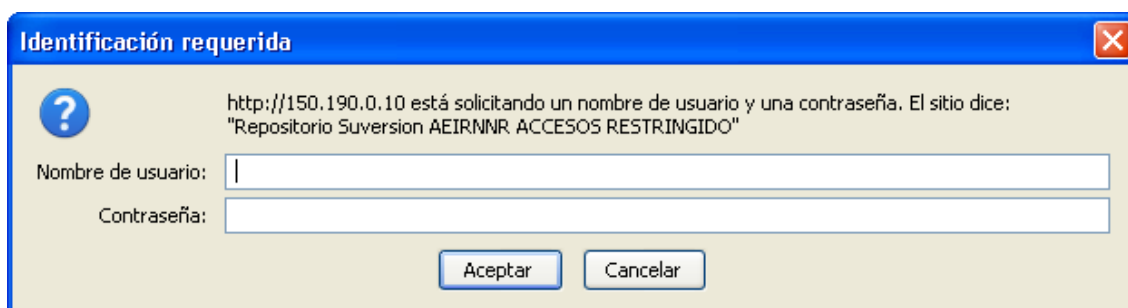
Esta definición solo permite acceso a usuarios validos (Requirevalid-user), si se quieren permitir usuarios anónimos o colocar otro tipo de restricciones a los accesos debe consultar la documentación de Apache.

**Nota:** Fíjense en los cuatro valores a modificar:



- **AuthName:** Mensaje que muestra el usuario final al momento de acceder al repositorio.
- **SVNParentPath:** Debe apuntar al directorio padre de los repositorios.  
**AuthUserFile:** archivo con la definición de usuarios y los **passwords**.
- **AuthzSVNAccessFile** Debe apuntar al directorio donde tengamos el archivo de configuración por grupo de usuarios **/etc/apache2/.svn\_access\_conrol**.

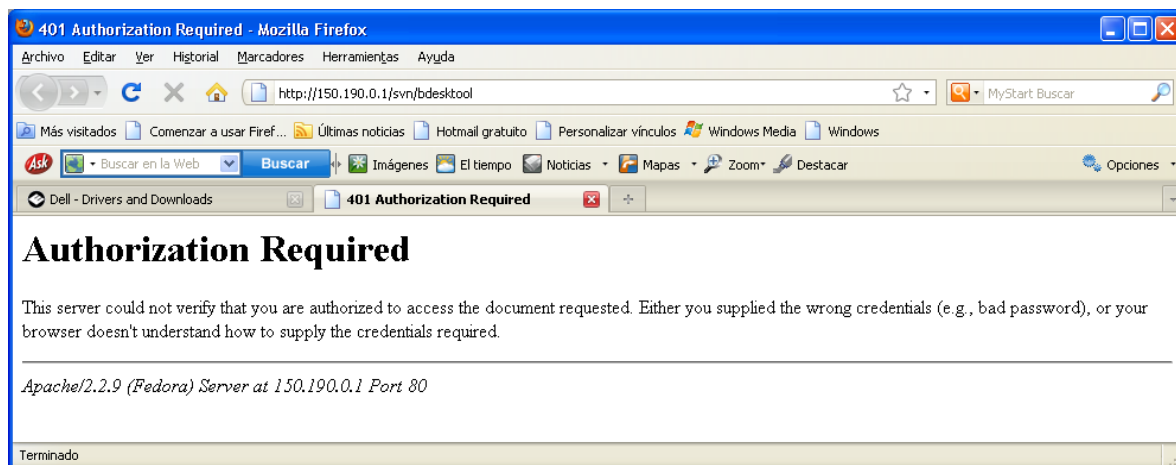
Para comprobar si está bien la configuración abrimos el navegador que tengamos y escribimos la **http://ip de la máquina**. Y nos debe salir la siguiente vista figura



**Figura 5.15: Vista protocolo http de apache para acceder al repositorio SVN.**

Al momento que accedemos al repositorio de subversión <http://ipdelservidos/svn/repo> nos presenta la vista de la Figura 5.15 que nos pide el nombre de usuario y la clave, si digitamos la clave correcta y el nombre de usuario pero no tiene permiso al

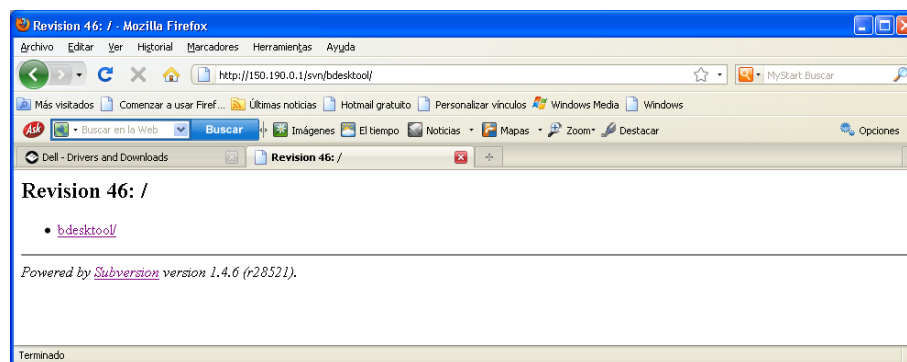
repositorio, nos envía a una página de error de permiso para el usuario ver (Figura 5.16)



**Figura 5.16: Vista desde el protocolo http de apache**

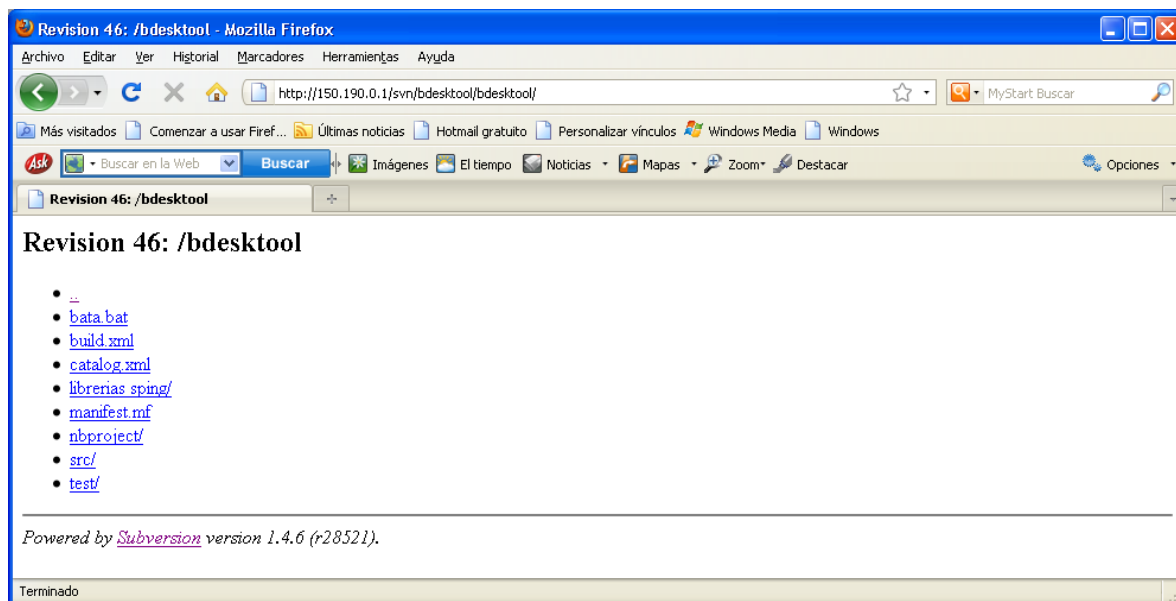
Si no ingresamos el nombre y la clave y escogemos la opción cancelar, nos envía a una página de error de permiso para el usuario que necesita la autorización como indica la Figura 5.17.

Si se ingresa la clave correcta nos dará la vista siguiente ver Figura 5.17



**Figura 5.17: Vista desde el protocolo http de apache acceso correcto**

Realizamos un click en el nombre del repositorio y nos muestra todo los contenidos del mismo y nos dará la vista siguiente ver figura 5.18.



**Figura 5.18. Vista desde el protocolo http de apache archivos Subversionados**

**Nota:** Si no muestra esta página debe revisar la configuración nuevamente.

#### **5.1.17 Configuración del archivo apache2.conf para la habilitación el css para acceder mediante http://**

Esta habilitación del archivo **css** para subversion en Ubuntu 9.04 (**opcional**) para la presentación del usuario final al momento de acceder por el protocolo **http** de apache.

Para habilitar esta opción se debe realizar los siguientes pasos:

Abrir la **shell**, tenemos que iniciar como **root**.

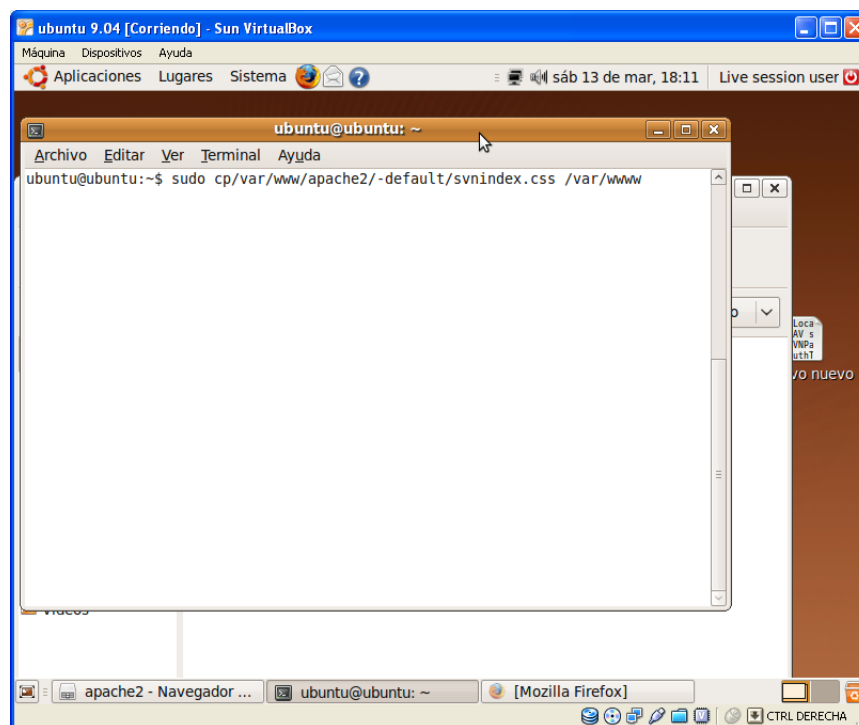
Debemos copiar los archivos de **svnindex.css** y **svnindex.xml** a la carpeta **/var/www**

Mediante las siguientes líneas de código:

```
sudo cp /var/www/apache2-default/svnindex.css /var/www
```

```
sudo cp /var/www/apache2-default/svnindex.xml /var/www
```

Modificamos el archivo de **apache.conf**



**Figura 5.19: Configuración del repositorio http css**

Para que subversion tome los cambios se debe modificar el archivo de **apache2.conf**

Para realizar cambios en este archivo de configuración debemos ingresar como **root** digitando en la **Shell su root** y nos pide la clave del mismo.

Podemos elegir cualquier editor de Linux en mi caso elegí el **vim**.

Abrimos la Shell y digitamos `sudo vim etc/apache2/apche2.conf`

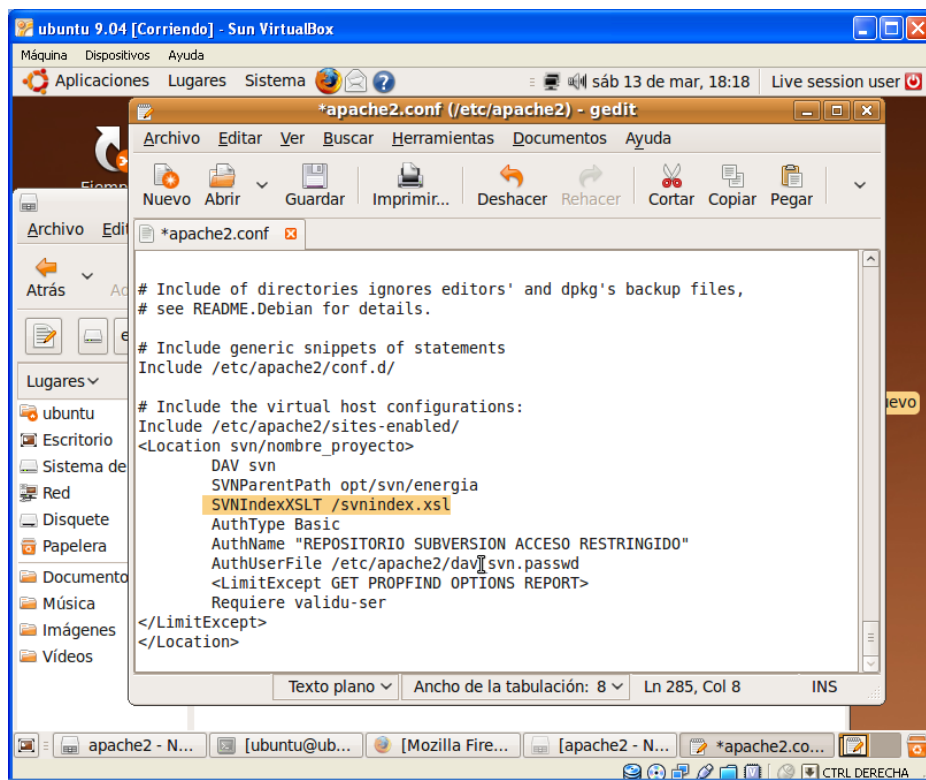


Figura 5.20: Configuración del repositorio http

<Location /svn/proyecto>

DAV svn

SVNParentPath /opt/svn/energia/tesis

AuthzSVNAccessFile /etc/apache2/.svn\_access\_control

Require valid-user

SVNIndexXSLT /svnindex.xsl

AuthType Basic

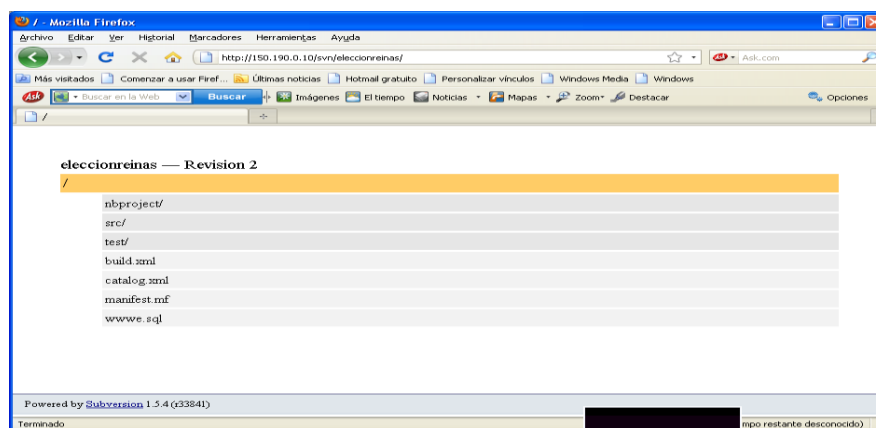
AuthName "REPOSITORIO SUBVERSION UNL AEIRNNR ACCESO RESTRINGIDO"

AuthUserFile /etc/apache2/dav\_svn.passwd

</Location>

Luego guardamos y reiniciamos el servidor de apache para que los cambios se realicen. Mediante las siguientes líneas de código **sudo /etc/init.d/apache2 restart**

Para comprobar si está bien la configuración abrimos el navegador que tengamos y escribimos la **http://ip de la máquina**. Y nos debe salir la siguiente vista ver figura 5.21.



**Figura 5.21: Vista http del repositorio utilizando css**

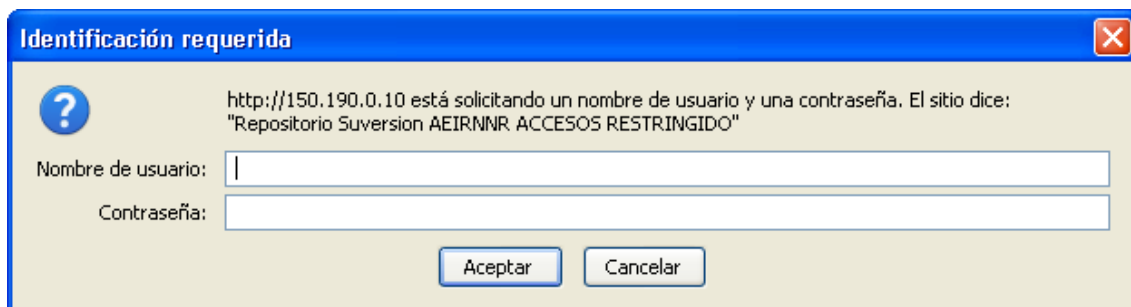
Esta definición solo permite acceso a usuarios validos (Require valid-user), si se quieren permitir usuarios anónimos o colocar otro tipo de restricciones a los accesos debe consultar la documentación de Apache. Fíjense en los cinco valores a modificar:

- **AuthName:** Mensaje que muestra el usuario final al momento de acceder al repositorio.
- **SVNParentPath:** debe apuntar al directorio padre de los repositorios.
- **AuthUserFile:** archivo con la definición de usuarios y los passwords.
- **AuthzSVNAccessFile:** Debe apuntar al directorio donde tengamos el archivo.
- **SVNIndexXSLT/svnindex.xsl** habilita la opción del css de subversion configuración por grupo de usuarios **/etc/apache2/svn\_access\_control**.

Para comprobar si está bien la configuración abrimos el navegador que tengamos y escribimos la **http://ip de la máquina**. Y nos debe salir la siguiente vista ver Figura 5.22.

Al momento que accedemos al repositorio de subversión <http://ipdelservidos/svn/repo>.

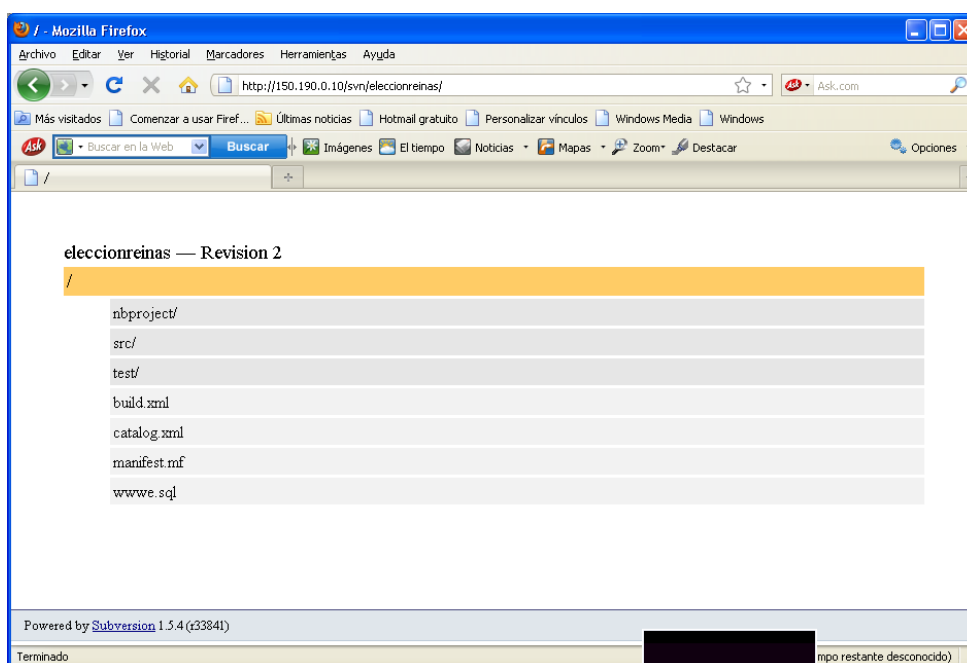
Nos presenta la vista de la figura 5.22 que nos pide el nombre de usuario y la clave si digitamos la clave correcta y el nombre de usuario pero no tiene permisos al repositorio, nos envía a una página de error de permiso para el usuario.



**Figura 5.22: Vista desde el protocolo http de apache**

Si no ingresamos el nombre y la clave y escogemos la opción cancelar nos envía a una página de error de permiso para el usuario que necesita la autorización como indica la Figura 5.21, si ingresamos la clave correcta nos da la vista Figura 5.22.

Realizamos un click en el nombre del repositorio y nos muestra todo los contenidos del mismo.



**Figura 5.23: Vista desde http**



Como se puede observar en la figura 6.18, la vista de la página web utilizado el servidor de apache ha cambiado en una forma más amigable para el usuario.

Si no muestra esta página debe revisar la configuración nuevamente.

## **5.2 Configuración de servidor de repositorios subversion svn**

Este documento describe la instalación y configuración del repositorio Subversion SVN. Es más como un lugar donde puede venir cuando sepa qué quiere hacer, pero no recuerde exactamente cómo hacerlo.

### **5.2.1 REQUERIMIENTOS DEL SOFTWARE Y HARDWARE**

#### **5.2.1.1 Requerimientos del hardware para la instalación SUBVERSION SERVIDOR**

Como requerimientos del hardware tenemos:

Requisito	Profesional
Sistema Operativo	Sistema operativo Linux ,(Ubuntu 9.04) o superior Linux Ubuntu 9.0.4 o superior, fedora, 10 o superior
Procesador	PC con un procesador de tipo Pentium  IV a 600 MHz  Se recomienda: 1 gigahercio (GHz)
RAM	512 MB RAM, se recomienda 1 GB o más
Vídeo	800 X 600, 256 colores  Se recomienda: 1024 X 768, color de alta densidad de 16 bits
Disco Duro	80 GB o más de espacio disponible dependiendo de los proyectos



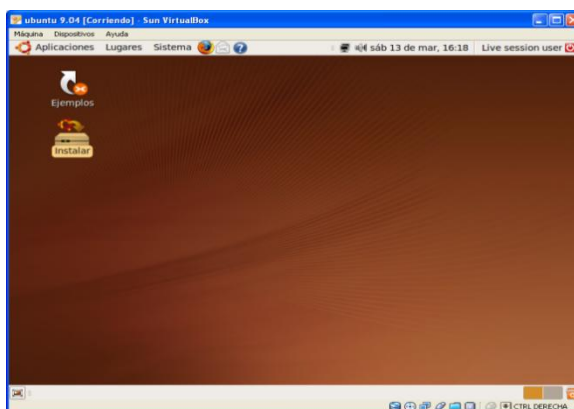
### 5.2.1.2 Requerimientos del software para la instalación SUBVERSION SERVIDOR

Como requerimientos del Software tenemos:

- Un sistema operativo Linux Ubuntu 9.0.4 o superior, fedora 10 en adelante.
- Soporte para protocolo TCP/IP.
- Suficiente espacio en disco duro para descomprimir, instalar los paquetes de subversion, y crear los repositorios que se van a crear en el servidor. Generalmente se recomienda un mínimo de 1GB dependiendo del proyecto.
- Cualquier lenguaje de programación.

## GUÍA DE INSTALACIÓN

Instalar un servidor de subversión en Ubuntu 9.04



**Figura 5.2.1: Vista Ubuntu 9.04**

Ingresa al sistema operativo llenando los campos **usuario= diego** y **contraseña= #####**, a continuación se presenta la pantalla del sistema operativo (Ver Figura 5.2.1).

Ingresa a la consola del servidor como **root**:

- `sudo apt-get install subversion subversion-tools`
- Apache y sus dependencias

```
sudo apt-get install apache2 apache2svn
```

```
sudo apt-get install subversion libapache2-svn
```

Otra forma de instalar los paquetes para la subversion es:

Sistema, gestor de paquetes **synaptic**, buscar **subversion-tools**, y **apacha2**. (Ver figura 5.2.3)

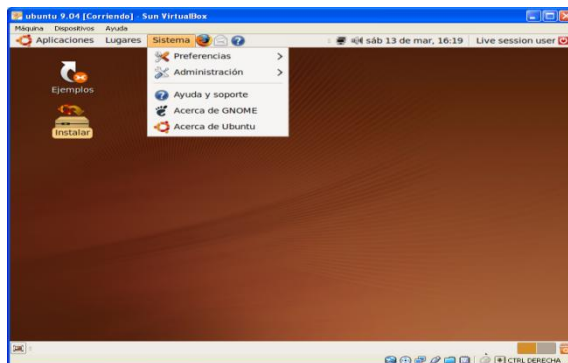
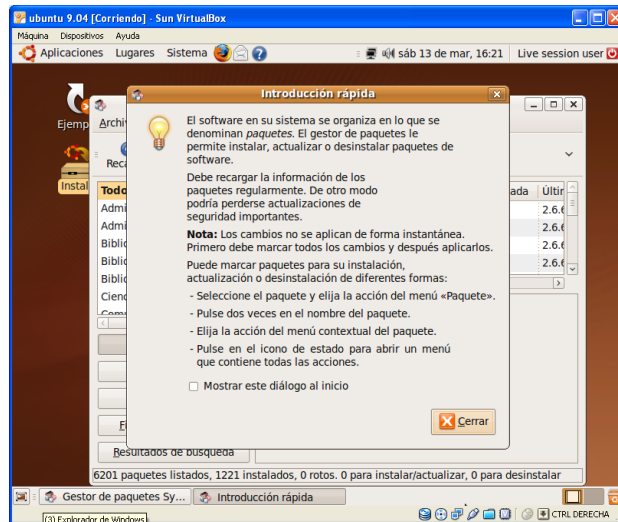


Figura 5.2.2: Vista Instalar SVN



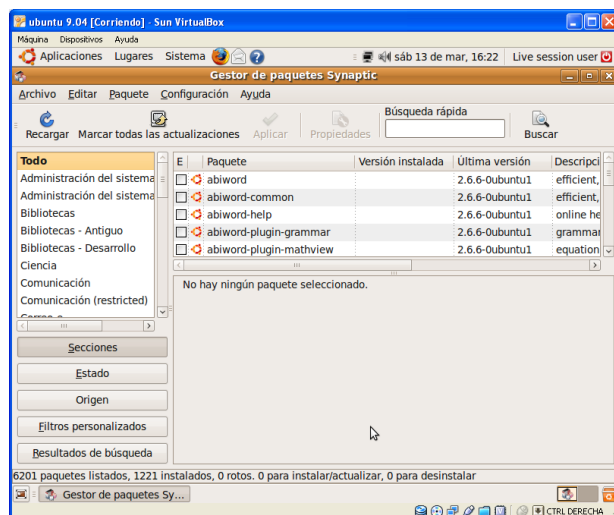
Figura 5.2.3: Vista abrir Gestor de paquetes synaptic

Se presenta un mensaje de información (Ver Figura 5.2.4) esta se presenta la primera vez de utilizar Synaptic, escogemos la opción Cerrar, luego se visualiza la pantalla del Gestor de Paquetes Synaptic (Ver Figura 5.2.5).



**Figura 5.2.4: Vista Mensaje de Información abrir Gestor de paquetes synaptic**

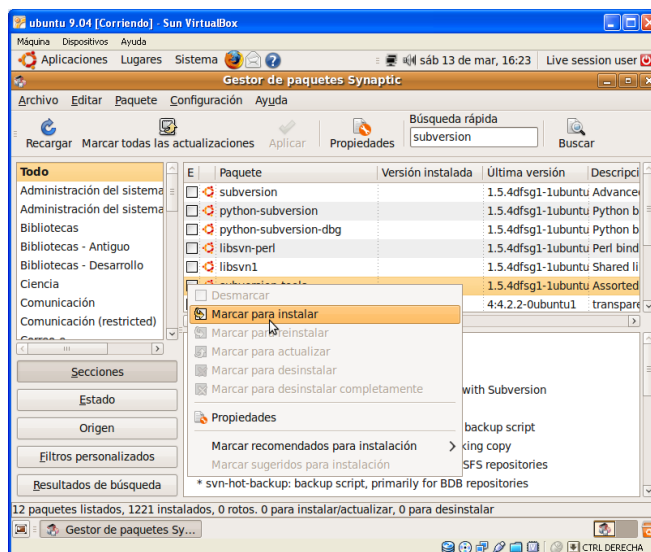
Busque programas para la instalación del servidor (Ver Figura 5.2.5).



**Figura 5.2.5: Vista buscar paquetes Subversión**

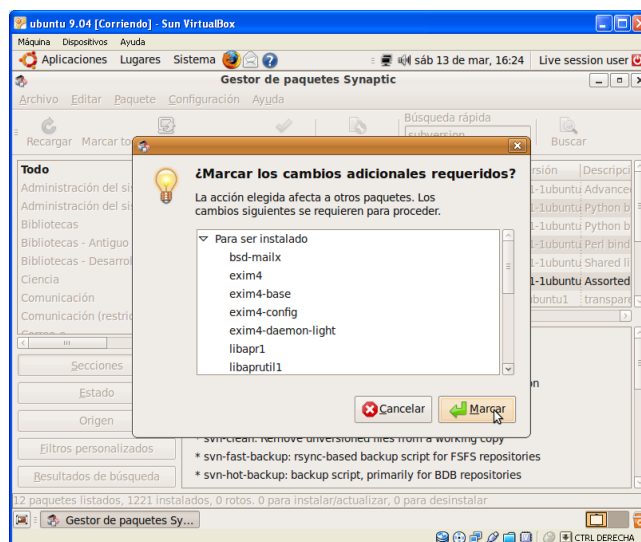
### 5.2.1.3 Instalación de Subversion

Escriba `subversion` en el campo búsqueda rápida y elija la opción **Buscar** (Ver Figura 5.2.6), luego se presentan los paquetes o dependencias correspondientes a la búsqueda realizada (Ver Figura 5.2.6).

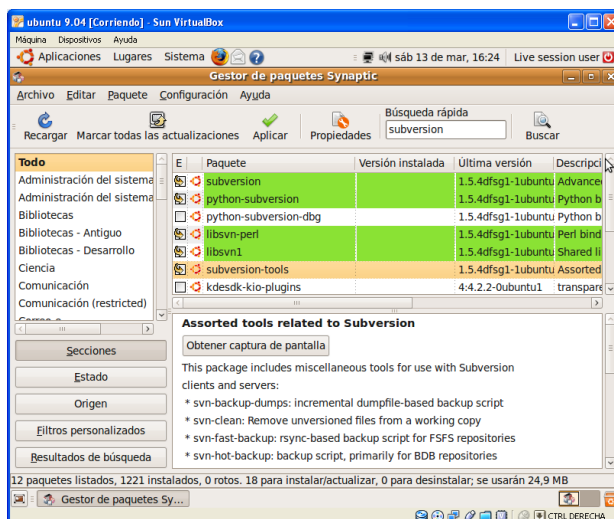


**Figura 5.2.6. Vista Seleccionar paquetes Subversion**

Seleccione el paquete que requiera instalar, y elija la opción Marcar para Instalar (Ver Figura 5.2.6), luego se presenta un mensaje de confirmación (Ver Figura 5.2.7) elija la opción Marcar, luego se puede visualizar todos los paquetes seleccionados para ser instalados (Ver Figura 8)

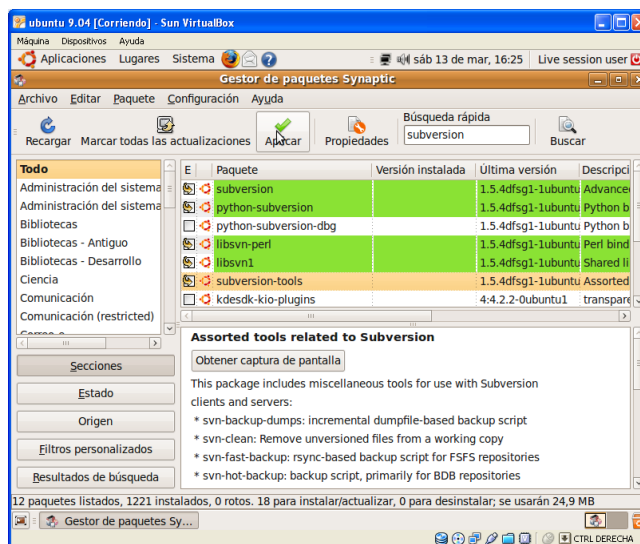


**Figura 5.2.7: Vista Instalación paquetes Subversion**

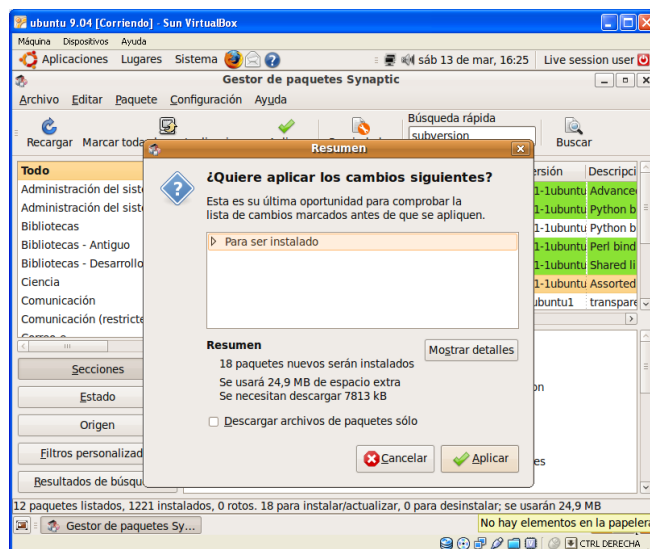


**Figura 5.2.8: Vista Selección de paquetes Subversion**

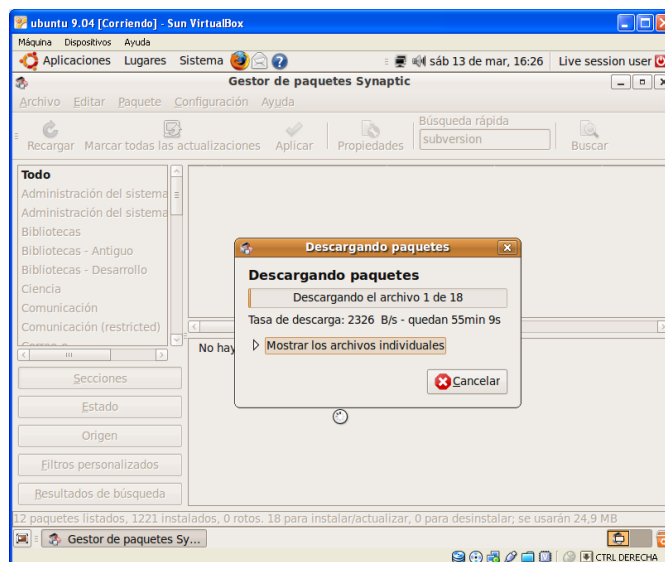
Empiece a descargar los paquetes, seleccione Aplicar (Ver Figura 5.2.9), luego se presenta un mensaje de confirmación para instalar los paquetes anteriormente seleccionados (Ver Figura 5.2.10) elija la opción Aplicar, a continuación se presenta una pantalla (Ver Figura 5.2.11) en donde se muestra el avance de la descarga.



**Figura 5.2.9: Vista Aplicar paquetes Subversion**

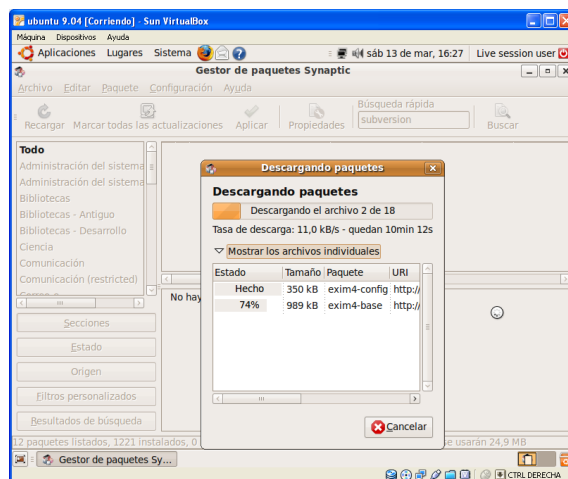


**Figura 5.2.10: Vista Aceptar paquetes Subversion**



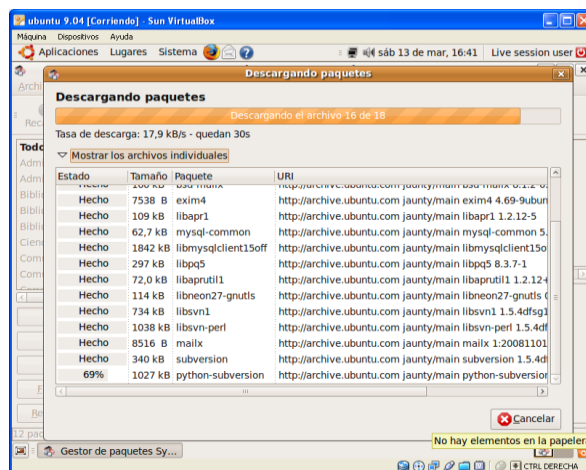
**Figura 5.2.11: Vista Descarga de paquetes Subversion**

Para visualizar el estado de la descarga de cada archivo (Ver Figura 5.2.12), seleccione la opción Mostrar los archivos individuales (Ver figura 5.2.11)



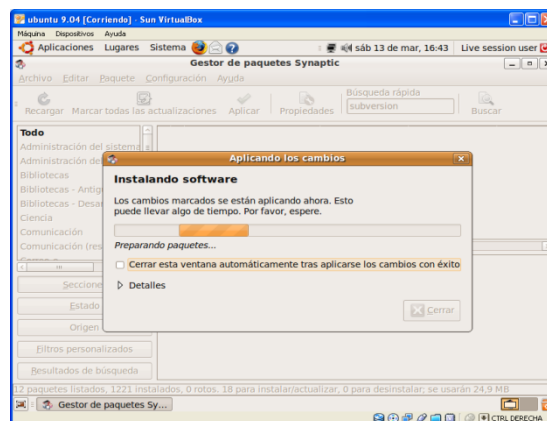
**Figura 5.2.12. Vista proceso de descarga de paquetes Subversion**

Visualice el estado de descarga de todos los paquetes (Ver Figura 5.2.13)



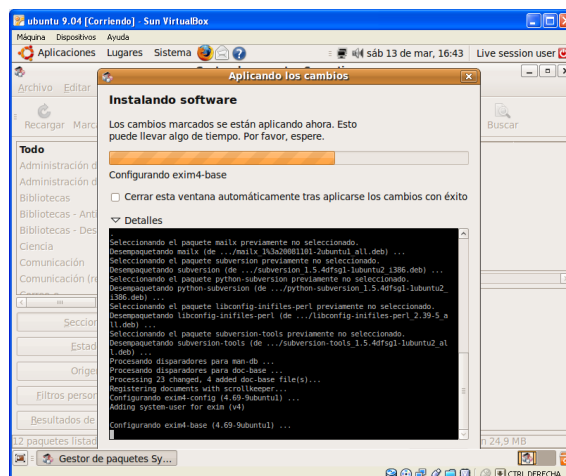
**Figura 5.2.13: Vista descarga detallada de los paquetes Subversion**

Visualice el estado de instalación de los paquetes o dependencias de la subversion (Ver Figura 5.2.14)



**Figura 5.2.14: Vista Instalando paquetes Subversion**

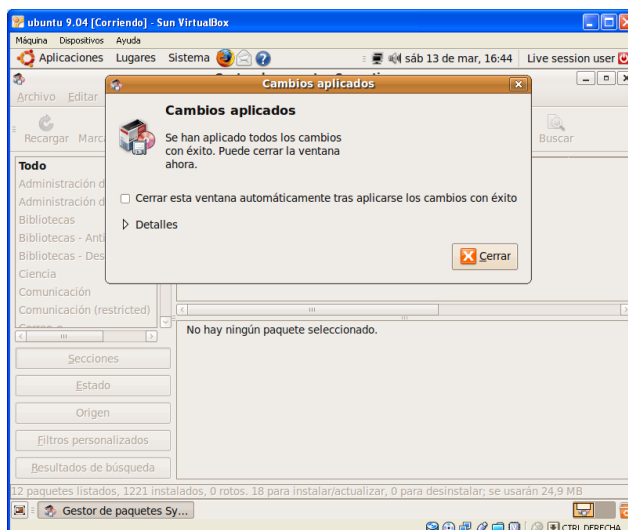
Para visualizar con mayor detalle la instalación de todos los paquetes elija la opción Detalles (Ver Figura 5.2.14), luego se visualizan todos los detalles correspondientes a la instalación de cada paquete subversión (Ver Figura 5.2.15)



**Figura 5.2.15: Vista proceso de instalación de paquetes Subversion**

Cuando ha finalizado la instalación se presenta un mensaje de información (Ver Figura 5.2.16) elija la opción Cerrar.

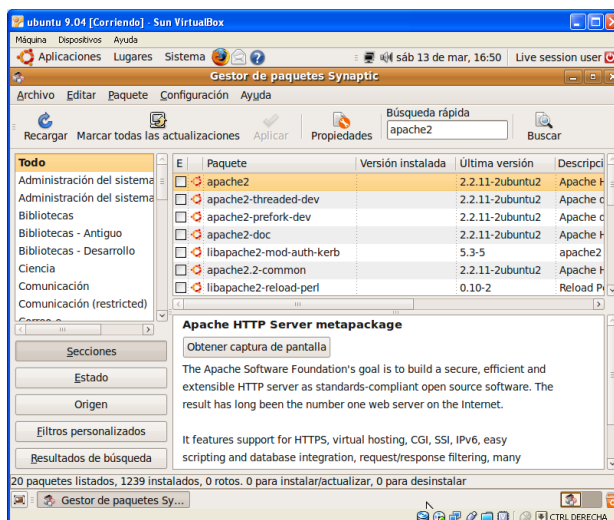




**Figura 5.2.16: Vista proceso de finalización del proceso de instalación de paquetes Subversion**

#### 5.2.1.4. Instalar Apache2

Escriba `apache2` en el campo búsqueda rápida y elija la opción Buscar (Ver Figura 5.2.17), luego se presentan los paquetes o dependencias correspondientes a la búsqueda realizada (Ver Figura 5.2.17).



**Figura 5.2.17: Vista proceso de marcación de paquetes apache2**

Seleccione el paquete que requiera instalar, y elija la opción Marcar para Instalar (Ver Figura 5.2.18), luego se presenta un mensaje de confirmación (Ver Figura 5.2.19) elija la opción Marcar, luego se puede visualizar todos los paquetes seleccionados para ser instalados (Ver Figura 5.2.20)

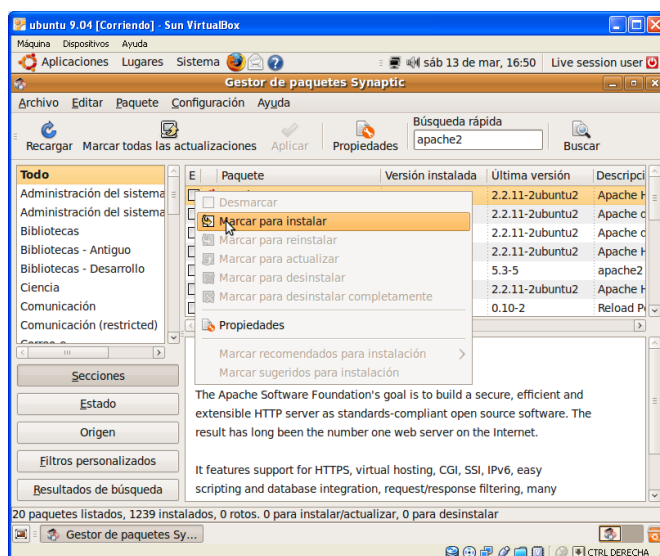


Figura 5.2.18: Vista proceso de descarga de paquetes apache2

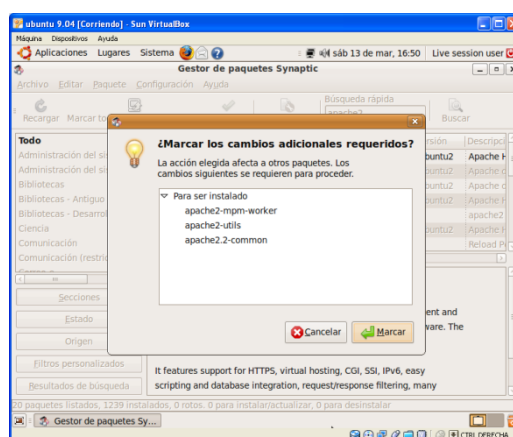
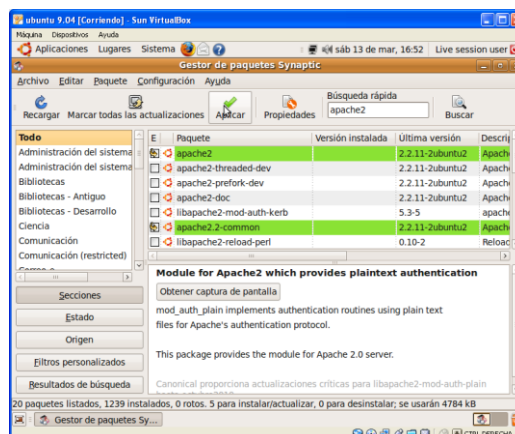
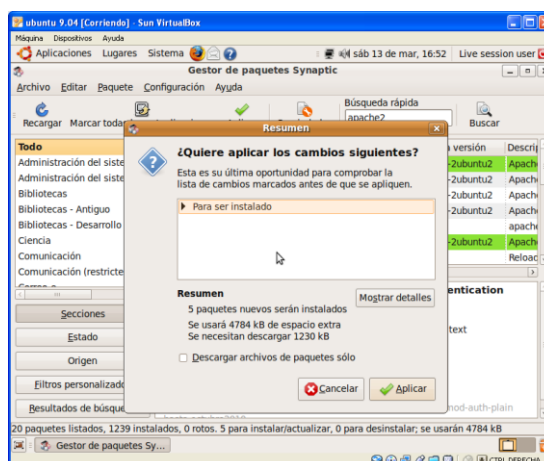


Figura 5.2.19: Vista proceso de descarga de paquetes apache2 marcados



**Figura 5.2.20: Vista proceso de aplicar los paquetes para la descarga de apache2**

Empiece a descargar los paquetes, seleccione Aplicar (Ver Figura 5.2.20), luego se presenta un mensaje de confirmación para instalar los paquetes anteriormente seleccionados (Ver Figura 5.2.21) elija la opción Aplicar, a continuación se presenta una pantalla (Ver Figura 5.2.22) en donde se muestra el avance de la descarga, una vez que ha finalizado la instalación se presenta un mensaje de información (Ver Figura 5.2.23)



**Figura 5.2.21: Vista proceso de aplicar los cambios de los paquetes apache2**

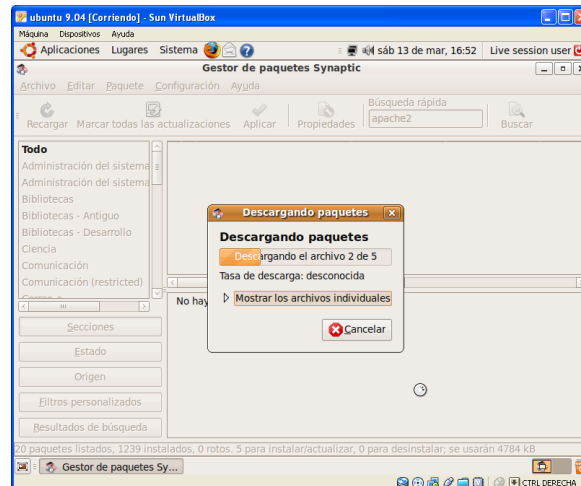


Figura 5.2.22: Vista proceso de descarga de los paquetes de apache2

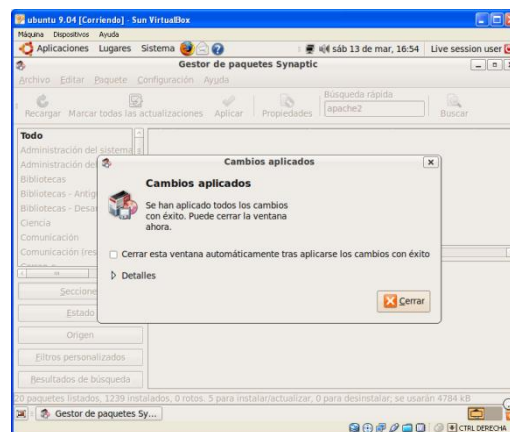
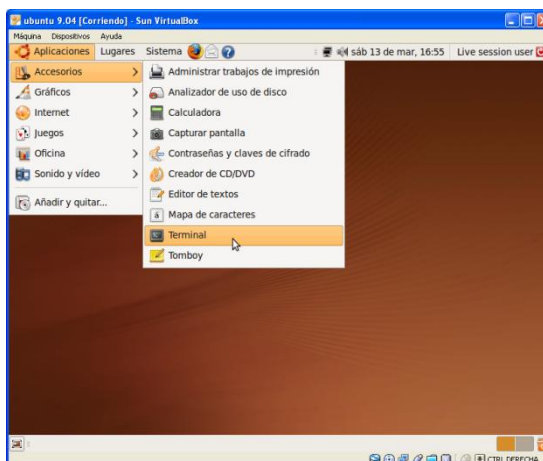


Figura 5.2.23: Vista proceso de finalizar los cambios de los paquetes apache2

#### 5.2.1.5. Configuración de los repositorios:

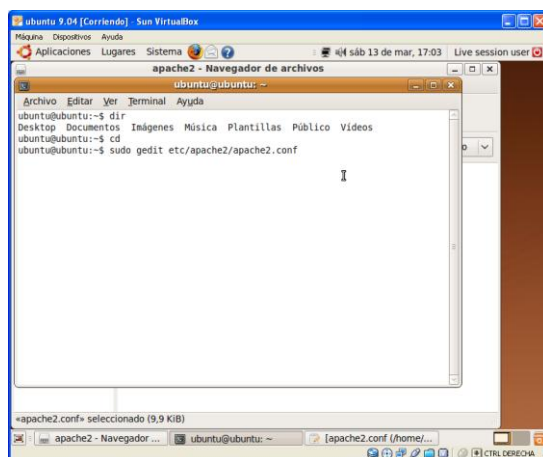
Realice la configuración del servidor de Subversion y de los repositorios padre e hijos (Ver Figura 5.2.24).

Elija Aplicaciones, Accesorios y finalmente Terminal



**Figura 5.2.24: Vista abrir el terminal de Ubuntu**

Visualice la pantalla en donde realizara la configuración (Ver Figura 5.2.25)



**Figura 5.2.25: Vista terminal de Ubuntu**

**Crear carpetas para el repositorio padre.**

```
sudo mkdir /opt/svn/energía/tesis
```

Se otorga permiso a la carpeta padre, en este caso el repositorio svn principal.

```
sudo chmod 777 -R /opt/svn/energía/tesis
```

Configuramos el archivo de **configuración de apache2**

Podemos **utilizar vim, nano, gedit o el editor que más guste.**

```
sudo gedit /etc/apache2/apache2.conf (Ver Figura 5.2.26)
```

Al final del archivo de configuración apache2.conf le agregamos las siguientes líneas. Ver figura 5.2.27.

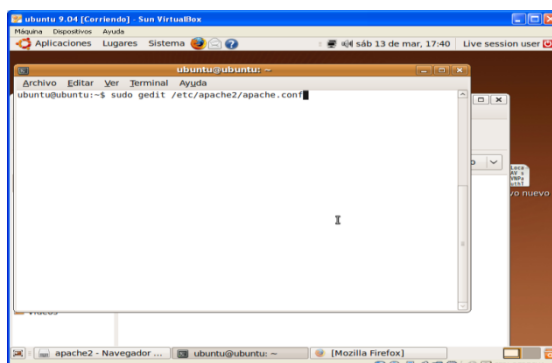


Figura 5.2.26: Vista Archivo de configuración de apache2 con Subversion

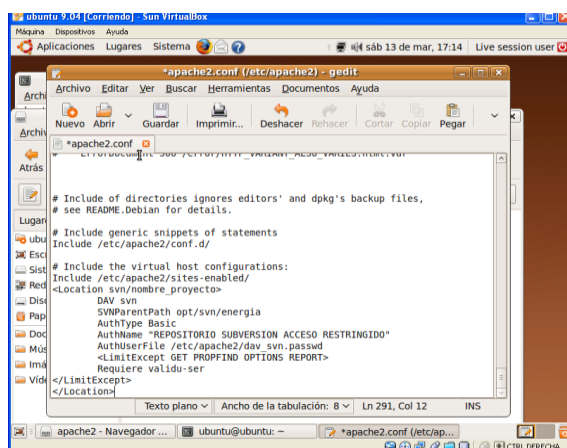


Figura 5.2.27: Vista Archivo de configuración de apache2 con Subversion

< Location	/svn/tesis>
DAV	Svn
SVNParentPath	/opt/svn/energia/tesis
AuthzSVNAccessFile	/etc/apache2/.svn_access_control
Require	valid-user
SVNIndexXSLT	/svnindex.xsl
AuthType Basic	
AuthName	"REPOSITORIO SUBVERSION UNL AEIRNNR

## ACCESO RESTRINGIDO”

AuthUserFile            /etc/apache2/dav\_svn.passwd

</Location>

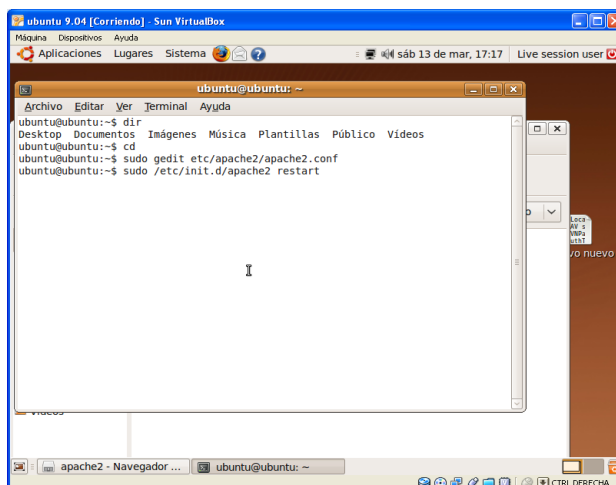
Esta definición solo permite acceso a usuarios validos (Require valid-user), si se quieren permitir usuarios anónimos o colocar otro tipo de restricciones a los accesos deben consultar la documentación de Apache.

Fijense en los 3 valores a modificar

- **SVNParentPath:** debe apuntar al directorio padre de los repositorios.
- **AuthzSVNAccessFile:** define los accesos de los usuarios a los repositorios.
- **AuthUserFile:** archivo con la definición de usuarios y los passwords.

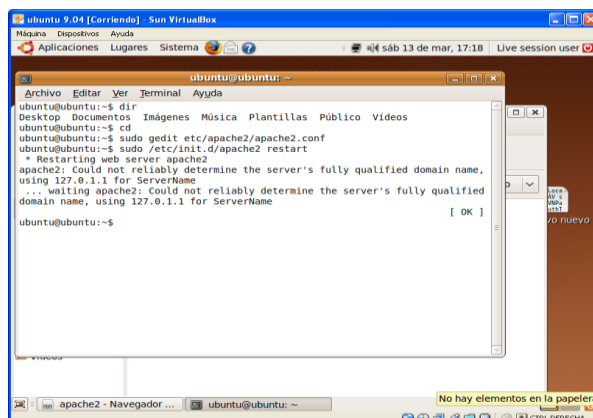
Reiniciamos el servidor de apache2

sudo /etc/init.d/apache2 restart.Ver figura 28



**Figura 5.2.28: Vista reinicio del apache2**

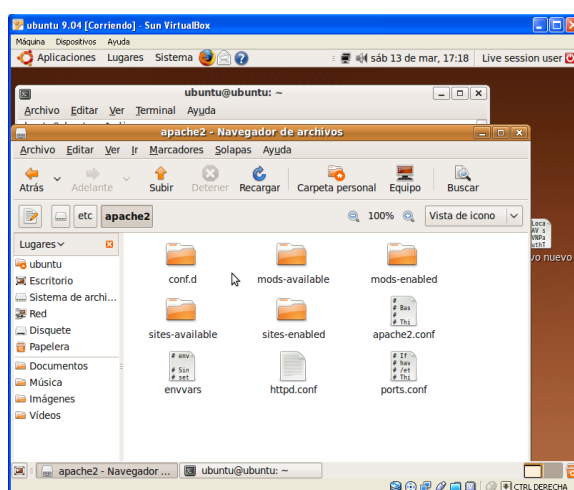
Visualice la pantalla de finalización de reinicio del apache2 (Ver Figura 5.2.29)



**Figura 5.2.29: Vista finalización del reinicio del apache2**

Acceda a los archivos del apache2:

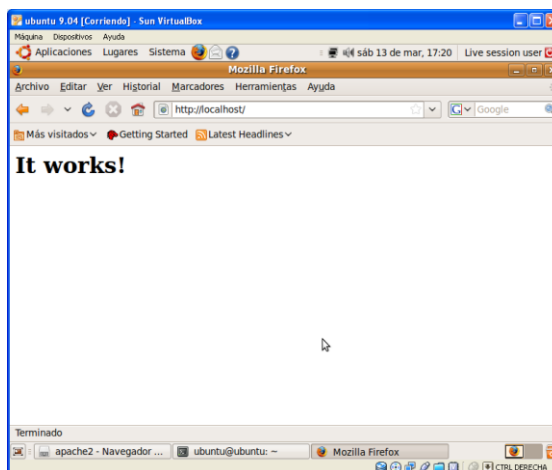
Seleccione lugares, sistema de archivos, seleccione la carpeta etc y finalmente la carpeta apache2, a continuación se pueden visualizar todos los archivos que son parte de la carpeta apache2 (Ver Figura 5.2.30)



**Figura 5.2.30: Vista archivos del servidor apache2**

Para comprobar si la configuración del apache esta correcta: escriba <http://localhost/> , de un ENTER, a continuación se presenta el mensaje It works! (Ver Figura 5.2.31) en el caso de que esté funcionando correctamente.





**Figura 5.2.31: Vista verificación del servidor apache2 mediante http://**

Si creamos otros repositorios más adelante no será necesario reiniciar Apache, sólo ésta primera vez ya que hemos cambiado uno de sus archivos de configuración

### **Habilitamos el css (OPCIONAL)**

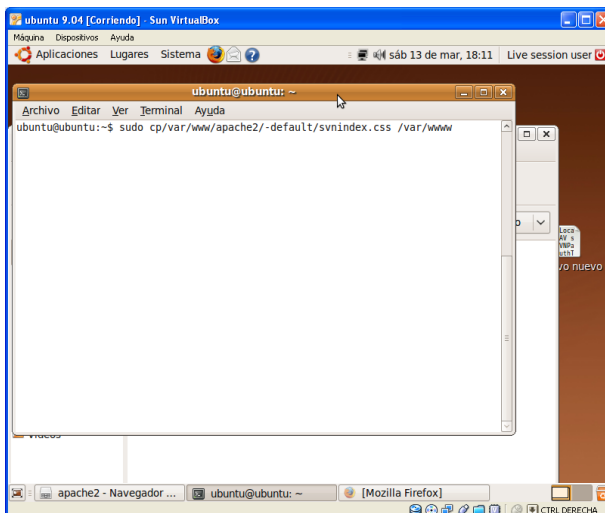
Opcionalmente podemos dar un mejor aspecto al front del repositorio, esto es mediante la habilitación del css, que tiene subversión por defecto, debemos mover los archivos de `svnindex.css` y `svnindex.xsl` a la carpeta `/var/www`

```
sudo cp /var/www/apache2-default/svnindex.css /var/www (Ver Figura 5.2.32)
```

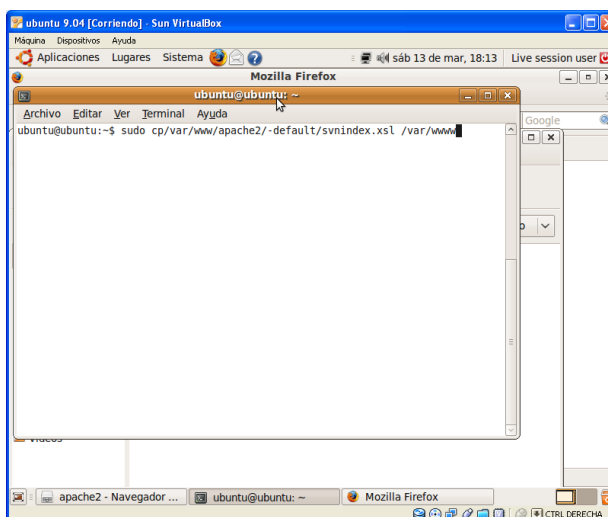
```
sudo cp /var/www/apache2-default/svnindex.xsl /var/www (Ver Figura 5.2.33)
```

Luego al archivo **dav\_svn.conf** ingresamos el siguiente párrafo después del **SvnPath**:

**SVNIndexXSLT /svnindex.xsl**



**Figura 5.2.32: Vista Habilitamos el css apache2**



**Figura 5.2.33: Vista Habilitar el css apache2**

Edite el css para el navegador, configure el archivo del apache (Ver Figura 5.2.34); repita el proceso descrito en CONFIGURACION. Capítulo 5.2.1.5

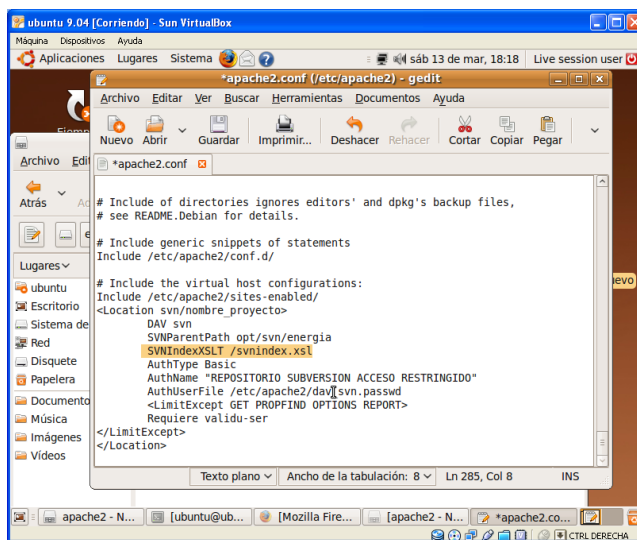


Figura 5.2.34: Vista habilitar css en el servidor

Crear nuestro repositorio svn

- `sudo svnadmin create /opt/svn/energía/tesis/mi_repositorio`
- `sudo chmod 777 -R /opt/svn/energía/tesis/mi_repositorio`
- `sudo chown -R www-data /opt/svn/energía/tesis/mi_repositorio`

Con Apache 2 instalado tiene más sentido que el propietario de tus repositorios sea el usuario www-data ver Figuras 5.2.35, 5.2.36

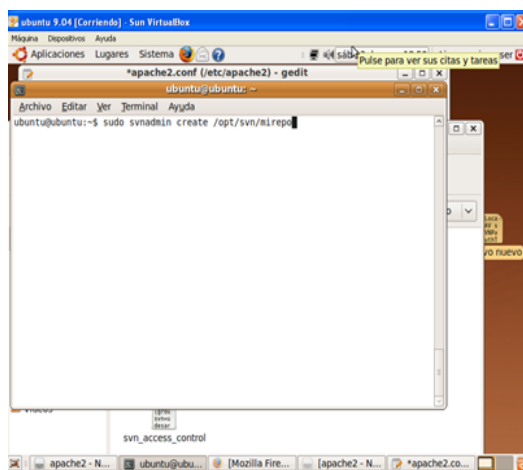
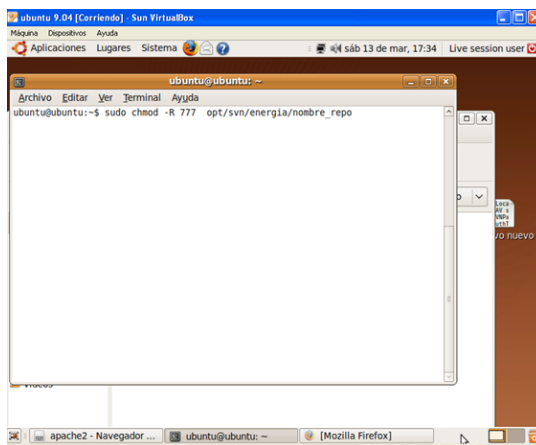


Figura 5.2.35: Vista Crear repositorio Subversion



**Figura 5.2.36: Vista habilitar permiso del repositorios**

Asignar propiedad y permisos al repositorio y sus subarchivos

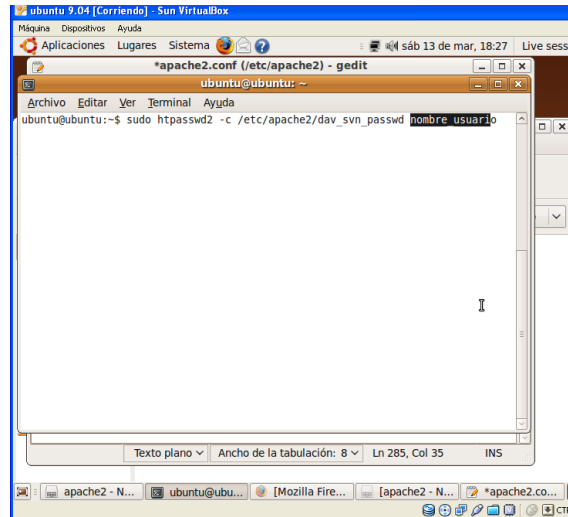
- a. `sudo htpasswd2-c /etc/apache2/dav_svn.passwd us1`
- b. `sudo htpasswd2 /etc/apache2/dav_svn.passwd us2`

Para el primer usuario que autoricemos debemos especificar el modificador `-c` para que el archivo almacén de contraseñas se cree por primera vez. Después podremos añadir tantos usuarios como queramos sin especificar `-c`, tal y como muestra la segunda línea.

- new password:
- re-type new password:
- updating password for user user1

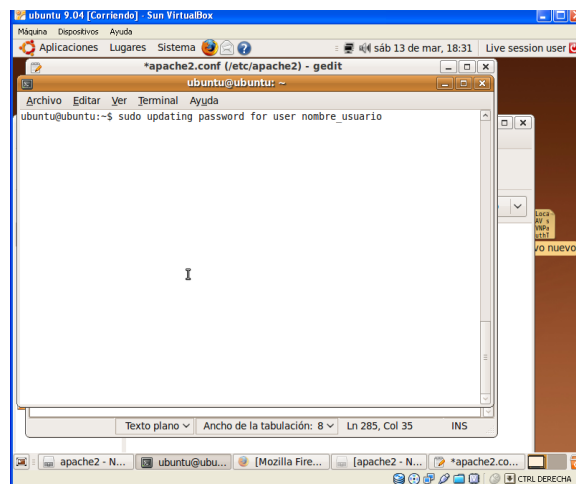
*Si se quiere limpiar el archivo y definir usuarios desde cero se debe ejecutar el comando descrito en el literal a, (se recomienda hacer esto la primera vez)*

Cree el usuario y la clave con la cual puede acceder el usuario al servidor.



**Figura 5.237: Crear usuario y clave de acceso**

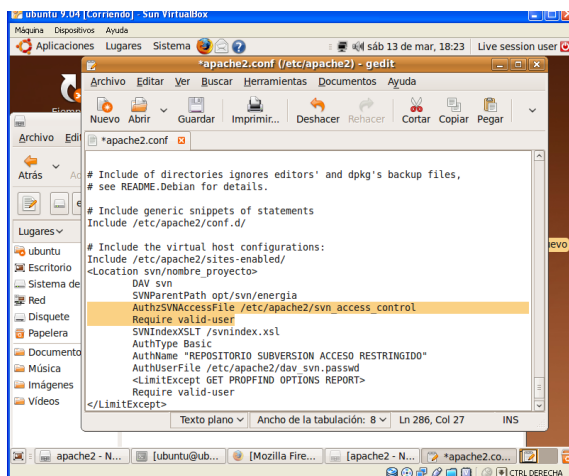
Actualice el usuario de Subversion (Ver Figura 5.238).



**Figura 5.238: Vista actualizar Usuario de svn**

Asignamos permiso por grupo a los repositorios

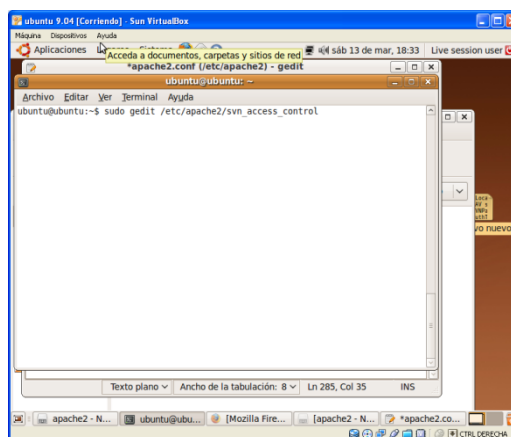
La definición de los accesos se hace en el archivo definido por la entrada **AuthzSVNAccessFile**, en nuestro caso es **/etc/apache2/svn\_access\_control**. En este archivo se definen grupos de accesos y permisos de acceso a cada repositorio.



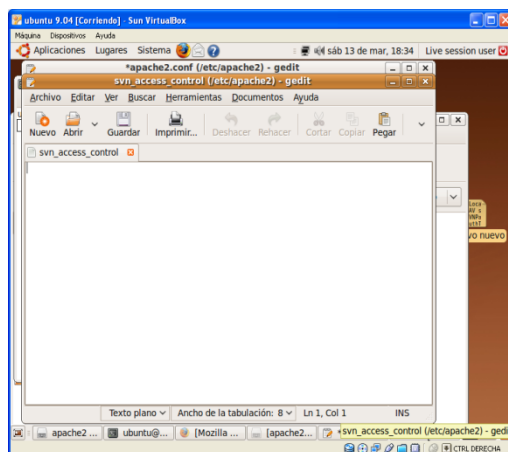
**Figura 5.2.39: Vista Actualizar Archivo de configuración para acceso de control**

Crear el archivo de accesos por grupo.

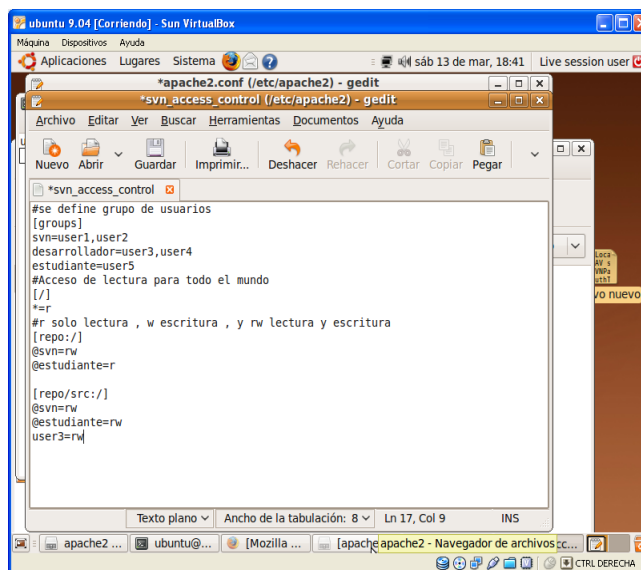
```
sudo vim /etc/apache2/svn_access_control
```



**Figura 5.2.40: Vista Editar Archivo svn\_access\_control**



**Figura 5.2.41: Vista Administrar Archivo acceso\_control**



**Figura 5.2.42: Vista Archivo acceso\_control**

```
# Se definen los grupos de trabajo
[groups]
svn=user1, user2
guest=user3, user4
deveryone=@devteam, @guest
# Acceso de lectura para todo el mundo
[/]
* = r
# guest pueden leer pero solo devteam actualizar en
# el repositorio repo
[repo:/]
@devteam=rw
@guest=r
# Todo el mundo puede leer el repo test pero solo
# user5 lo puede modificar
[test:/]
user5=rw
@everyone=r
# Solo el grupo devteam tiene acceso total al
# repositorio bcol. Guest no tienen ningun acceso
[bcol:/]
```

@devteam

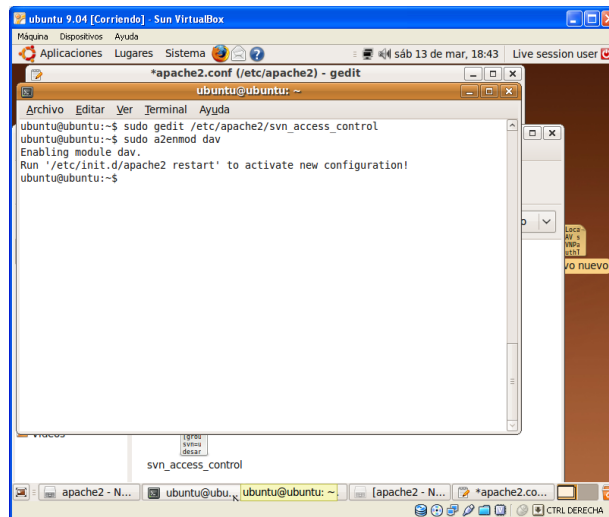
=

rw

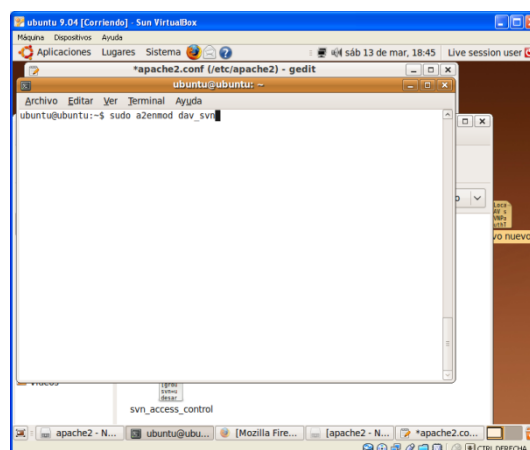
@guest =

Para que Apache tome los cambios se deben habilitar los módulos de svn y reiniciar el servicio.

- `sudo a2enmod dav`
- `sudo a2enmod dav_svn`
- `sudo /etc/init.d/apache2 force-reload`

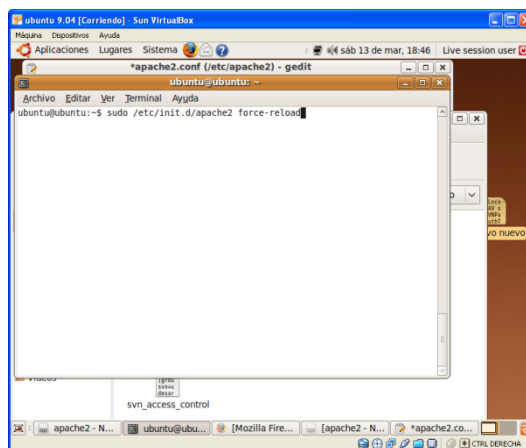


**Figura 5.2.43: Vista reiniciar El servidor apache2**



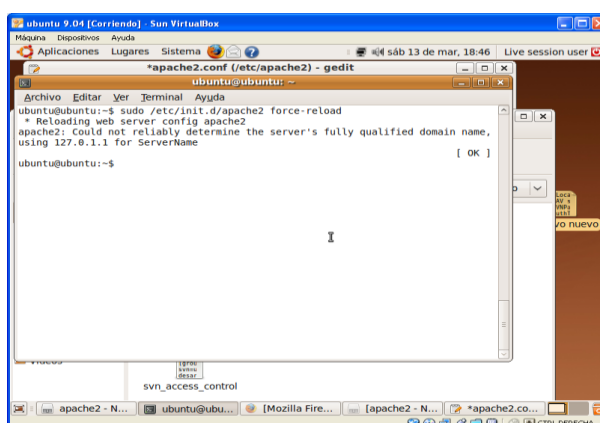
**Figura 5.2.44: Vista reiniciar El servidor a2enmod dav**





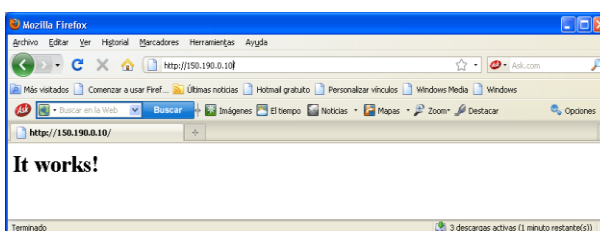
**Figura 5.2.45: Vista reiniciar El servidor apache2 force-reload**

Visualice el reinicio de los servicios del apache2 (Ver Figura 5.2.46)



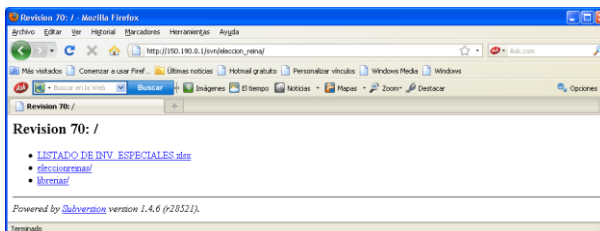
**Figura 5.2.46: Vista finalizando el reinicio de los servicios apache2**

Si está instalado el servidor apache2 ingresamos a un explorador e ingresamos a la dirección del servidor ver figura 5.2.47, a continuación se presenta el mensaje It Works! si el servidor está funcionando correctamente.

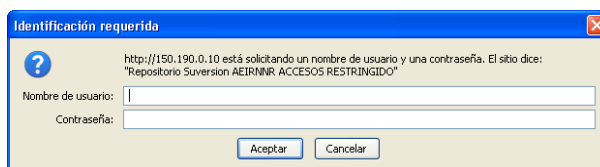


**Figura 5.2.47: Vista inicio del servidor de apache2**

Para ingresar a un repositorio en específico, ingrese a un explorador, escriba la dirección del servidor en el cual se encuentra el proyecto y el nombre del repositorio (Ver Figura 5.2.48), luego puede visualizar las carpetas y archivos con los que cuenta el mismo para ello se ingrese el usuario y la contraseña (Ver Figura 5.2.49)

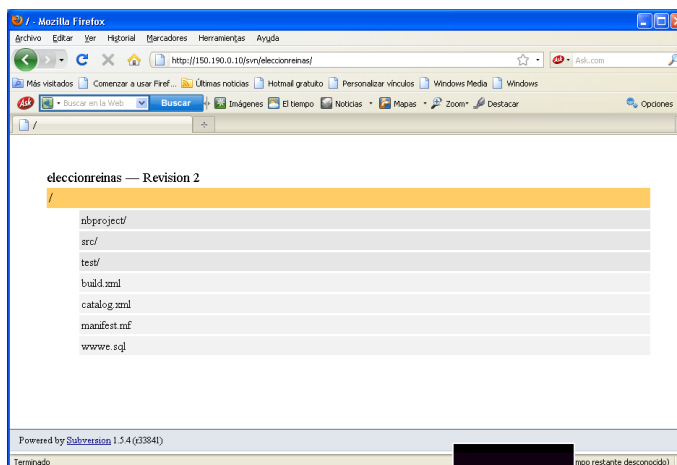


**Figura 5.2.48: Vista de un repositorio Subversion desde el servidor de apache2**



**Figura 5.2.49: Vista de un repositorio Subversion desde el servidor de apache2 mediante acceso por usuario. (Petición de la clave de acceso)**

Para ingresar a un repositorio en específico que está utilizando css, ingrese a un explorador, escriba la dirección del servidor en el cual se encuentra el proyecto y el nombre del repositorio, luego puede visualizar las carpetas y archivos con los que cuenta el mismo (Ver Figura 5.2.50).



**Figura 5.2.50: Vista de un repositorio Subversion accediendo al proyecto con CSS.**

#### 5.2.1.6. Comandos en Subversion

- **svn:** es el programa cliente, en líneas de comandos, es decir que se puede agregar, importar y actualizar archivos al repositorio central.
- **svnadmin:** herramienta para crear, modificar o reparar un repositorio de Subversion.

#### 5.2.1.7. Comandos para SVN:

- **checkout** Este genera una referencia o copia del repositorio central hacia un repositorio local.

**svn checkout [URL] [PATH]** Si se omite el **path**, se toma el **path** base de la URL que se está utilizando como destino. Nos encontramos en el directorio **“/home/diego/pruebas”** y el repositorio central está en **“/var/svn-repos/practica1/”**, entonces el comando seria: **svn comando file:///var/svn-repos/practica1/**

**svn checkout** Importa una copia de trabajo que tengamos en el directorio.

*Otros ejemplos:*

- **svn checkout file:///tmp/repos/test file:///tmp/repos/quiz svn checkout http://host\_name/svn\_dir/repository\_name/project/trunk proyecto**
- **copy:** Copia un archivo en una copia de trabajo o en el repositorio.
  - **svn copy FUENTE DESTINO**



Ejemplo:

- `svn copy foo.txt bar.txt`

- **update:** Actualiza los cambios que hay en el repositorio hacia nuestra copia de trabajo actual.

- `svn update [PATH...]`

Ejemplo:

- `svn update`

- **import:** Realiza un commit recursivo de lo que hay en path hacia URL. Este comando sirve para añadir archivos al repositorio. Por lo general cuando se crea un proyecto, la primera vez se hace un import para copiar todo el proyecto al servidor (repositorio local o remoto).

- **`svn import path_al_directorio nombre_repositorio`**

Ejemplo:

`svn import /home/mario/pruebas/proyecto1 /var/svn-repos/practica1`

`svn import proyecto`

`http://host_name/svn_dir/repository_name/project -m "inicio"`

- **add:** Añadir archivos, directorios o enlaces simbólicos al repositorio central.
  - `svn add PATH`
  - Ejemplo:
  - `svn add archivo.txt`

Este se agregara hasta darle un COMMIT.

- **commit:** Envía los cambios realizados en nuestra copia de trabajo hacia el repositorio.

- `svn commit [PATH...]`

Luego de añadir los archivos o proyectos con ADD se tiene que dar commit para que realice los cambios en el repositorio central. Ejemplo:

`svn add archivo.txt`

**svn commit** Luego aparece una especie de LOG, le agregan un comentario al archivo commit que se crea (LOG) para que los demás usuarios sepa que se hizo con ese commit. Y para salir un CTRL- X. Para crear el log automáticamente se pone el parámetro "m" y luego el mensaje.

**`svn commit -m "mensaje" [PATH..]`**

`svn commit -m "haciendo cambios"`  
`http://localhost/svn_dir/repository/project_dir`



- **delete:** Igual que el ADD solo que este elimina archivos o directorios.
  - `svn delete hellworld.txt`
  - **svn commit -m "elimino el fichero helloworld"**
- **svn info.-** Información sobre la última revisión:
- **svn status --show-updates --verbose**
- **svn checkout --revision [número de revisión]**
- **svn update --revision [número de revisión]**

#### 5.2.1.8. Comandos para SVNADMIN:

- **create:** Crea un directorio en la ruta especificada.  
`svnadmin create /var/svn-repos/practica1`
- **recover:** Este comando debe ser ejecutado cuando obtenemos un error de que nuestro repositorio debe ser recuperado.  
`svnadmin recover REPOS_PATH`  
Ejemplo: `$ svnadmin recover /usr/local/svn/repos`
- **dump:** Envía el contenido de un repositorio del sistema hacia un archivo "dumpfile".
- **svnadmin dump REPOS\_PATH [-r LOWER[:UPPER]] [--incremental].**Ejemplo:

### 5.3 Configuración del repositorio subversion cliente

Este documento describe el uso diario del cliente TortoiseSVN *no* es una introducción a los sistemas de control de versiones, y *no* es una introducción a Subversion (SVN). Es más como un lugar donde puede venir cuando sepa qué quiere hacer, pero no recuerde exactamente cómo hacerlo.

## INSTALACIÓN

Para aprovechar al máximo el manual del uso diario:

- Debe tener ya instalado TortoiseSVN.



- Debe estar familiarizado con los sistemas de control de versiones.
- Debe conocer las bases de Subversion.
- Debe haber preparado un servidor y/o tener acceso a un repositorio de Subversion.

### 5.3.1. REQUERIMIENTOS DEL SOFTWARE Y HARDWARE

#### Requerimientos del hardware para la instalación SUBVERSION CLIENTE

Como requerimientos del hardware tenemos:

Requisito	Profesional
Sistema Operativo	Windows milenio, 2000 SP4 o superior Windows XP Service Pack 1 o superior Para un equipo de 64 bits, los requisitos son: <ul style="list-style-type: none"><li>• Ediciones de x64 de Windows Server 2003 Service Pack 1</li><li>• Edición de x64 para Windows XP Professional</li><li>• Linux Ubuntu 9.0.4 o superior, fedora, 10 o superior</li></ul>
Procesador	PC con un procesador de tipo Pentium IV a 600 MHz Se recomienda: 1 gigahercio (GHz)
RAM	512 MB RAM, se recomienda 1 GB o más
Vídeo	800 X 600, 256 colores Se recomienda: 1024 X 768, color de alta densidad de 16 bits
Disco Duro	1,5 GB NTFS o 3 GB FAT o más de espacio disponible dependiendo de los proyectos

#### Requerimientos del software para la instalación SUBVERSION CLIENTE

Como requerimientos del Software tenemos:

- Un sistema operativo Windows de 32 bits, tal como 9x, Me, NT, 2000, XP, o Windows Server 2003, Linux Ubuntu 9.0.4 o superior, fedora 10 en adelante
- Soporte para protocolo TCP/IP.



- Suficiente espacio en disco duro para descomprimir, instalar, y crear las copias de los repositorios clientes del servidor. Generalmente se recomienda un mínimo de 200 megabytes dependiendo del proyecto.
- Cualquier lenguaje de programación que soporte el sistema operativo.

### 5.3.1.1 Guía de Instalación

#### Requerimientos del sistema

TortoiseSVN se ejecuta en Windows 2000 SP2, Windows XP o superiores. Windows 98, Windows ME y Windows NT4 ya no se soportan desde TortoiseSVN 1.2.0, pero aún puede descargar las versiones más antiguas si realmente las necesita.

#### Instalación

TortoiseSVN viene con un instalador fácil de utilizar. Haga doble click en el fichero de instalación y siga las instrucciones. El instalador se encargará del resto

- Se procede con la instalación del cliente del servidor tortoiseSvn*
- Seleccionar La carpeta de instalación donde se encuentra el instalador de TortoiseSVN-1.6.5.16974-win32-svn-1.6.5.ms*
- Hacer clic en el icono Setup "TortoiseSVN-1.6.5.16974-win32-svn-1.6.5.ms" para la instalación del cliente (figura 5.3.1.)*

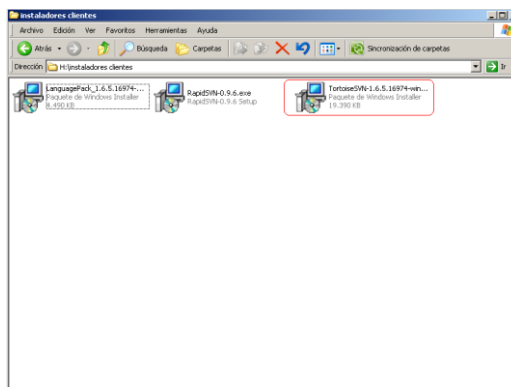


Figura 5.3.1: Instalador del Cliente TortoiseSVN

- c. Al hacer clic en Setup, se inicia la instalación
- d. al Hacer clic en el botón “Next”. (Figura 5.3.2).

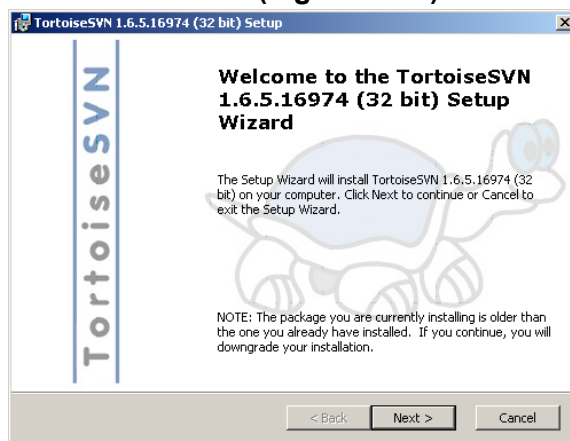


Figura 5.3.2: Instalación de TortoiseSVN

- e. En la siguiente vista aceptamos los términos de la licencia ver figura 5.3.3 y 5.3.4

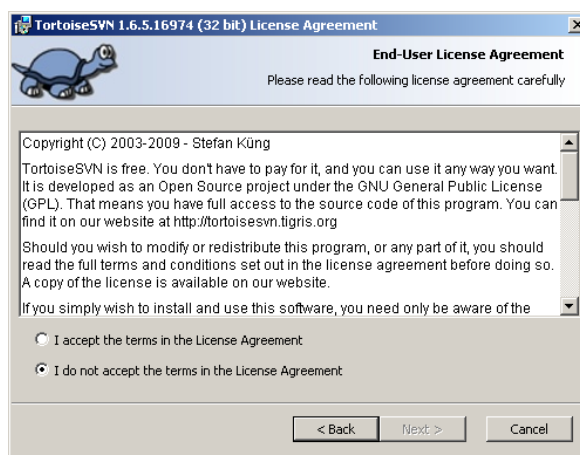


Figura 5.3.3: Instalación de Subversion acepta Licencia

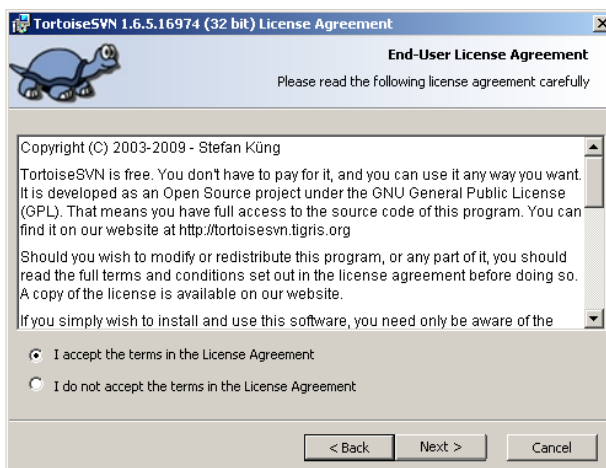


Figura 5.3.4: Instalación de SubversionSVN aceptar término



f. Hacer click en el botón next ver Figura 5.3.5

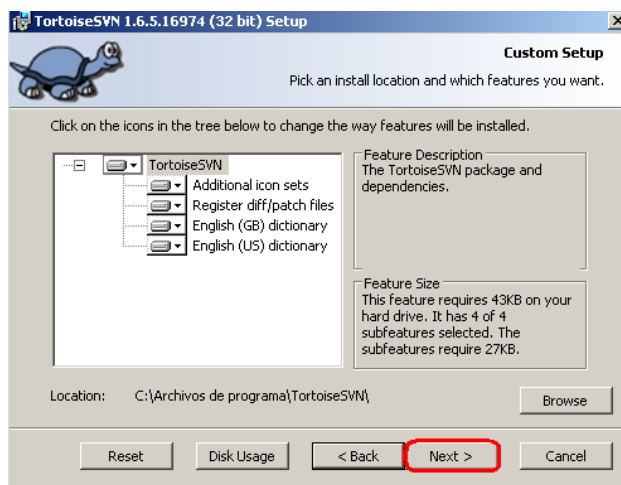


Figura 5.3.5: Instalación de SubversionSVN paquetes

d. Escogemos la opción Install ver figura 5.3.6

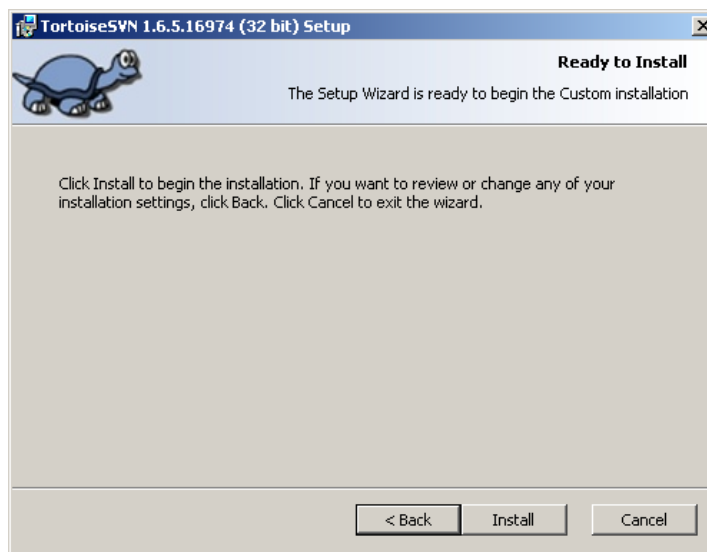


Figura 5.3.6: Instalación de SubversionSVN inicio instalación

- e. Al término de la instalación escogemos la opción Finish ver Figura 5.3.7

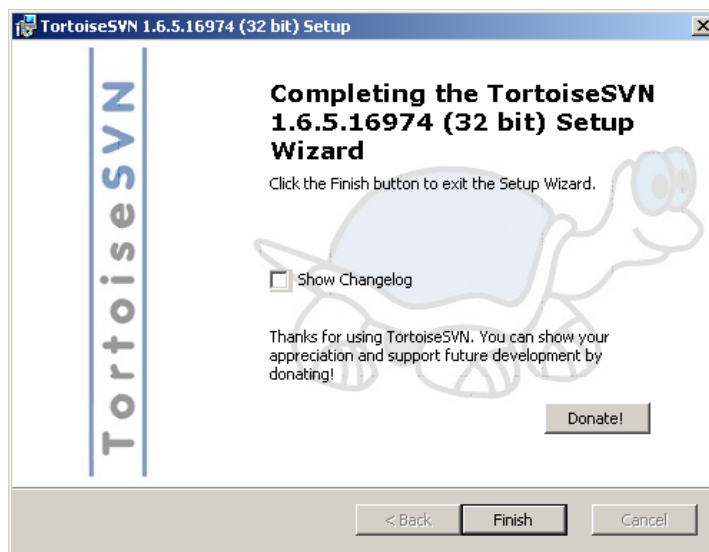


Figura 5.3.7: Finalizando la Instalación de SubversionSVN

- f. Reiniciamos el equipo
- g. Instalación del lenguaje español (Opcional)
- h. Escogemos el icono de instalación languagePack\_1.6.5.16974-win32-es.msi ver figura 5.3.8.

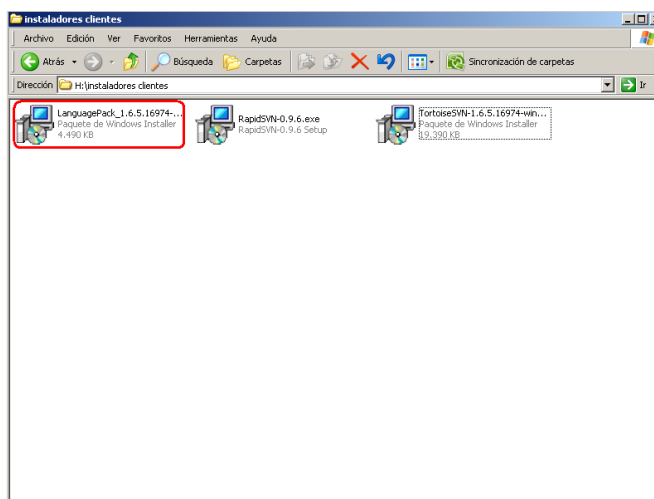


Figura 5.3.8: Paquete de idioma español SubversionSVN

- i. Hacer click en el icono LanguagePack\_1.6.5.16974-win32-es.msi el sistema mostrara una ventana, escogemos la opción next ver figura 5.3.9

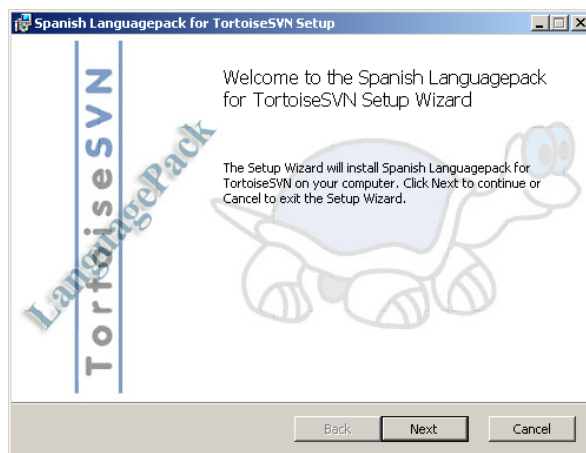


Figura 5.3.9: Instalación del lenguaje Español

- j. Se instalará el lenguaje español a subversionSVN , escogemos la opción Finish ver figura 5.3.9.



Figura 5.3.10: Terminación de la instalación del lenguaje Español

- k. Reiniciamos el equipo.

normal	eliminado.cpp	no-versionado.h	sólo-lectura.java	ignorado.doc	en conflicto.php
modificado	bloqueado.cpp	normal.txt	eliminado.java	no-versionado.doc	sólo-lectura.php
en conflicto	añadido.cpp	modificado.txt	bloqueado.java	normal.pl	eliminado.php
sólo-lectura	ignorado.cpp	en conflicto.txt	añadido.java	modificado.pl	bloqueado.php
eliminado	no-versionado.cpp	sólo-lectura.txt	ignorado.java	en conflicto.pl	añadido.php
bloqueado	normal.h	eliminado.txt	no-versionado.java	sólo-lectura.pl	ignorado.php
añadido	modificado.h	bloqueado.txt	normal.doc	eliminado.pl	no-versionado.php
ignorado	en conflicto.h	añadido.txt	modificado.doc	bloqueado.pl	normal.asp
no-versionado	sólo-lectura.h	ignorado.txt	en conflicto.doc	añadido.pl	modificado.asp
normal.cpp	eliminado.h	no-versionado.txt	sólo-lectura.doc	ignorado.pl	en conflicto.asp
modificado.cpp	bloqueado.h	normal.java	eliminado.doc	no-versionado.pl	sólo-lectura.asp
en conflicto.cpp	añadido.h	modificado.java	bloqueado.doc	normal.php	eliminado.asp
añadido.asp	modificado.vb	bloqueado.xml	normal.dpr	eliminado.dfm	no-versionado.res
ignorado.asp	en conflicto.vb	añadido.xml	modificado.dpr	bloqueado.dfm	normal.asm
no-versionado.asp	sólo-lectura.vb	ignorado.xml	en conflicto.dpr	añadido.dfm	modificado.asm
normal.cs	eliminado.vb	no-versionado.xml	sólo-lectura.dpr	ignorado.dfm	en conflicto.asm
modificado.cs	bloqueado.vb	normal.pas	eliminado.dpr	no-versionado.dfm	sólo-lectura.asm
en conflicto.cs	añadido.vb	modificado.pas	bloqueado.dpr	normal.res	eliminado.asm
sólo-lectura.cs	ignorado.vb	en conflicto.pas	añadido.dpr	modificado.res	bloqueado.asm
eliminado.cs	no-versionado.vb	sólo-lectura.pas	ignorado.dpr	en conflicto.res	añadido.asm
bloqueado.cs	normal.xml	eliminado.pas	no-versionado.dpr	sólo-lectura.res	ignorado.asm
añadido.cs	modificado.xml	bloqueado.pas	normal.dfm	eliminado.res	no-versionado.asm
ignorado.cs	en conflicto.xml	añadido.pas	modificado.dfm	bloqueado.res	normal.aspx
no-versionado.cs	sólo-lectura.xml	ignorado.pas	en conflicto.dfm	añadido.res	modificado.aspx
normal.vb	eliminado.xml	no-versionado.pas	sólo-lectura.dfm	ignorado.res	en conflicto.aspx

Una de las funciones más visibles de TortoiseSVN son los iconos sobrepuestos que aparecen en los ficheros de su copia de trabajo. Estos le muestran de un vistazo qué ficheros han sido modificados

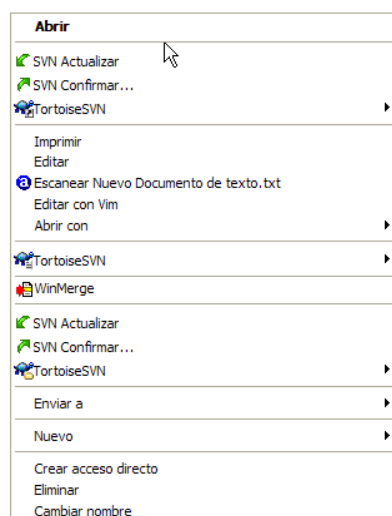
The screenshot shows the TortoiseSVN context menu. The menu is open, displaying a list of actions. The 'TortoiseSVN' menu item is highlighted in blue. The actions listed are:

- Mostrar registro
- Navegador de repositorios
- Comprobar modificaciones
- Gráfico de revisiones
- Resuelto...
- Actualizar a la revisión...
- Renombrar...
- Eliminar...
- Revertir...
- Limpiar
- Obtener bloqueos...
- Quitar Bloqueo
- Rama/etiqueta...
- Cambiar...
- Fusionar...
- Exportar...
- Relocalizar...
- Añadir...
- Crear parche...
- Aplicar parche...
- Propiedades
- Ayuda
- Configuración
- Acerca de

**Figura 5.3.12: Menú contextual para un directorio bajo el control de versiones**

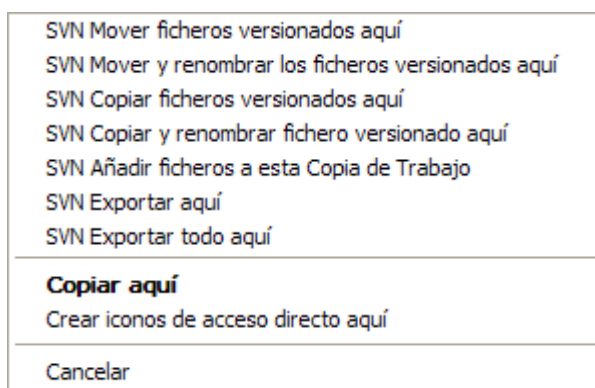
Todos los comandos de TortoiseSVN se invocan desde el menú contextual del explorador de Windows. La mayoría se ven directamente, cuando hace click con el botón derecho en un fichero o una carpeta. Los comandos disponibles dependen de si el fichero o la carpeta o su carpeta padre está bajo el control de versiones o no. También puede ver el menú de TortoiseSVN como parte del menú archivo del explorador.

En algunos casos puede ver varias entradas de TortoiseSVN. ¡Esto no es un error!



**Figura 5.3.13: Menú archivo del explorador para un acceso directo en una carpeta versionada**

#### 5.3.1.1.3. Arrastrar y soltar



**Figura 5.3.14: Menú de arrastre con el botón derecho para un directorio bajo el control de versiones**

Otros comandos están disponibles como manejadores de arrastre, cuando arrastra con el botón derecho ficheros o carpetas a un nuevo destino dentro de copias de trabajo, o cuando arrastra con el botón derecho un fichero o una carpeta no versionados a un directorio que está bajo el control de versiones.

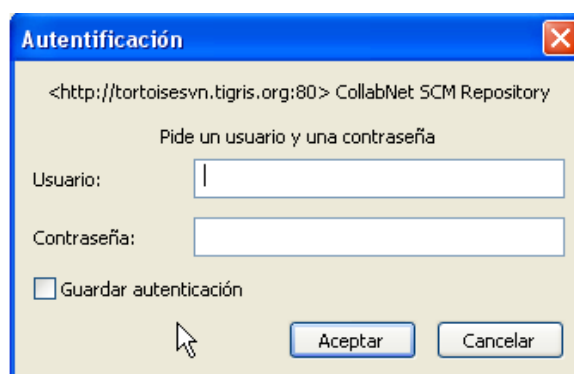
#### 5.3.1.1.4. Atajos comunes

Algunas operaciones comunes tienen atajos de Windows bien conocidos, pero no aparecen en botones o en los menús. Si no puede averiguar cómo hacer algo obvio, como refrescar una vista.

- F1 La ayuda, por supuesto.
- F5 Refresca la vista actual. Este es quizás el comando de una tecla más útil. Por ejemplo... en el Explorador esto refresca los iconos sobreimpresionados en su copia de trabajo. En el diálogo de confirmación volverá a re escanear la copia de trabajo para ver qué puede necesitar ser confirmado. En el diálogo de Mostrar Registro contactará con el repositorio de nuevo buscando los cambios más recientes.
- Ctrl-A Selecciona todo. Esto puede ser útil si obtiene un mensaje de error y quiere copiar y pegarlo en un email. Utilice Ctrl-A para seleccionar el mensaje de error y luego Ctrl-C copia el texto seleccionado.

#### 5.3.1.1.5. Autenticación

Si el repositorio al que intenta acceder está protegido por contraseña, aparecerá un diálogo de autenticación.



**Figura 5.3.15: Diálogo de autenticación**



El cliente `svn.simple` contiene las credenciales para la autenticación básica (usuario/contraseña).

- **svn.ssl.server** contiene los certificados SSL de servidor.
- **svn.username** contiene las credenciales para autenticación sólo por usuario (sin necesidad de contraseña).

#### 5.3.1.1.6. Maximizando ventanas

Muchos de los diálogos de TortoiseSVN tienen montones de información que mostrar, pero a menudo es más útil maximizar sólo la altura o sólo la anchura, mejor que maximizar para ocupar toda la pantalla. Como ayuda existen atajos para esto en el botón Maximizar. Utilice el botón central del ratón para maximizar verticalmente, y el botón derecho del ratón para maximizar horizontalmente.

#### 5.3.1.2. Importando datos en un repositorio

##### 5.3.1.2.1. Importar

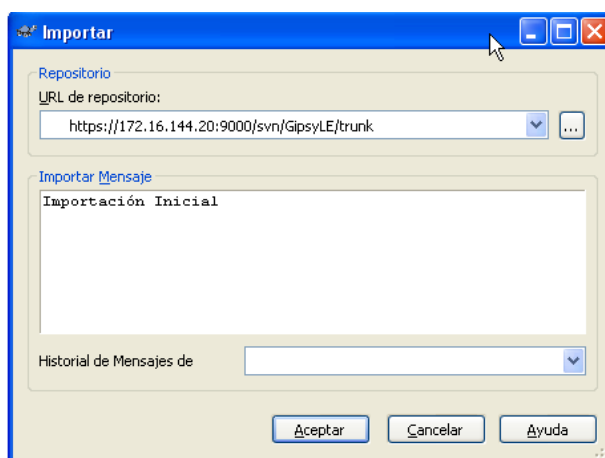
Si está importando en un repositorio que ya tiene algunos proyectos, entonces la estructura del repositorio ya estará decidida. Si está importando datos a un nuevo repositorio entonces merece la pena tomar el tiempo para pensar en cómo debería organizarse.

Antes de importar su proyecto en un repositorio debería:

1. Quitar todos los ficheros que no se necesitan para construir el proyecto (ficheros temporales, ficheros que se generan por un compilador como los \*.obj, binarios compilados, ...)
2. Organizar los ficheros en carpetas y subcarpetas. Aunque es posible renombrar/mover los ficheros más tarde, ¡es muy recomendable que tenga la estructura del proyecto antes de importarlo!

Ahora seleccione la carpeta superior de la estructura de directorios del proyecto en el explorador de Windows, y haga click con el botón derecho para abrir el menú

contextual. Seleccione el comando TortoiseSVN → Importar... y aparecerá un cuadro de diálogo:



**Figura 5.3.17: El diálogo Importar**

En este diálogo tiene que introducir la URL del lugar del repositorio donde desea importar su proyecto. Es muy importante darse cuenta de que la carpeta local que está importando no aparece en sí misma en el repositorio, sólo su contenido.

#### **5.3.1.2.2. Importar en el sitio**

Asumiendo que ya tiene un repositorio, y que quiere añadir una nueva estructura de carpetas e él, sólo tiene que seguir estos pasos:

1. Utilice el navegador de repositorios para crear nuevas carpetas de proyecto directamente en el repositorio.
2. Ejecute la operación obtener de la nueva carpeta sobre la carpeta de más alto nivel que desea importar. Obtendrá una advertencia porque la carpeta local no está vacía. Ahora tiene una carpeta de más alto nivel versionada con contenido no versionado.
3. Utilice TortoiseSVN → Añadir... en esta carpeta versionada para añadir parte o todo su contenido. Puede añadir y eliminar ficheros, establecer las propiedades svn: ignore en las carpetas y hacer cualquier otro cambio que necesite.
4. Confirme la carpeta de más alto nivel, y ya tiene un nuevo árbol versionado, y una copia de trabajo local, creada desde su carpeta existente.



### 5.3.1.2.3. Ficheros especiales

A veces necesitará tener un fichero bajo control de versiones que contenga datos específicos del usuario. Esto significa que tiene un fichero que cada desarrollador/usuario necesita modificar para que se ajuste a su configuración local. Pero versionar ese fichero es difícil, porque cada usuario haría confirmaciones de sus cambios cada vez en el repositorio.

En estos casos le sugerimos que utilice ficheros plantilla. Cree un fichero que contenga todos los datos que sus desarrolladores puedan necesitar, añádalo al control de versiones y haga que sus desarrolladores lo obtengan. Luego, cada desarrollador tendrá que *hacer una copia* de ese fichero y renombrar esa copia. Después de eso, modificar la copia no vuelve a ser un problema.

### 5.3.1.3 Obteniendo una copia de trabajo

Para tener una copia de trabajo necesita *obtener* una de un repositorio.

Seleccione un directorio en el explorador de Windows donde quiera poner su copia de trabajo. Haga click con el botón derecho para mostrar el menú contextual y seleccione el comando TortoiseSVN → Obtener..., que mostrará el siguiente cuadro de diálogo:

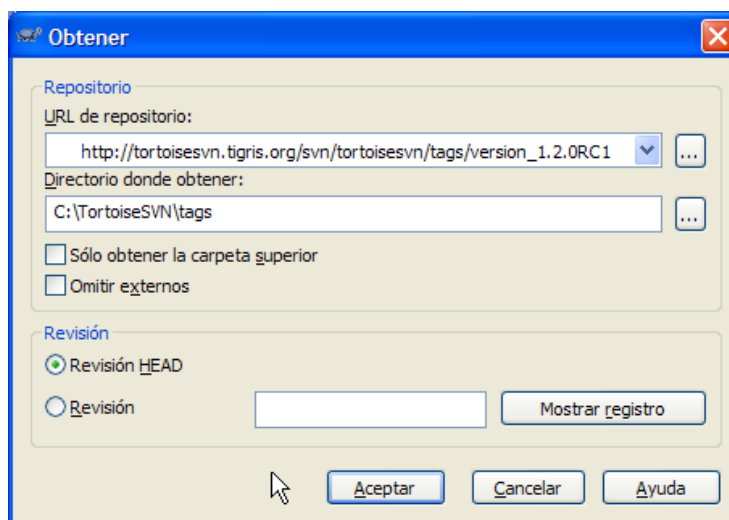


Figura 5.3.17: El diálogo Obtener

Si introduce un nombre de carpeta que no aún no exista, se creará un directorio con ese nombre.



#### 5.3.1.3.1. Profundidad de obtención

Puede elegir la *profundidad* que desea para la obtención, lo que le permite especificar la profundidad de la recursión en las carpetas hijas. Si sólo desea unas pocas secciones de un árbol grande, puede obtener sólo la carpeta de más alto nivel, y luego actualizar las carpetas seleccionadas de forma recursiva.

*Totalmente recursivo*

Obtener el árbol entero, incluyendo todas las carpetas hijas y subcarpetas.

*Hijos inmediatos, incluyendo carpetas*

Obtener el directorio especificado, incluyendo todo el fichero syn carpetas hijas, pero no rellenar las carpetas hijas.

*Sólo los ficheros hijos*

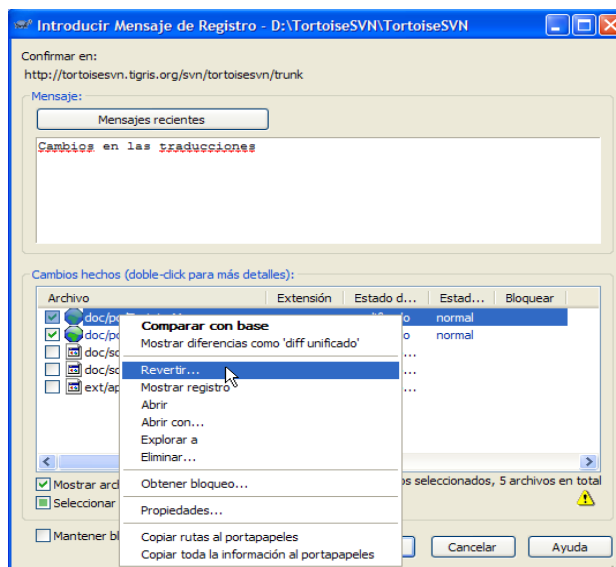
Obtener la carpeta especificada, incluyendo todos los ficheros pero no obtener ninguna carpeta hija.

#### 5.3.1.4 Confirmando sus cambios en el repositorio

Enviar los cambios que ha hecho al repositorio se conoce como *confirmar* los cambios. Pero antes de confirmar tiene que estar seguro de que su copia de trabajo está actualizada. Puede o bien ejecutar TortoiseSVN → Actualizar directamente, o bien ejecutar TortoiseSVN → Comprobar Modificaciones primero, para ver qué se ha cambiado localmente o en el servidor.

##### 5.3.1.4.1. El diálogo de Confirmación

Si su copia de trabajo está actualizada y no hay conflictos, ya está preparado para confirmar sus cambios. Seleccione los ficheros y/o carpetas que desee confirmar y seleccione TortoiseSVN → Confirmar....



**Figura 5.3.18: El diálogo de Confirmación**

El diálogo de confirmación le mostrará todos los ficheros cambiados, incluso los ficheros añadidos, borrados o no versionados. Si no desea que un fichero cambiado se confirme, simplemente desmarque ese fichero. Si desea incluir un fichero no versionado, márkelo para añadirlo a la confirmación.

Los ítems que han sido cambiados a una ruta de repositorio diferente también se indican utilizando un marcador (s). Puede haber cambiado algo mientras trabaja en una rama y

#### **5.3.1.4.2. Listas de cambios**

El diálogo de confirmación da soporte a las listas de cambios de Subversion para ayudar a agrupar ficheros relacionados.

#### **5.3.1.4.3. Excluyendo ítems de la lista de confirmación**

A veces tiene ficheros versionados que cambian con frecuencia pero que realmente no desea confirmar. En ocasiones esto indica un fallo en su sistema de compilación - ¿por qué están esos ficheros versionados? ¿Debería utilizar ficheros de plantilla? Pero ocasionalmente es inevitable. Una razón clásica es que su IDE cambie

una fecha en el fichero de proyecto cada vez que lo compile. El fichero de proyecto debe estar versionado ya que contiene todas las configuraciones de la compilación, pero no necesita confirmarse sólo porque la fecha haya cambiado.

Para ayudarle en casos tan extraños como estos, hemos reservado una lista de cambios llamada ignore-on-commit. Cualquier fichero añadido a esta lista de cambios se desmarcará automáticamente en el diálogo de confirmación. Aún puede confirmar los cambios, pero tendrá que seleccionarlo manualmente en el diálogo de confirmación.

#### 5.3.1.4.4. Mensajes de registro de confirmación

Asegúrese de introducir un mensaje de registro que describa los cambios que está confirmando. Esto le ayudará a saber qué ocurrió y cuando según navegue por los mensajes de registro del proyecto en el futuro. El mensaje puede ser tan extenso o escueto como desee; muchos proyectos tienen directrices sobre qué debe incluirse en ellos, el idioma que debe utilizarse, y a veces incluso un formato estricto.

Puede aplicar formatos sencillos en sus mensajes de registro utilizando una convención similar a la usada en los emails. Para aplicar un estilo a un texto, utilice *\*texto\** para la negrita, \_texto\_ para el subrayado, y ^texto^ para la cursiva.

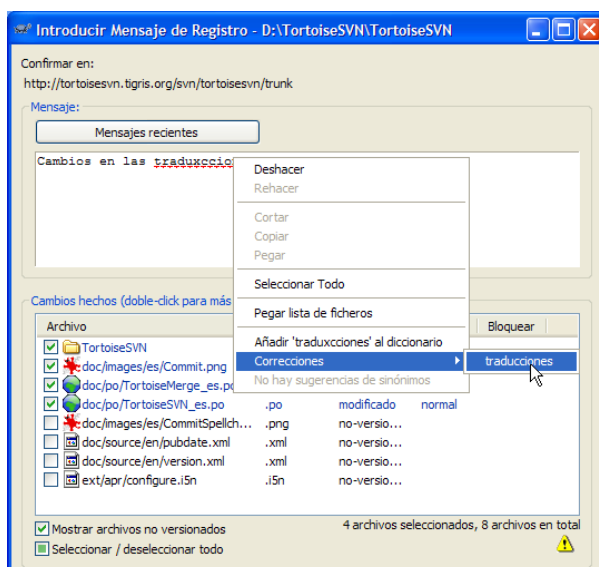


Figura 5.3.19: El corrector ortográfico del diálogo de Confirmación

TortoiseSVN incluye un corrector ortográfico para ayudarle a escribir sus mensajes de registro correctamente. Este corrector señalará cualquier palabra mal escrita. Utilice el menú contextual para acceder a las correcciones sugeridas. Por supuesto, el corrector no conoce *todos* los términos técnicos que utiliza, así que a veces palabras bien escritas aparecerán como errores. Pero no se preocupe. Puede simplemente añadirlas a su diccionario personal utilizando el menú contextual.

La ventana de mensajes de registro también incluye una facilidad de autocompletar nombres de ficheros y funciones. Esto utiliza expresiones regulares para extraer clases y nombres de funciones de los ficheros (de texto) que está confirmando, y también los propios nombres de ficheros. Si una palabra que está tecleando concuerda con algo en la lista (después de haber tecleado al menos 3 caracteres, o de pulsar **Ctrl+Espacio**), aparecerá un desplegable que le permitirá seleccionar el nombre completo.

#### 5.3.1.4.5. Progreso de confirmación

Tras pulsar Aceptar aparece un diálogo mostrando el progreso de la confirmación.

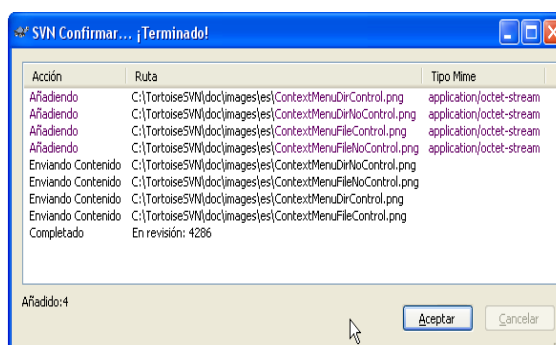


Figura 5.3.20: El diálogo Progreso mostrando el progreso de una confirmación

El diálogo de progreso utiliza una codificación de colores para resaltar las diferentes acciones de confirmación:

- Azul Confirmando una modificación.
- Púrpura Confirmando un ítem añadido.
- Rojo oscuro Confirmando un borrado o un reemplazo.
- Negro Todos los demás ítems.

### 5.3.1.5. Actualice su copia de trabajo con los cambios de otros

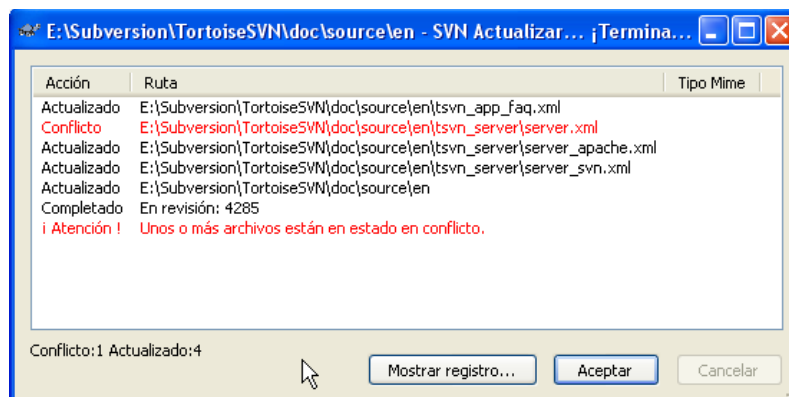


Figura 5.3.21: Diálogo de progreso mostrando una actualización terminada

Periódicamente, debería asegurarse de que los cambios que hacen los demás se incorporen en su copia de trabajo local. El proceso de incorporar los cambios desde el servidor a su copia de trabajo local se conoce como *actualización*. El diálogo de progreso utiliza un código de colores para resaltar diferentes acciones de actualización:

- **Púrpura** Nuevo ítem añadido a su copia de trabajo
- **Rojo oscuro** Ítem redundante borrado de su copia de trabajo, o ítem faltante reemplazado en su copia de trabajo.
- **Verde** Cambios del repositorio que se han fusionado satisfactoriamente con sus cambios locales.
- **Rojo brillante** Cambios del repositorio fusionados con sus cambios locales, pero que han dado lugar a conflictos que debe resolver.
- **Negro** Ítems sin cambios en su copia de trabajo actualizados con una versión más nueva desde el repositorio.

### 5.3.1.6. Resolviendo conflictos

De vez en cuando, obtendrá un *conflicto* cuando actualice/fusione sus ficheros desde el repositorio o cuando cambie su copia de trabajo a una URL diferente. Hay dos tipos de conflictos:



**Conflictos de fichero** .- Un conflicto de fichero ocurre si dos (o más) desarrolladores han cambiado las mismas líneas de un fichero.

**Conflictos de Árboles.-** Un conflicto de árbol ocurre cuando un desarrollador mueve/renombra/elimina un fichero o una carpeta, que otro desarrollador también ha movido/renombrado/borrado, o quizás lo haya modificado.

#### 5.3.1.6.1. Conflictos de ficheros

Un conflicto de fichero ocurre cuando uno o más desarrolladores han hecho cambios en las mismas líneas de un fichero. Dado que Subversion no sabe nada de su proyecto, delega la resolución de los conflictos en los desarrolladores. Cuando se le informa de un conflicto, debería abrir el fichero en cuestión, y buscar líneas que empiecen con el texto.

<<<<<<<. El área conflictiva se marca así:

```
<<<<<<< nombre-del-fichero
          sus cambios
=====
          código fusionado del repositorio
>>>>>>> revisión
```

Además, para cada fichero en conflicto Subversion deja tres ficheros adicionales en su directorio:

nombre -del-fichero.ext.mine

Este es su fichero tal y como estaba en su copia de trabajo antes de que actualizara su copia de trabajo - esto es, sin marcadores de conflicto. Este fichero tiene sus últimos cambios en él y nada más.

nombre-del-fichero.ext.rREV-ANTIGUA

Este es el fichero que era la revisión BASE antes de que actualizara su copia de trabajo. Esto es, el fichero que obtuvo antes de empezar a hacer sus últimos cambios.

nombre-del-fichero.ext.rREV-NUEVA



#### **5.3.1.6.2. Conflictos de árbol**

Un conflicto de árbol ocurre cuando un desarrollador mueve/renombra/elimina un fichero o una carpeta, que otro desarrollador también ha movido/renombrado/borrado, o quizás lo haya modificado. Hay diferentes situaciones que puede resultar en un conflicto de árbol, y todas ellas requieren pasos diferentes para resolver el conflicto.

Cuando se elimina un fichero de forma local en Subversion, el fichero también se elimina del sistema local de ficheros, por lo que incluso si es parte de un conflicto de árbol no se puede mostrar una sobreimpresión de conflicto y no puede hacer clic con el botón derecho sobre él para resolver el conflicto. En este caso, utilice el diálogo Comprobar modificaciones para acceder a la opción Editar conflictos.

TortoiseSVN puede ayudarle a encontrar el lugar correcto para fusionar los cambios, pero puede que necesite realizar un trabajo adicional para arreglar los conflictos. Recuerde que tras una actualización la BASE de trabajo siempre contendrá la revisión de cada ítem tal y como estaba en el repositorio en el momento de la actualización. Si revierte un cambio tras la actualización, se revierte a su estado del repositorio, no a como estaba cuando empezó a hacer sus propios cambios locales.

#### **5.3.1.7. Obteniendo información del estado**

Mientras está trabajando en su copia de trabajo a menudo necesitará saber qué ficheros ha cambiado/añadido/borrado o renombrado, o incluso qué ficheros han sido cambiados y confirmados por los demás.

##### **5.3.1.7.1. Iconos sobreimpresionados**

Ahora que ha obtenido una copia de trabajo desde un repositorio de Subversion, puede ver sus ficheros en el explorador de Windows con los iconos cambiados. Ésta es una de las razones por las que TortoiseSVN es tan popular. TortoiseSVN añade lo que se llama un icono sobreimpresionado al icono de cada



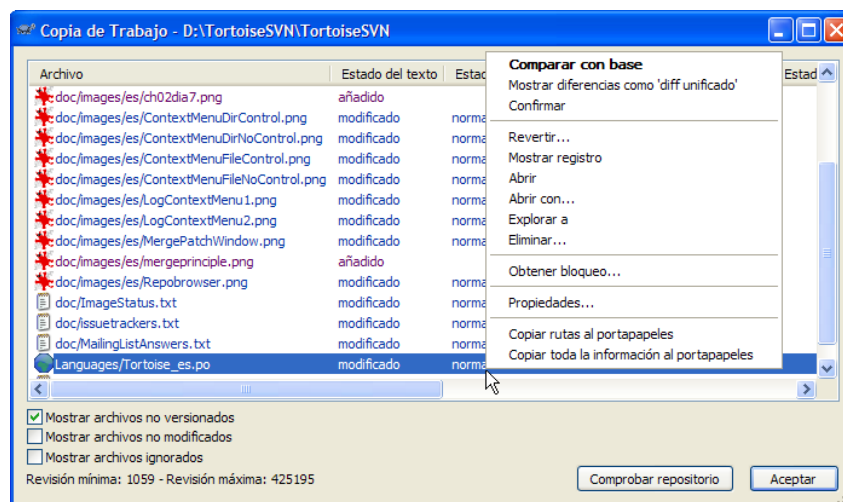
fichero que se superpone al icono original del fichero. Dependiendo del estado en Subversion del fichero, el icono sobreimpresionado es diferente.

### 5.3.1.7.2. Columnas de TortoiseSVN en el Explorador de Windows

Se puede ver la misma información que está disponible en los iconos sobreimpresionados como columnas adicionales en la Vista Detalles del Explorador de Windows.

Simplemente haga click con el botón derecho en la cabecera de una columna y seleccione Más... en el menú contextual que aparece. Se mostrará un diálogo donde puede especificar las columnas que se mostrarán en la “vista Detalles”, y su orden. Baje hasta que vea las entradas que empiezan por SVN. Marque aquellas que desee mostrar y cierre el diálogo pulsando Aceptar. Las columnas aparecerán a la derecha de las que ya se mostraban. Puede reorganizarlas utilizando arrastrar y soltar, o cambiarlas de tamaño, para que se ajusten a sus necesidades.

### 5.3.1.7.3. Estado local y remoto



**Figura 5.3.22. Comprobar modificaciones**

A menudo es muy útil saber qué ficheros ha cambiado y también qué ficheros han cambiado y confirmado los demás. Ahí es donde viene bien el comando TortoiseSVN → Comprobar Modificaciones.... Este diálogo le muestra todos los ficheros que ha



cambiado de alguna forma en su copia de trabajo, y además todos los ficheros no versionados que pueda tener.

#### 5.3.1.7.4. Viendo diferencias

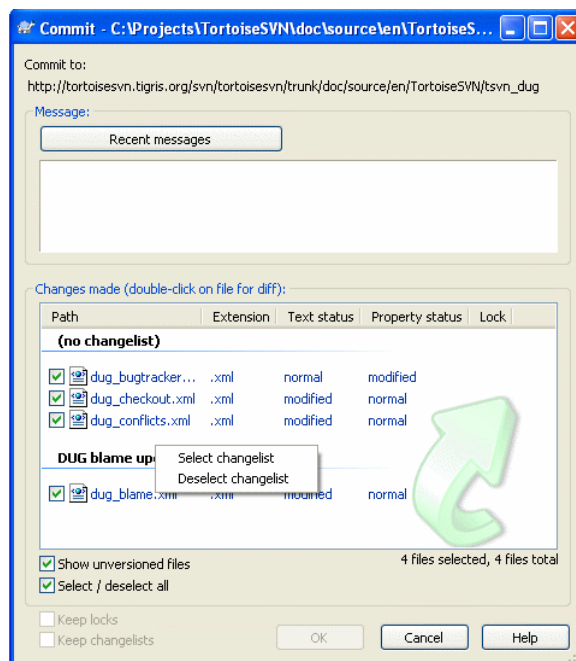
A menudo querrá mirar dentro de sus ficheros, para echar un vistazo a lo que ha cambiado. Puede llevar esto a cabo seleccionando un fichero que haya cambiado, y seleccionando Diferenciar desde el menú contextual de TortoiseSVN. Esto inicia el visor externo de diferencias, que comparará el fichero actual con la copia prístina (revisión BASE), que se guardó tras su obtención o tras la última actualización.

#### 5.3.1.8. Listas de cambios

En un mundo ideal, sólo trabajará en una cosa cada vez, y su copia de trabajo sólo contendrá un conjunto de cambios lógicos. Vale, de vuelta al mundo real. A menudo ocurre que tiene que trabajar en varias tareas sin relación entre sí a la vez, y cuando mira en el diálogo de confirmar, todos los cambios están juntos y mezclados. La característica *lista de cambios* le ayuda a hacer agrupaciones de ficheros, facilitando ver qué se está haciendo. Por supuesto esto sólo funciona si los cambios no se superponen. Si dos tareas diferentes afectan al mismo archivo, no hay forma de separar los cambios.

En el diálogo de confirmación puede ver esos mismos ficheros, agrupados por listas de cambios. Además de dar una indicación visual inmediata de las agrupaciones, también puede utilizar los encabezados de grupo para seleccionar qué ficheros confirmar.

En **XP** hay un menú contextual que aparece cuando hace click con el botón derecho en una cabecera de grupo, y que le ofrece la posibilidad de marcar o desmarcar todas las entradas de grupo. En Vista sin embargo el menú contextual no es necesario. Haga click en la cabecera de grupo para seleccionar todas las entradas, y luego marque la casilla de una de las entradas seleccionadas para marcarlas todas.



**Figura 5.3.23: Diálogo de confirmación con listas de cambios**

#### 5.3.1.9. Diálogo de Registro de revisiones

Para cada cambio que haga y confirme, debería proporcionar un mensaje de registro de ese cambio. Así podrá averiguar después qué cambios hizo y por qué, y tendrá un registro detallado para su proceso de desarrollo.

El diálogo de Registro de revisiones recopila todos esos mensajes de registro y se los enseña. La pantalla se divide en tres paneles.

- El panel superior le muestra una lista de revisiones donde se confirmaron cambios a los ficheros/carpetas. Este sumario incluye la fecha y la hora, la persona que confirmó la revisión y el inicio del mensaje de registro. Las líneas azules indican que algo se ha copiado a esta línea de desarrollo (quizás desde una rama).
- El panel medio le muestra el mensaje de registro completo para la revisión seleccionada.
- El panel inferior le muestra una lista de todos los ficheros y carpetas que se cambiaron como parte de la revisión seleccionada.

Pero hace mucho más que eso - le proporciona comandos del menú contextual que puede utilizar para obtener aún más información de la historia del proyecto.

#### 5.3.1.9.1. Invocando el diálogo de Registro de revisiones

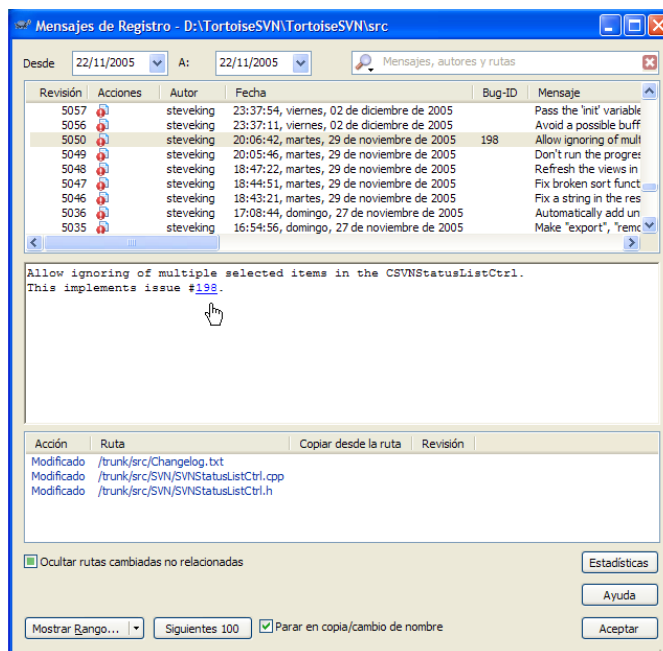


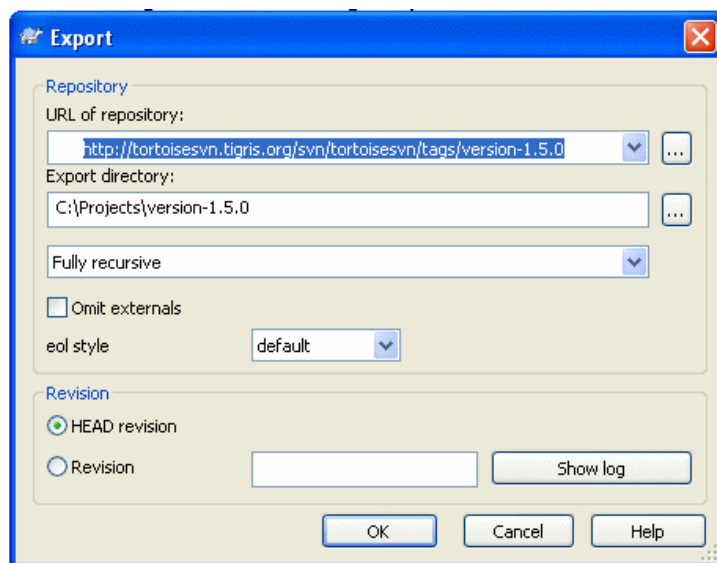
Figura 5.3.24: El diálogo de Registro de revisiones

Hay varios lugares desde los que puede mostrar el diálogo de Registro:

- Desde el submenú contextual de TortoiseSVN
- Desde la página de propiedades
- Desde el diálogo de Progreso después de que termine una actualización. En ese caso el diálogo de Registro sólo le mostrará aquellas revisiones que cambiaron desde su última actualización

#### 5.3.1.10 Exportando una copia de trabajo de Subversion

A veces el programador quiere una copia de su árbol de trabajo sin ninguno de esos directorios .svn, por ejemplo para crear un fichero comprimido de sus fuentes, o para exportarlo a un servidor web. En vez de hacer una copia y borrar todos esos directorios .svn manualmente, TortoiseSVN ofrece el comando TortoiseSVN → Exportar.... Las operaciones exportar desde una URL y exportar desde una copia de trabajo se tratan de forma ligeramente distinta.



**Figura 5.3.25. Diálogo de exportar**

Si ejecuta este comando sobre una carpeta sin versionar, TortoiseSVN asumirá que la carpeta de destino es el objetivo, y abrirá un diálogo para que introduzca la URL y la revisión desde la que desea realizar la exportación. Este diálogo tiene opciones para exportar sólo la carpeta de más alto nivel, para omitir las referencias externas, y para obligar un estilo de fin de línea concreto en aquellos ficheros que tuvieran establecida la propiedad svn.

Por supuesto también puede exportar directamente desde el repositorio. Utilice el Navegador de repositorios para navegar al subárbol relevante en su repositorio, y luego utilice Menú contextual → Exportar. Obtendrá el diálogo Exportar desde URL descrito más arriba.

Si ejecuta este comando en su copia de trabajo, se le preguntará por el lugar donde guardar la copia de trabajo limpia sin las carpetas .svn. Por defecto, sólo se exportan los ficheros versionados, pero puede utilizar la casilla Exportar también los ficheros o versionados para incluir cualquier otro fichero no versionado que exista en su copia de trabajo y no esté en el repositorio. Las referencias externas utilizando svn:externals pueden omitirse si es necesario.

Otra forma de exportar su copia de trabajo es arrastrar con el botón derecho la carpeta con la copia de trabajo a otro lugar y elegir Menú contextual → SVN Exportar aquí o Menú contextual → SVN Exportar todo aquí. La segunda opción incluirá también los ficheros sin versionar.



Cuando exporte desde una copia de trabajo, si en la carpeta de destino ya existe otra con el nombre de la que está exportando, se le dará la opción de sobre escribir el contenido existente, o de crear una nueva carpeta con un nombre generado automáticamente, por ejemplo, Destino (1).

#### **5.3.1.10.1 Exportando ficheros sueltos**

El diálogo de exportar no permite exportar ficheros sueltos, aunque Subversion puede hacerlo.

Para exportar ficheros sueltos con TortoiseSVN, tendrá que utilizar el navegador de repositorios. Simplemente arrastre el o los ficheros que desea exportar desde el navegador de repositorios a donde los desee tener en el explorador, o utilice el menú contextual del navegador de repositorios para exportar los ficheros.

#### *Exportando un árbol de cambios*

Si quiere exportar una copia de su estructura de árbol del proyecto pero conteniendo sólo los ficheros que han cambiado en una revisión en concreto, o entre dos revisiones, utilice la característica de comparar revisiones.

#### **5.3.1.10.2 Eliminando una copia de trabajo del control de versiones**

A veces tiene una copia de trabajo que desea reconvertir en una carpeta normal sin los directorios .svn. Lo que realmente necesita es un comando exportar-en-el-sitio, que simplemente elimine las carpetas de control, en vez de generar un nuevo árbol limpio.

La respuesta es sorprendentemente sencilla - exporte el árbol en si mismo. TortoiseSVN detecta este caso especial y le pregunta si quiere convertir su copia de trabajo en no-versionada. Si responde sí, los directorios de control se eliminan y tendrá un simple árbol sin directorios de control.

### 5.3.1.11 Tutorial de RapidSVN

Es un cliente gráfico que nos permite manipular nuestros repositorios de Subversion. Esta puede ser una muy buena alternativa para usar en lugar del ya conocido TortoiseSVN para Windows ya que es multiplataforma.

Si desean probarlo en Debian o Ubuntu es muy fácil, bastará con instalar el paquete **rapidsvn** desde sus repositorios. Cabe mencionar que para **Ubuntu** deberán tener agregados los repositorios **universe**

#### 5.3.1.11.1 Instalación

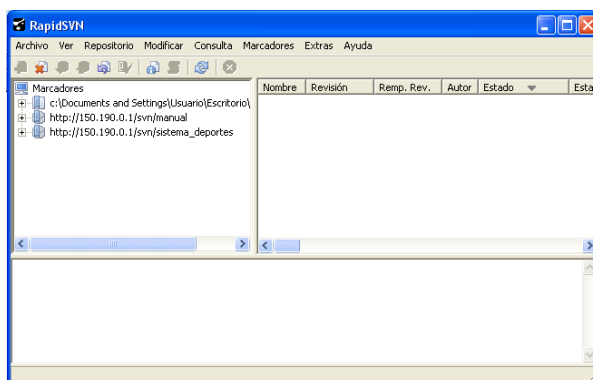
Para instalarlo, se tiene que ingresar como **root**. mediante el comando **sudo su root**

Ahora ejecutamos el siguiente comando:

```
sudo apt-get install rapidsvn
```

Listo, ha quedado instalado **RapidSVN**. Para usarlo, bastará con ejecutar el comando **rapidsvn**:

También podemos seleccionarlo del menú de **Gnome** desde: **Aplicaciones -> Programación RapidSVN**.



**Figura 5.3.26: Tutorial de RapidSVN**

RapidSVN tiene la noción de un servidor (repositorio) y un cliente (una copia de trabajo). Tanto los depósitos y las copias de trabajo puede ser el marcador. Al crear un favorito, el usuario tiene un acceso rápido o repositorio. Lo primero que un usuario debe hacer es marcar un repositorio

### 5.3.1.11.2 Configuración de las preferencias

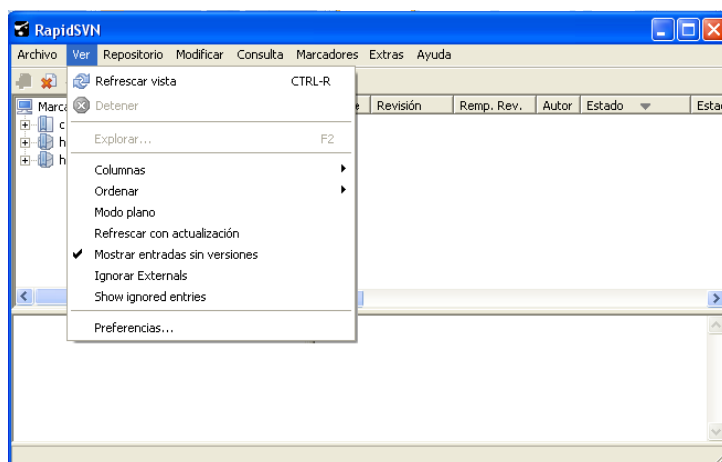


Figura 5.3.27: Tutorial de RapidSVN Preferencias

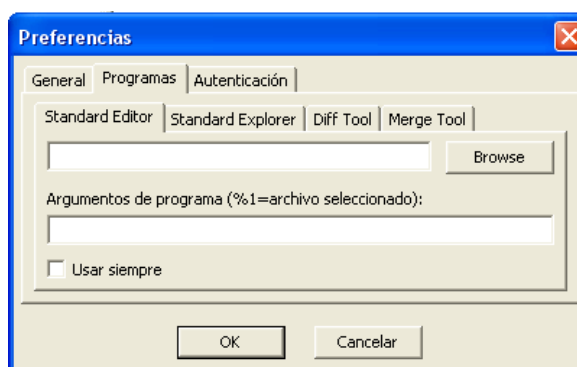


Figura 5.3.28: Tutorial de RapidSVN

### 5.3.1.11.3 Deshacer una edición local (a través de revertir):

1. Haga un cambio en el archivo que esté trabajando y guárdalo.
2. Luego haga clic sobre el archivo modificado y seleccione **Revertir**.
3. Haga clic en **Aceptar** al cuadro de dialogo Recuperar y observe que los cambios realizados en el paso 1 se han perdido.

### *Agregar un archivo (a través de agregar)*

1. Haga clic en el botón derecho sobre el archivo que quiere añadir y **Agregar**.
2. Luego selecciona confirmar
3. Escriba un mensaje en la ventana de registro y haga clic **en aceptar**.



#### 5.3.1.11.4 Añadir un directorio (a través de agregar recursiva)

1. Crear un nuevo directorio que hemos estado trabajando.
2. Agregar un archivo (puede crear subdirectorio, con más archivos en ella).
3. Volver a RapidSVN. En la ventana derecha te darás cuenta del nuevo directorio aparece como “no versionados”.
4. Clic derecho en el nuevo directorio y presione **Agregar Recursiva**, y **Confirmar**.
5. Escriba un mensaje en la ventana de registro y haga clic en **Aceptar**

El nuevo directorio y su contenido se han añadido al repositorio. Nota: Es posible añadir directorios vacíos.

#### 5.3.1.11.5 Añadir un directorio a través de importación

1. En el menú archivo seleccione importar
2. Asegúrese de especificar la ruta completa al nuevo directorio.
3. A continuación introduzca su ruta de acceso local en el camino de campo
4. Escriba algo en el “Enter mensaje de registro de campo”, “recursivas”, “ruta de tipo confirmar” y haga clic en **Aceptar**.

Esto agregara su directorio y todos los archivos y directorios en su proyecto. Desafortunadamente haciendo una importación no da su pago y a los archivos. Sus archivos están en el repositorio, pero no se pueden empezar a trabajar con ellos.

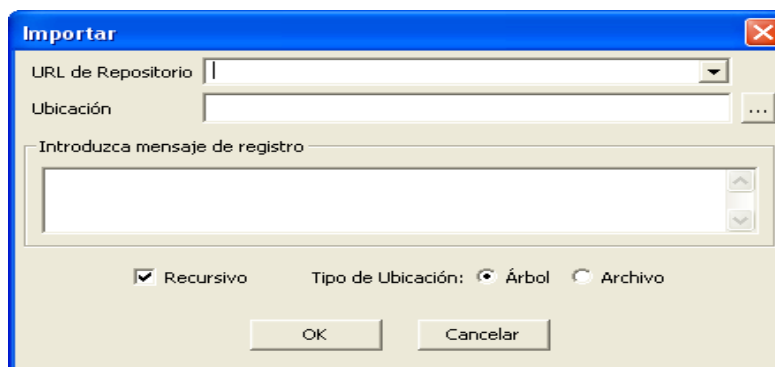
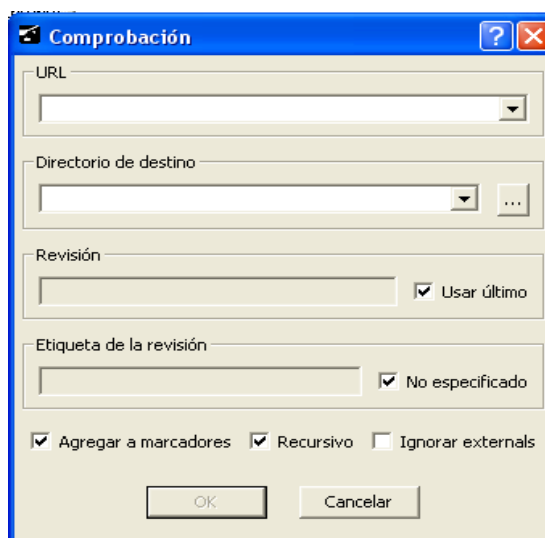


Figura 5.3.29: Tutorial de RapidSVN Importar

1. En el explorador de marcador, seleccione el directorio principal para el nuevo directorio. Esto puede ser el favorito repositorio o en un subdirectorio.
2. En la mano de la ventana derecha, botón derecho del ratón en una fila vacía y seleccione **crear directorio (F7)**
3. En el navegador de favoritos seleccione el nuevo directorio
4. Clic derecho y seleccione **nuevo pedido copia de trabajo**, especifique el “Directorio de destino” y haga clic en **Aceptar**

### ***Una copia de trabajo***

1. Haga clic en Repositorio de menú
2. Haga clic en el elemento de menú Pagar (ctrl-o), se abrirá el cuadro “pagar”
3. Introduzca la URL del repositorio Subversión en el “Cuadro de texto URL”
4. Seleccione un directorio local para su caja de arena (la copia de trabajo)
5. Haga clic en el botón **Aceptar**



**Figura 5.3.30: Tutorial de RapidSVN copia de trabajo**

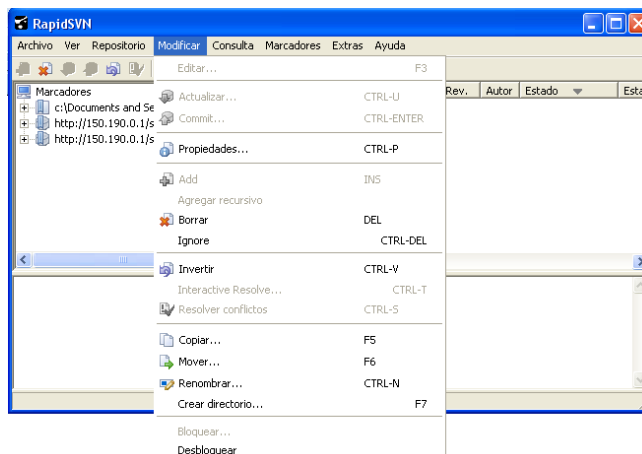
### ***Actualice su copia de trabajo***

Cuando se trabaja en un proyecto con un equipo, usted querrá actualizar su copia de trabajo para recibir los cambios realizados por otros desarrolladores en el proyecto. El comando de actualización hará que su copia de trabajo este con la última revisión en el repositorio.

- En el navegador de favoritos, haga clic en un directorio y seleccione **Actualizar**. Haga clic en **Aceptar** en el cuadro de diálogo de actualización.

- Si un archivo se ha eliminado en el repositorio, se eliminarán de la copia de trabajo.
- Si el archivo se ha eliminado localmente con la intención de recuperar la copia más reciente del repositorio- el revertir comando debería ser utilizado

**En modificar encontramos Copiar, Mover, Renombrar, etc.**



**Figura 5.3.31: Tutorial de RapidSVN actualizar**

Estas operaciones se pueden producir en el repositorio o en un marcador. Si el cambio se realiza en el repositorio de la operación se producirá de inmediato. Si el cambio se realiza en el marcador que los cambios necesitan ser confirmados.

Estas operaciones pueden hacerse en un directorio.

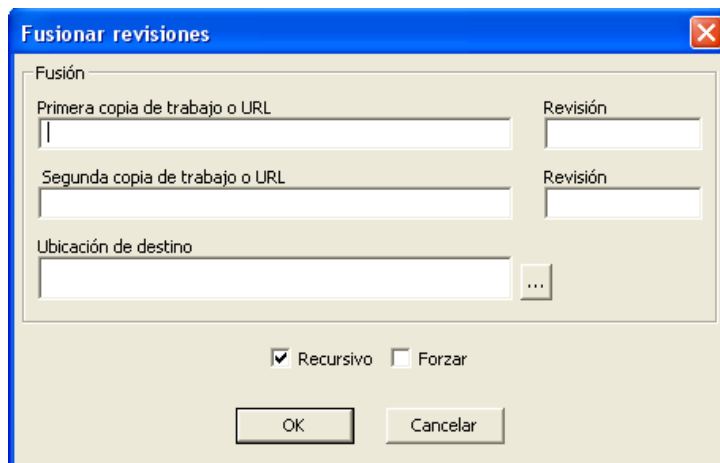
Si mueve archivos dentro de un marcador dará como resultado un error.

## **Función**

Suponiendo que vamos a combinar branch1 y branch2

1. Actualice su copia local branch1 desde el repositorio
2. Actualice su copia local branch2 desde el repositorio
3. Seleccione combinar
4. En la copia "En primer lugar de trabajo o URL", introduzca la dirección local de branch1
5. En la "Segunda copia de trabajo o URL", introduzca la dirección local de branch2
6. Introduzca la versiones en los dos "Revisión"
7. En el camino "destino", escriba la dirección local de la carpeta que desea que su función se cree y haga clic en **Aceptar**

8. Seleccione la ubicación donde la combinación de correspondencia se ha puesto y confirme en el repositorio remoto(utilice Ctrl-M)



**Figura 5.3.32: Tutorial de RapidSVN fusionar**

- RapidSVN no le permite borrar archivos modificados, solo si utiliza forzar a borrar se anulara la prohibición y se eliminara no bajo control de versiones.

## 5.4 Pruebas y Validación de Resultados

### 5.4.1 Análisis de Resultados

La fase de validación de la aplicación se la llevó a cabo en el departamento de desarrollo Informático (UDI) que se encuentra alojado en la carrera de Ingeniería en Sistema del A.E.I.R.N.N.R de la Universidad Nacional de Loja, en la biblioteca del A.E.I.R.N.N.R y con los estudiantes de la carrera de Ingeniería en Sistemas del quinto módulo paralelos A y B quienes estuvieron al docente Ing. Milton Labanda, y el Ing. Luis Chamba, inicié con el **Departamento de Desarrollo Informático** donde para el efecto las pruebas se iniciaron el día 20 de Octubre del 2009, esta etapa duró un tiempo aproximado de 2 meses lo que permitió depurar errores existentes en este módulo, realizados las correcciones necesarias el servidor funcionó con normalidad el tiempo propuesto. Cumplido esto, se procedió a realizar la validación mediante la aplicación de la encuesta al personal que hizo uso del sistema el Mg Sc. Ing German Patricio Villamarin Coronel, para esto se aplicó 1 encuestas. **(Ver Anexo 2)**



Para continuar con las pruebas se trabajó con el departamento de la Biblioteca del A.E.I.R.N.N.R, donde se puso en funcionamiento la aplicación el día 04 de Enero de 2010, el cual se instaló el cliente de Subversion para subir los proyectos de tesis terminadas al servidor de repositorios y así estar bajo el control de la Subversion, que se encuentra alojado en la carrera de Ingeniería en Sistemas, tuvo un funcionamiento correcto, seguido se aplicó las encuesta al usuario final quien se encuentra a cargo la Lic. Sandra Castillo. **(Ver Anexo 3)**

Finalmente para culminar con la fase de pruebas y validación, se realizó la instalación del cliente de Subversion TortoiseSVN en el **Laboratorio de computo N° 1 del A.E.I.R.N.N.R**, las pruebas las realizaron los usuarios estudiantes en el mes de enero y Febrero, para esto ayudaron 4 profesionales: Ingeniero Hernán Torres, Ing. Luis Chamba, Ing Ivan Borrero, Ing Patricio Villamarin, los cuales brindaron su tiempo para realizar las pruebas, y así con los proyectos que se encontraban en ese tiempo en desarrollo, lo que mejoró sustancialmente el desarrollo de las tareas tanto en tiempo y fusión de código fuente, el sistema funcionó de forma satisfactoria no se presentaron errores de ejecución, una vez funcionando de forma correcta se aplicó las encuestas para la validación del servidor de repositorios de Subversion. **(Ver Anexo 3)**

Para el planteamiento de la encuestas se tomaron en cuenta parámetros importantes como son el cliente de subversión sea amigable, fácil manejo, rendimiento en la ejecución en las diferentes tareas, entre otros aspectos importantes.

Luego de terminar las configuraciones del software, para la implementación del Servicio de repositorios SVN en Linux distribución (Ubuntu) en la Universidad Nacional de Loja del A.E.I.R.N.N.R de la carrera de Ingeniería en Sistemas, en el departamento de Desarrollo Informático (U.D.I.) se lo describiré a continuación.

El servidor de repositorio SVN ubicado en el departamento antes llamado Unidad de Desarrollo Informático (U.D.I), actualmente situado en la carrera de Ingeniería en Sistemas del A.E.I.R.N.N.R., está funcionando adecuadamente desde el día 12 de octubre del 2009 hasta la actualidad.



Actualmente el servidor tiene la ip 172.16.44.119, mediante acceso al repositorio en las direcciones:

[http://172.16.44.119/svn/tesis/nombre\\_del\\_repositorio](http://172.16.44.119/svn/tesis/nombre_del_repositorio)

[http://172.16.44.119/svn/pruebas/nombre\\_del\\_repositorio](http://172.16.44.119/svn/pruebas/nombre_del_repositorio)

[http://172.16.44.119/svn/tutoriales/nombre\\_del\\_repositorio](http://172.16.44.119/svn/tutoriales/nombre_del_repositorio)

[http://172.16.44.119/svn/programas/nombre\\_del\\_repositorio](http://172.16.44.119/svn/programas/nombre_del_repositorio).

Al momento de acceder al repositorio con cualquiera de los clientes que soporta Subversion presentara un mensaje de control del servidor SVN, el cual permite ingresar el usuario y la clave para acceder, según el nivel de acceso que se le otorgue.

Existen usuarios para cada proyecto según los permisos otorgados. Además los usuarios pueden acceder desde un navegador.

Los usuarios principales tienen permitido acceder a todos los repositorios creados de lectura y escritura.

Los usuarios (clientes) pueden comunicarse de acuerdo al privilegio o grupo de usuario

Como parte del proceso de pruebas y validación, se ha realizado:

**Subversionar las Tesis terminadas de la carrera de ingeniería en sistemas a los repositorios correspondientes en el servidor:**

**Datos del responsable:**

**Nombre:** Lic. Sandra Castillo

**Institución:** Universidad Nacional de Loja.

**Departamento:** Biblioteca del A.E.I.R.N.N.R.

**Fecha:** Loja, 04 de enero de 2010

Estimada Licenciada, le solicito respetuosamente aplicar el siguiente plan de validación, el cual tiene como finalidad, comprobar el correcto funcionamiento de la



herramienta SVN Subversion para el control de los proyectos en desarrollo y desarrollados, por lo cual le solicito me permita subir los proyectos que se encuentre a su responsabilidad

1. Instalación del cliente de repositorio **TortoiseSVN**.
2. El usuario ingresa con el nombre de usuario y clave de administrador.
3. La herramienta presenta la vista amigable
4. Acceso vía http
5. Los archivos de las tesis terminadas suben al servidor de repositorio normalmente
6. El usuario modifica los datos del repositorio.
7. La herramienta almacena los cambios.

Resultado del plan

**Pasó X**

**Falló**

**Gracias por su colaboración**

Licenciada Sandra Castillo

### **Plan de Pruebas II en la carrera de Ingeniería en Sistemas:**

El Ing. Milton Labanda y el Ing. Patricio Villamarín, con la ayuda de los estudiantes del quinto A, y B de la promoción 2007, quienes subieron sus proyectos al servidor de repositorios de Subversion, y actualizaron los proyectos el día Jueves 14 de enero del 2010 a las 16:00:00 hasta las 18:00:00 con la ayuda del Ingeniero Hernán Torres y el Ing. Luis Chamba para recabar información sobre el servicio de **SUBVERSION** en la institución. La encuesta se adjunta a continuación:

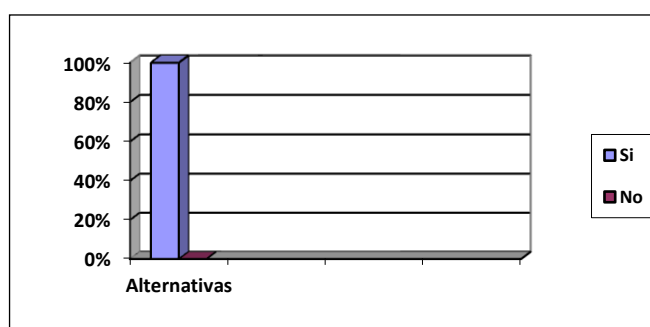
Para constancia de que el servidor está funcionando correctamente se adjunta la certificación del departamento de la Unidad de Desarrollo Informático de la Universidad Nacional de Loja (Anexo 2), además se planteó una prueba con los estudiantes de los quintos módulos acerca del servicio del manejo del repositorio (Anexo 4) aplicada al personal de los siguientes departamentos:

***Resultados de la Encuesta Aplicada a los Usuarios del Servicio de repositorios de Subversion***

**1.- Considera usted que el servicio de repositorio Subversion, implementado en el departamento de la Unidad de Desarrollo Informático de la Universidad Nacional de Loja es útil.**

Alternativas	Frecuencia	%
Sí	6	100
No	0	0
Total	6	100

**Tabla 5.4.1: Resultados Pregunta 1**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**



**Figura 5.4.1: Resultado Pregunta 1**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**





El 100% de los encuestados consideran que el servicio de repositorio Subversion implementado en la Universidad Nacional de Loja es útil.

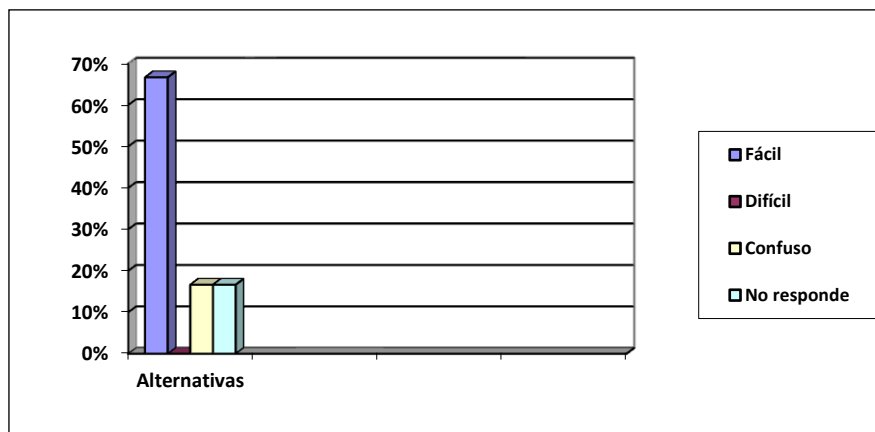
Al explicar los encuestados el por qué describen lo siguiente:

- Permite comunicarse de manera fácil desde la red, ahorrando recursos.
- Porque es más barato implementar a largo plazo frente a las soluciones tradicionales.
- Economiza Tiempo y recursos.
- Permite tener una comunicación fluida entre los diferentes desarrolladores.
- 

**2.- Usuario Subversion con Software. El manejo del Software TortoiseSVN es:**

Alternativas	Frecuencia	%
Fácil	4	66.7
Difícil	0	0
Confuso	1	16,65
No Responde	1	16,65
Total	6	100

**Tabla 5.4.2: Resultados pregunta 2**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**



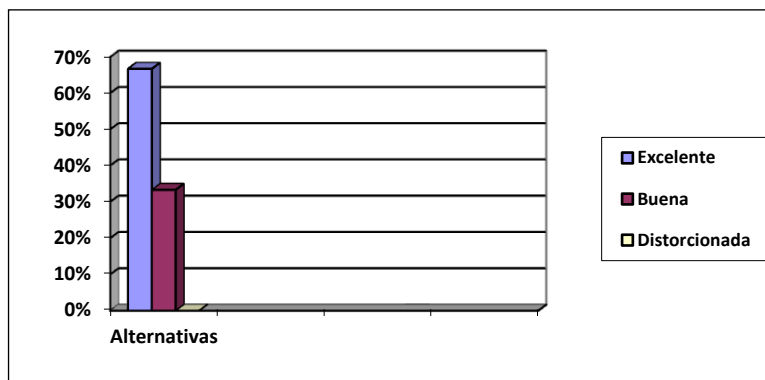
**Figura 5.4.2: Resultado Pregunta 2**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**

De lo anterior se concluye que el 66,7% de los encuestados responden que el manejo del Software TortoiseSVN es fácil; el 16,65% de los encuestados afirma que el manejo es confuso; otro 16,65% no contesta a la pregunta. Ningún encuestado dice que el manejo es difícil.

### 3.- La rapidez de actualización al recibir o realizar una actualización es:

Alternativas	Frecuencia	%
Excelente	4	66.7
Buena	2	33.3
Distorsionada	0	0
Total	6	100

**Tabla 5.4.3: Resultados pregunta 3**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**



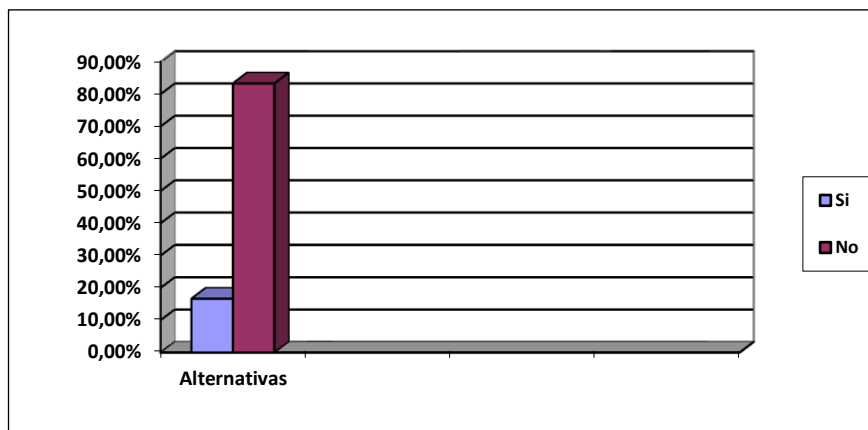
**Figura 5.4.3: Resultado Pregunta 3**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**

En base a lo anterior se concluye que el 66.7% considera que la calidad del servicio es excelente, mientras que el 33.3% piensa que esta calidades buena. Ninguno de los encuestados opina que es la calidad es distorsionada.

#### 4.- Ha tenido problemas con el servicio de Subversion.

Alternativa	Frecuencia	%
Sí	1	16.65
No	5	83.35
Total	6	100

**Tabla 5.4.4: Resultados pregunta 4**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**



**Figura 5.4.4: Resultado Pregunta 4**  
**Fuente: Encuesta a Usuarios SUBVERSION**  
**Elaboración: El Autor**

El 16.65% dice haber tenido problemas con el servicio de Subversion, los problemas son:

- Existe problemas con el servicio de Subversion cuando existe un corte de energía eléctrica.
- Existe problemas con el servicio de Subversion cuando el servidor se encuentra apagado.
- Existe problemas con el servicio de Subversion cuando no exista conexión, por medio de la intranet.

El 83.35% afirma que no ha tenido problemas al momento de hacer uso del servicio de Subversion.

Con lo expuesto anteriormente, la culminación formal del proyecto de tesis **“CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMÁTICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA”**, se daría con la aprobación de las pruebas de validación.



## 6 VALORACIÓN TÉCNICA - ECONÓMICA – AMBIENTAL

En la culminación del presente proyecto se ha evidenciado que la aplicación cumple con los objetivos planteados al inicio del mismo, entre las características que posee la aplicación están:

- Mejora el tiempo de fusión de código fuente de los proyectos de los estudiantes o desarrolladores.
- La interfaz gráfica del cliente poseen una apariencia visual que es amigable al usuario final.
- Gran aceptación por parte de los diferentes usuarios, ya que ahorra tiempo y recursos.

Lo que se refiere a la inversión económica del presente proyecto, se a podido dar cuenta que el costo es mínimo, debido a que se ha tomado en cuenta lo que conocemos actualmente como software Libre GNU. Además el equipo para el alojamiento del servidor de repositorio será donado por mi persona.

A continuación se detalla los recursos utilizados para la elaboración del proyecto:

### Recurso Humano:

Descripción	Cantidad	# Horas V	Unitario V	Total
Programador	1	500	5.00	2500.00
Asesor	1	20	10.00	200.00
Director de Tesis	1	-	-	0.00
				2700.00

**Tabla 6.1: Recursos Humanos**



### Recursos Técnicos Hardware:

Descripción	Cantidad	# Horas V	Unitario V	Total
Computador PIV, 768 Mb, 120Gb	1	-	350.00	350.00
Impresora	1	-	60	60.00
Dispositivo USB	1	-	15.00	15.00
				375.00

Tabla 6.2: Recursos Técnicos Hardware:

### Recursos Técnicos Software:

Descripción	Cantidad	# Horas V	Unitario V	Total
S. Operativo GNU/Linux	1	-	0.00	00.00
Paquetes, dependencias svn Apache 2.0 svn Cliente svn Tortoiseclientsvn (Windows)	1	-	-	0.00
Open Office	1	-	0.00	00.00
				00.00

Tabla 6.3: Recursos Técnicos Software

### Recursos Materiales:

Descripción	Cantidad	Unitario V	Total
Tinta Impresora.	5	36.00	180,00
Resma Papel	5	3.20-	16.00
Copias	1350	0.02	27.00
Anillados	10	1.50	15.00
Transporte	-	60	60.00
Servicios Básicos	-	75,00	75.00



Empastados	12	6.00	72.00
Derechos Estudiantiles	6	5	30.00
Varios	-	50,00	50
			525.00

**Tabla 6.4: Recursos Materiales**

**Total Inversión:**

<b>Recurso</b>	<b>Cantidad</b>	<b>V. Unitario</b>	<b>V. Total</b>
Humano	1	36.00	2700.00
Técnico Hardware	1	3.20-	375.00
Técnico Software	1	0.02	0.00
Recursos Materiales	1	1.50	525.00
			<b>3600.00</b>

**Tabla 6.5: Total Inversión**



## 7. CONCLUSIONES

Al terminar la presente investigación en el estudio de la implantación y configuración de la herramienta de Subversion, en base a una extensa fundamentación teórica utilizando los métodos antes mencionados para la presente tesis, y como fruto de los conocimientos, pruebas realizadas y experiencias que se ha podido obtener de la presente, se ha llegado a las siguientes conclusiones:

- La realidad en la que se encuentra el A.E.I.R.N.N.R de la carrera de Ingeniería en Sistema de la Universidad Nacional de Loja, permitió identificar problemas como: no contaba con un equipo para la instalación del servidor de repositorios svn en Linux, para los proyectos de desarrollo e investigaciones que realizan los estudiantes cada módulo.
- En los laboratorios de cómputo del A.E.I.R.N.N.R de la carrera de Ingeniería en Sistema de la Universidad Nacional de Loja se pudo detectar en el año 2009, que en los equipos no se encuentran instalado sistema operativo Linux, por lo que se optó en instalar un cliente de versiones svn en Windows llamado TortoiseSvn.
- La distribución que se empleó para el servidor de repositorio de versiones svn es Ubuntu 10.04, el cual permitió integrar el servidor de apache2 con svn y sus respectivas dependencias.
- El sistema de control de subversiones permitió realizar las actividades de administración de los archivos mediante el uso de nombres de usuario y contraseñas con la ayuda del servidor de apache, lo cual da seguridad en el acceso al sistema para registrar quien realizó un cambio o para mantener además un histórico del mismo.
- La configuración de los archivos para la Subversion, para la creación del repositorio padre es fácil de realizar. Siempre y cuando sigan las instrucciones del manual del administrador el cual indicará paso a paso la creación de los mismos.
- En la instalación del repositorio svn, se optó por instalar y configurar Subversion ya que es una herramienta centralizada de gran utilidad al momento de administración de los proyectos, a la diferencia de otras herramientas distribuidas como el Bazaar que aún no poseen un control en los sistemas versionados.





## 8 RECOMENDACIONES

Al finalizar el trabajo investigativo considero pertinente realizar las siguientes recomendaciones:

- El servidor de repositorio de Subversion debe estar iniciado, el cual permitirá que los estudiantes y docentes puedan enviar y recibir sus cambios del proyecto a realizar y así optimizar recursos y tiempo en la fusión de código.
- Realizar auditoría a los repositorios, para validar si cumplen con los requerimientos establecidos.
- Los cambios de configuración de los archivos `apache2.conf`, la creación de usuarios para acceder a los repositorios y dar permisos a los mismos se debe realizar por una persona que será la responsable de la administración del servidor.
- Se debe tener una infraestructura de red óptima para el envío y recepción de datos desde el servidor hacia los clientes de Subversion, para su mejor funcionalidad.
- Deben darle uso a los repositorios creados en el servidor de Subversion, para tener una mejor ordenación y control de los mismos.
- Para una reinstalación del servidor se recomienda **respaldar los repositorios creados** que se encuentran en la ruta establecida, el archivo, `dav_svn_passwd`, el archivo de configuración `apache2.conf`, y `svn_access_control`,
- Se debe impulsar el uso de tecnologías de software libre, ya que dichas tecnologías tienen un buen desempeño y aportan de manera efectiva a la utilización de Subversion para el desarrollo de software de calidad. y así optimizar recurso y tiempo.
- Los estudiantes que realicen los proyectos de desarrollo de software, deben usar a la herramienta de subversiones utilizando los estándares que permitan delimitar y establecer la calidad del software que se ha desarrollado o a desarrollar, ya que es un servidor centralizado de gran ayuda al fusionar código fuente del proyecto.



## 9 BIBLIOGRAFÍA Y REFERENCIAS

### Sitios web:

1. monta-un-servidor-web-en-ubuntu-con-apache
  - <http://www.alejandrox.com/2008/02/monta-un-servidor-web-en-ubuntu-con-apache/> [Consulta: 24 enero 2009]
2. metodo-cientifico-experimental
  - <http://apuntes.rincondelvago.com/metodo-cientifico-experimental.html> [Consulta: 26 enero 2009]
3. *Hosting SVN gratuito*
  - <http://code.google.com/hosting/> [Consulta: 28 enero 2009]
4. Subversion
  - <http://www.collab.net/downloads/subversion/> [Consulta: 29 enero 2009]
5. subversion-over-apache-2-on-ubuntu/
  - <http://davidwinter.me.uk/articles/2006/02/16/subversion-over-apache-2-on-ubuntu/> [Consulta: 24 enero 2009]
6. eAthena Support Board latest news Blog dedicado en su mayoría a SVN
  - [\[http://www.eathena.ws/board/index.php?showtopic=124056\]](http://www.eathena.ws/board/index.php?showtopic=124056) [Consulta: 24 enero 2009]
7. Control\_de\_versiones
  - [http://es.wikipedia.org/wiki/Control\\_de\\_versiones](http://es.wikipedia.org/wiki/Control_de_versiones) [Consulta: 24 enero 2009]
8. Distribuciones Linux
  - [\[http://es.wikipedia.org/wiki/Lista\\_de\\_Distribuci%C3%B3n\\_Linux\]](http://es.wikipedia.org/wiki/Lista_de_Distribuci%C3%B3n_Linux), [Consulta: 16 enero]
9. Subversion
  - <http://es.wikipedia.org/wiki/Subversion>
10. Software Libre, Programación y algo mas... Blog dedicado en su mayoría a Linux, software libre y programación;
  - [\[http://estrada-david.blogspot.com/2009/01/netbeanssvn.html\]](http://estrada-david.blogspot.com/2009/01/netbeanssvn.html), [Consulta: 14 enero 2009].
11. Rafael Martínez El rincón de Linux
  - [\[http://www.linux-es.org/distribuciones\]](http://www.linux-es.org/distribuciones), [Consulta: 15 enero 2009].
12. Rafael Martínez. Copyright 1998-2009 El rincón de Linux -
  - [\[http://www.linux-es.org/sobre\\_linux\]](http://www.linux-es.org/sobre_linux), [Consulta: 6 enero 2009].
13. Distribuciones Linux
  - [\[http://www.monografias.com/trabajos14/linux/linux.shtml\]](http://www.monografias.com/trabajos14/linux/linux.shtml), [Consulta: 19 enero 2009].
14. ManualUsuarioSvn
  - <http://plataforma.cenditel.gob.ve/wiki/ManualUsuarioSvn> [Consulta: 26 enero 2009]



## 15. Open Source Software Engineering Tools

- [<http://subversion.tigris.org/>], [Consulta: 22 enero 2009].

## 16. Sitio oficial de Subversion

- <http://subversion.tigris.org/>[Consulta: 28 enero 2009]

## 17. Libro libre 'Control de versiones con Subversion'

- <http://svnbook.red-bean.com/>

## 18. TortoiseSVN

- [http://tortoisesvn.net/docs/nightly/TortoiseSVN\\_es/tsvn-basics.html](http://tortoisesvn.net/docs/nightly/TortoiseSVN_es/tsvn-basics.html)][Consulta: 24 enero 2009]

## 19. tortoiseSvn

- <http://tortoisesvn.tigris.org>

## 20. Subversion

- <http://ututo.org/manual/index.php/Subversion>][Consulta: 26 enero 2009]

## 21. GreenstoneWiki.- Welcome to GreenstoneWiki: Documentation for Greenstone

- [[http://wiki.greenstone.org/wiki/index.php/Install\\_SVN\\_on\\_Linux](http://wiki.greenstone.org/wiki/index.php/Install_SVN_on_Linux)][Consulta: 27 enero]



## **10 ANEXOS**

### **Anexo 1: Anteproyecto**

#### **1. TITULO**

**“CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE  
REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA  
UNIDAD DE DESARROLLO INFORMATICO DEL A.E.I.R.N.N.R DE LA  
UNIVERSIDAD NACIONAL DE LOJA “**



## **2. PROBLEMÁTICA**

### **2.1 Situación Problemática**

Se suele observar todavía muchos entornos de programación donde el proceso de desarrollo presenta un único repositorio central de archivos en el cual varios programadores se encuentran trabajando al mismo tiempo. Lo anterior lleva a varios problemas como conflictos entre versiones de código, sobre escritura de del mismo, imposibilidad para devolverse a versiones anteriores del programa, dificultad para trabajar en diferentes funcionalidades al mismo tiempo, entre muchos otros más.

Básicamente un sistema para control de versiones facilita la administración de las distintas versiones para un proyecto en desarrollo. Los sistemas de almacenamiento centralizado son los más comunes, entre ellos tenemos SVN y CVS, estos crean un repositorio central en el cual el proyecto es almacenado, por otro lado, los desarrolladores pueden generar copias locales de trabajo donde deben introducir sus cambios y luego enviarlos al repositorio central.

Otra gran ventaja que ofrecen estas herramientas es la generación automática de un historial de cambios: quién modificó cierto archivo? cuándo? por qué? qué cambios hizo?, diferencias entre revisiones específicas, y muchas más.

Los aspectos más importantes de la Unidad de Desarrollo Informático (U.D.I.) son: el desarrollo de proyectos de software.

Los desarrolladores se enfrentan a diversos desafíos en el logro de sus objetivos en las Organizaciones que prestan sus servicios.

En términos concretos, el principal desafío es lograr el mejoramiento permanente del software que se crea en la Área de la Energía Industria y Recursos Naturales No Renovables de la carrera de Ingeniería en Sistemas del la Universidad Nacional de Loja, haciéndola más eficiente y más eficaz. Ser eficiente implica utilizar la cantidad mínima de recursos necesarios para el logro de los objetivos propuestos. Ser eficaz implica alcanzar dichos objetivos, de manera que sean aceptables. Estos dos factores conducen a mejores niveles de productividad.



La respuesta a este desafío, de mi parte, es el ahorro de tiempo y de Recurso Humano que labora en la Unidad de Desarrollo Informático, para lo cual se realizará un la Configuración e implementación de un repositorios de control versiones de proyectos de software utilizando SVN.

Con esto se podrá llevar a cabo un trabajo en conjunto entre mi persona y usuarios finales, permitiendo tener una aproximación acerca de cómo podrían trabajar con el sistema una vez implementado.

La Unidad de Desarrollo Informático del Área de la Energía Industria y Recursos Naturales No Renovables de la Universidad Nacional de Loja encargado de realizar la administración, planificación la parte de desarrollo de Software y la supervisión de los cumplimientos de objetivos de los mismos. Internamente cumple con las tareas de tanto en la parte educativo, profesional, en la parte de entregas de avance de los proyectos en desarrollo.

En la actualidad La Unidad de Desarrollo Informático, las actualizaciones se realizan de forma manual, provocando problemas al momento de generar nuevos cambios en los proyectos de desarrollo de software.

Tomando en cuenta la necesidad de un control y optimización de los recursos para desarrollar software, así como también de una herramienta que brinde al desarrollador un mejor control de versiones se propone la implementación y configuración del siguiente trabajo:

**“CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMATICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA “**

## **2.2 Problema General de investigación**

En la actualidad no se ha implementado una herramienta SVN, para plataformas Linux, que permita realizar el control de versiones de proyectos



en desarrollo para “La Unidad de Desarrollo Informático del A.E.I.R.N.N.R. De la Universidad Nacional de Loja”.

El cual se presentan las siguientes características:

- **Los programas de Desarrollo de software no están integrados.** No se comunican entre sí. La comunicación se da vía documentos, o la opción de copiar y pegar, lo cual implica la posibilidad creciente de cometer errores, duplicidad de la información, imposibilidad de establecer controles de versiones, inversión innecesaria de tiempo y gastos innecesarios en insumos. Además de la integración del módulo consigo mismo, es necesario que exista comunicación con otros sistemas (módulos).

Por ello, el interés de realizar este proyecto es contribuir a mejorar el rendimiento en ahorro de tiempo al momento de unir código de varias personas que se encuentren desarrollando dicho software en La Unidad de Desarrollo Informático del A.E.I.R.N.N.R. de la Universidad Nacional de Loja mediante la configuración e implementación y configuración de un repositorio SVN utilizando un servidor en Linux.

Con el nuevo diseño se pretende obtener las siguientes ventajas:

- **Escalabilidad.** El nuevo diseño podrá ser utilizado para cubrir las necesidades de la Administración de proyectos de tesis y una nueva forma de programar ya sea con la parte docente y estudiantil.
- **Bajos Costos de Mantenimiento y de operación.** El mantenimiento involucra la modificación de un sistema para reflejar cambios en el programa o proyecto, modificaciones para apresurar ciertos aspectos operacionales o modificaciones para reflejar cambios en los requerimientos de la misma.

Para contribuir a la reducción de costos y favorecer la flexibilidad en la implantación y la actualización de soluciones, el sistema permitirá un mejor aprovechamiento de los módulos.



- **Mínimos requerimientos de hardware.** Se reducirán los requerimientos de hardware ya que la aplicación funcionaría con mínimas características, como Pentium IV de 512Mg de memoria, disco de 160GB, se podrá usar el equipamiento disponible actualmente en La Unidad de Desarrollo Informático. a cargo del Ing. Patricio Villamarín.
- **Excelente documentación.** El sistema brindará documentación, manuales, formularios, e información descriptiva que detalle o de instrucciones sobre el empleo y operación del mismo. Asimismo describirá los procedimientos o pasos para el uso específico de cada uno de los procesos y las reglas de su manejo y mantenimiento.
- **Alta confiabilidad y robustez.** Se dice que un sistema tiene confiabilidad si no produce fallas costosas o peligrosas al usarse de manera razonable. A fin de prevenir estos errores, en esta implementación, se usaran métodos y técnicas que incluyen la detección de los mismos, su corrección y tolerancia.
- **Seguridad de la información.** La información, como recurso valioso de una organización, está expuesta a actos tanto intencionales como accidentales de violación de su confidencialidad, alteración, borrado y copia, por lo que el sistema brindará claves de acceso para los distintos niveles de usuario del mismo. La posibilidad de establecer claves de seguridad personales conferirá confidencialidad absoluta a la información reservada. Además, permitirá realizar copias actualizadas de la información, con el fin de garantizar la oportuna recuperación de datos y programas en caso de pérdidas o daños.
- **Integridad.** Los módulos del sistema se encontraran totalmente comunicados entre sí, permitiendo de esta forma que la información que se ingrese al mismo sea actualizada en forma inmediata en cada uno de ellos.

Esta integración evitará la duplicidad de los datos y disminuirá la posibilidad de errores.





- **Consistencia.** El sistema ofrecerá una alta consistencia de los datos basada en procesos de doble validación, en los controles propios de la aplicación.
- **Arquitectura cliente-servidor.** El diseño presenta un esquema Cliente-Servidor, el que contribuirá a una disminución de los costos de capacitación del personal, ya que favorecerá la construcción de interfaces gráficas interactivas, las cuales serán más intuitivas y fáciles de usar por el usuario final. Otra de las ventajas de esta arquitectura es que facilitará el suministro de información a los usuarios, brindando una mayor consistencia a la información de la administración al contar con un control centralizado de los elementos compartidos.

## ENUNCIADO DE LA SITUACIÓN PROBLEMÁTICA

En La Unidad de Desarrollo Informático del A.E.I.R.N.N.R. en la Universidad Nacional de Loja, trae consigo la demora en el tiempo de unir los códigos fuentes de los programas en desarrollo de software, obtención de resultados que son poco confiables y consecuentemente no contribuyen a un mejoramiento en la parte desarrolladores, en lo estudiantil, que estén orientadas a las nuevas formas de programación utilizando una herramienta de SVN.

### Problema de investigación

**En la actualidad no existe un control en los cambios de actualizaciones de los proyectos en desarrollo utilizando un almacenamiento de información de sus datos o código fuente centralizado SVN, en la Unidad de Desarrollo Informático de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.**

### 2.3 Delimitación

El proyecto se implementará en el departamento de Desarrollo Informático del A.E.I.R.N.N.R. la Universidad Nacional de Loja, dedicada a realizar desarrollo de Software, debido a la gran responsabilidad que tiene el departamento en la parte de desarrollo y control de sistemas en desarrollo,



necesita vigilar y recomendar el mejoramiento de los procesos, al momento, al no contar con un sistema para el control del mismo, está incidiendo para que se ocasione una pérdida de tiempo y recursos.

### **2.3.1 Problemas específicos de investigación**

Los problemas principales del departamento de la Unidad de desarrollo informático de la carrera de ingeniería en sistemas se enumeran a continuación:

1. Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones SVN en Linux para el desarrollo de proyectos de sistemas informáticos.
2. Actualmente el departamento de desarrollo Informático no cuenta con un repositorio de versiones de proyectos en desarrollo de los egresados de la carrera de ingeniería en sistemas.
3. Actualmente el departamento de desarrollo no cuenta con un manual de repositorio de versiones SVN para la administración de los proyectos en el departamento de desarrollo informático de la carrera de ingeniería en sistemas
4. Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones SVN cliente servidor en Linux para el seguimiento y control de proyectos mediante accesos a los clientes en forma visual
5. Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones para el seguimiento y control de proyectos mediante en el departamento de desarrollo informático de la carrera de ingeniería en sistemas
6. Actualmente el departamento de la unidad de desarrollo Informático, en los servicios de internet a los estudiantes no tienen un control de antivirus y seguridades de la red, es en el caso de la biblioteca del Área de la carrera de Ingeniería en sistemas.



### **2.3.2 Espacio**

Debido a los problemas que se presentan en los controles y seguimientos de proyectos que surgen en los diferentes proyectos en desarrollo de software en La Unidad de Desarrollo Informático de la carrera de Ingeniería en Sistemas, nuestro objeto de estudio de control de versiones (revisiones) en los proyectos de desarrollo, en ejecuciones y culminación del mismo el cual puedes ser utilizado por los estudiantes de la carrera que se encuentre en los módulos de programación.

#### **2.3.2.1 Tiempo**

El tiempo que dedicaremos para conocer y analizar el objeto de estudio es de seis meses. Como indicamos anteriormente el objeto de estudio es lo que se refiere al control y seguimiento de proyectos en subversiones, que se llevan a cabo en cada una de los programas y proyectos de desarrollos de software involucrados en la A.E.I.R.N.N.R. De la Universidad Nacional de Loja

### **2.3.3 Unidades de Observación**

Para realizar la implementación de la presente herramienta se tomará en cuenta los siguientes elementos de observación:

#### **Unidades:**

Unidad Informática (DEPARTAMENTO DE DESARROLLO INFORMATICO):

#### **Hardware y Software:**

Infraestructura de la red de Datos.

Estudio y manejo SVN.

Estudio de los módulos del Sistema Operativo Linux que permiten realizar el control de repositorio de SVN.



### **3. JUSTIFICACION**

#### **3.1 Justificación**

El progreso y desarrollo de los pueblos se logra en gran medida al aporte de las universidades ya que mantienen un gran interés por la investigación y de esta manera buscan la forma de interactuar con la sociedad.

La Universidad Nacional de Loja es una institución de desarrollo cultural, científica y social de la región sur del País, cuya misión es la de formar profesionales de excelencia, capaces de generar ciencia y tecnología y trabajo productivo a través de las diferentes áreas.

Dentro de las cuales se encuentran el Área De Energías Industrias Y Recursos Naturales No Renovables a la que pertenece la Carrera de Ingeniería En Sistemas.

La misma que ha venido contribuyendo a la formación académica de estudiantes capacitados para resolver problemas de su profesión que aquejan a la realidad social.

En la actualidad los avances tecnológicos tanto en informática como en electrónica y telecomunicaciones han dado un giro significativo en la sociedad actual motivo por la cual se considera que el Internet, son una parte fundamental de estos avances. Lo cual hace que el presente proyecto se justifique académicamente ya que como estudiantes estamos aptos para poner en práctica todos los conocimientos adquiridos a lo largo de nuestra formación académica, y poseemos la información necesaria para su desarrollo.

Este proyecto además de ayudarnos en el complemento y perfeccionamiento de nuestra formación profesional, es un aporte significativo a la sociedad, en este caso en el departamento de RRHH del área de la salud humana, será un gran aporte para el desarrollo del mismo.

El propósito de este proyecto es ofrecer un servicio a la carrera de ingeniería en sistemas de A.E.I.R.N.N.R de Universidad Nacional de Loja,



en el cual he decidido realizar un proyecto que aplique la implementación y configuración de un servidor de repositorios SVN en Linux con sus respectivos accesos a cada usuario.

La carrera de Ingeniería en Sistemas forma profesionales con conocimientos profundos de la estructura y particularidades del software, hardware, y para plasmar las enseñanzas vertidas durante los 5 años y 6 meses de estudio, me propongo poner en la práctica los conocimientos impartidos, y planteo la ejecución de un proyecto que simplifique los procesos que se necesitan para realizar el control de versiones mediante repositorio SVN en Linux.

### **Justificación Académica**

Los estudiantes de la escuela de ingeniería en sistemas a través del Área de Energía e Industrias Recursos Naturales No Renovables de la Universidad Nacional de Loja y gracias a su sistema de Enseñanza – Aprendizaje, (SAMOT) estamos en la capacidad de poner en práctica los conocimientos adquiridos durante el periodo de formación académica superior. Dichos conocimientos servirán de base para desarrollar nuestra aplicación propuesta.

### **Justificación Técnica**

Actualmente, la irrupción de las nuevas tecnologías en el mundo de la informática en la parte de desarrollo de software tanto en parte empresarial, educativo, gubernamental, ha puesto de manifiesto nuevas necesidades que deben resolver con rapidez y eficiencia, en una sociedad en la que la información se ha convertido en un importante valor económico, la más importante de estas nuevas necesidades es sin duda una gestión global y eficiente de los contenidos que permita controlar en todo momento la información suministrada a los docentes administrativos en nuestro caso. Gracias a su arquitectura basada en estándares abiertos. Esta funcionalidad hace posible no sólo disminuir el tiempo de desarrollo de la aplicación, sino también reducir los costes ligados a la adquisición de licencias.



Para la realización de la implementación del sistema de control se utilizara software libre.

### **Justificación Operativa**

Una vez realizada las investigaciones pertinentes, en el departamento de desarrollo informático del A.R.N.N.R. se han visto “La falta de un Software que permita controlar las actualizaciones versiones de software con el fin de ahorrar tiempo al momento unir los códigos fuentes.

Cabe mencionar que a la culminación del proyecto se realizarán las pruebas respectivas del funcionamiento de la aplicación y existe el compromiso por mi parte. Quede funcionando para el bien el área.

### **Justificación Económica**

De acuerdo a investigaciones realizadas se debe disponer de algunos elementos como: hardware, software, redes etc. De la misma el proyecto se justifica económicamente debido a que se cuenta con los recursos tanto económicos como tecnológicos (software libre) suficientes para cubrir todas las necesidades que se puedan presentar durante su implementación, tomando en cuenta el costo/beneficio y además contando con el apoyo del departamento, en lo que respecta al Hardware (donde se va a implementar el software).

### **3.2 Viabilidad**

El proyecto es viable, ya que las herramientas a utilizar son libres, y se tiene el apoyo de los encargados del departamento de Desarrollo Informático (UDI).



## **4. OBJETIVOS**

### **4.1 Objetivo general**

**“Configurar e implementar un servidor en Linux de repositorios de versiones utilizando SVN, para el control de versiones de proyectos en La Unidad de Desarrollo Informático del A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA.”**

### **4.2 Objetivos específicos**

- Instalación y configuración del servicio subversión SVN, con sus dependencias.
- Instalación y configuración del servicio subversión SVN visual, para clientes.
- Construir los repositorios SVN de los proyectos desarrollados y en desarrollo por los egresado de la carrera con sus respectivos controles de accesos.
- Desarrollar un manual de administrador para el control de versiones.
- Aplicar el plan de validación de las subversiones en la carrera de Ingeniería en sistemas.



## **5. MARCO TEORICO**

### **1.1 Qué es Control de Versiones**

### **1.2 Sistema de Control de Versiones**

#### **1.2.1 *Su propósito***

#### **1.2.2 *Es útil para***

### **1.3 Repositorio**

### **1.4 Introducción a subversión.**

#### **1.4.1 Historia de Subversión**

### **1.5 Qué es “Subversión**

### **1.6 Conceptos de Subversiones.**

### **1.7 Subversión. Características.**

### **1.8 Subversión proporciona.**

### **1.9 Arquitectura de Subversión**

## **2. 1 Linux**

### **2.1.1 ¿Qué es Linux?**

### **2.1.2 Historia**

### **2.1.3 Aplicaciones**

### **2.1.4 Características**

### **2.1.5 Distribuciones de Linux**

## **3.1 Definición Arquitectura Cliente Servidor**

### **3.1.1 Introducción**

### **3.1.2 Elementos principales**

### **3.1.3 Algunos antecedentes, ¿porque fue creado?**

### **3.1.4 Evolución de la arquitectura cliente servidor**

#### **3.1.4.1 La era de la computadora central**

#### **3.1.4.2 La era de las computadoras**

#### **3.1.4.3 La era de la conexión libre**

#### **3.1.4.4 La era del cómputo a través de la redes**

### **3.1.5 Que es una arquitectura**

### **3.1.6 Que es un cliente**

### **3.1.7 Que es un servidor**

### **3.1.8 Elementos de la arquitectura de un servidor**

### **3.1.9 Características del modelo cliente servidor**





#### **3.1.9.1 Tipos de clientes**

#### **3.1.9.2 Tipos de Servidor**

### **3.1.10 Estilo de modelo cliente servidor**

#### **3.1.10.1 Ventajas**

#### **3.1.10.2 Desventajas**

### **3.1.11 Lógica distribuida**

#### **3.1.11.1 Ventaja**

#### **3.1.11.2 Desventaja**

### **3.1.12 Funciones de un cliente servidor**



## 5- MARCO TEÓRICO

### 1.1 Qué es Control de Versiones

<sup>13</sup>Subversión es uno de los sistemas de control de versiones más modernos y utiliza un sistema con repositorio centralizado y fue diseñado como remplazo del sistema más utilizado hasta la fecha

<sup>1</sup> El control de versiones se basa en una serie de acciones más o menos estándar de comunicación entre la copia de trabajo y el repositorio. Estas acciones son precisamente las que permite el cliente Subversión. No se entrarán en detalles de cómo administrar un repositorio y cómo se resuelven conflictos entre el repositorio y la copia de trabajo.

#### 1.1.1 Sistema de Control de Versiones

<sup>14</sup>Herramienta de software que administra el acceso a un conjunto de archivos que componen un proyecto, y mantiene un historial de cambios realizados sobre los mismos.

##### 1.1.1.1 Su propósito

Registrar información del usuario que realizó algún cambio, cuando, cual es la diferencia con relación a la versión anterior del mismo archivo

##### 1.1.1.2 Es útil para

Guardar cualquier documento que cambie con frecuencia, *Existe un único repositorio central del cual cada usuario sincroniza su copia local, lo modifican y luego registran nuevamente sus versiones actualizadas de manera a que todos los usuarios posean las mismas versiones del archivo modificado, permitiendo volver a un estado anterior en caso de que sea necesario.*

#### 1.1.2 Repositorio

Subversión es un sistema centralizado para compartir información. La parte principal de Subversión es el repositorio, el cual es un almacén central de

---

<sup>13</sup> Juan Luis Serradilla ([juanlu@um.es](mailto:juanlu@um.es)) Sección de Metodología, Normalización y Calidad del Software

<sup>14</sup>Sistema de Control de VersionesCaso de estudio: Subversión



datos. El repositorio guarda información en forma de *árbol de archivos*, una típica jerarquía de archivos y directorios. Cualquier <sup>1</sup>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. Entonces, ¿qué tiene esto de interesante? Hasta ahora, suena como la definición del típico servidor de archivos. Y, de hecho, el repositorio es una especie de servidor de archivos, pero no del tipo habitual. Lo que hace especial al repositorio de Subversión 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 última versión del árbol de archivos. Sin embargo, el cliente también tiene la posibilidad de ver estados *previos* del sistema de archivos. Por ejemplo, un cliente puede hacer consultas históricas como, “¿Qué contenía este directorio el miércoles pasado?” Esta es la clase de preguntas que resulta esencial en cualquier *sistema de control de versiones*: sistemas que están diseñados para registrar y seguir los cambios en los datos a través del tiempo.

## 1.2 Introducción a subversión.

<sup>1</sup> Subversión, CVS y otros sistemas de control de versiones utilizan un modelo conocido como copiar-modificar-intercalar. En este modelo, cada cliente de usuario contacta al repositorio central y crea una copia de trabajo, que es un reflejo de los archivos y directorios en el repositorio central. Los usuarios trabajan en paralelo, modificando sus copias privadas y, finalmente, intercalan sus copias en una versión final nueva. El sistema de control usualmente asiste el proceso de intercalado, pero al final del día es un humano el encargado de que esto ocurra correctamente.

Existen otros modelos más sencillos en el papel, pero con grandes desventajas a largo plazo.



El modelo de copiar-modificar-intercalar puede sonar un tanto caótico, pero en la práctica funciona muy bien. Incrementa la productividad, ya que los usuarios pueden trabajar en paralelo, sin esperar a los otros. Cuando trabajan en los mismos archivos, en general, los cambios no se enciman y los conflictos son poco frecuentes y toma poco tiempo arreglarlos.

#### 1.2.1.1 Historia de Subversión

<sup>15</sup>A principios del 2000, CollabNet, Inc. (<http://www.collab.net>) comenzó a buscar desarrolladores para escribir un sustituto para CVS. CollabNet ofrece un conjunto de herramientas de software colaborativo llamado SourceCast, del cual un componente es el control de versiones. Aunque SourceCast usaba CVS como su sistema de control de versiones inicial, las limitaciones de CVS se hicieron evidentes desde el principio, y CollabNet sabía que tendría que encontrar algo mejor. Desafortunadamente, CVS se había convertido en el estándar *de facto* en el mundo del código abierto porque *no había* nada mejor, al menos no bajo una licencia libre. Así CollabNet decidió escribir un nuevo sistema de control de versiones desde cero, manteniendo las ideas básicas de CVS, pero sin sus fallos y defectos.

En febrero del 2000, contactaron con Karl Fogel, autor de *Open Source Development with CVS* (Coriolis, 1999), y le preguntaron si le gustaría trabajar en este nuevo proyecto. Casualmente, por aquel entonces Karl ya se encontraba discutiendo sobre el diseño de un nuevo sistema de control de versiones con su amigo JimBlandy. En 1995, los dos habían fundado Cyclic Software, compañía que hacía contratos de soporte de CVS, y aunque después vendieron el negocio, seguían usando CVS todos los días en sus trabajos. La frustración de ambos con CVS había conducido a Jim a pensar cuidadosamente acerca de mejores vías para administrar datos versionados, y no sólo tenía ya el nombre de “Subversion”, sino

---

<sup>15</sup>Historia de Subversion Control de versiones con Subversion <http://svnbook.red-bean.com/nightly/es/>



también el diseño básico del repositorio de Subversion. Cuando CollabNet llamó, Karl aceptó inmediatamente trabajar en el proyecto, y Jim consiguió que su empresa, RedHat Software, básicamente lo donara al proyecto por un período de tiempo indefinido. Collabnet contrató a Karl y a Ben Collins-Sussman, y el trabajo detallado de diseño comenzó en mayo. Con la ayuda de algunos ajustes bien colocados de Brian Behlendorf y Jason Robbins de CollabNet, y Greg Stein (por aquel entonces un activo desarrollador independiente del proceso de especificación de WebDAV/DeltaV), Subversión atrajo rápidamente a una comunidad activa de desarrolladores. Esto vino a demostrar que era mucha la gente que había tenido las mismas frustrantes experiencias con CVS, y que había recibido con agrado la oportunidad de hacer algo al respecto.

El equipo de diseño original estableció algunos objetivos simples. No querían abrir nuevos caminos en la metodología del control de versiones, sólo querían corregir CVS. Decidieron que Subversión incorporaría las características de CVS, y que preservarían el mismo modelo de desarrollo, pero sin duplicar los defectos obvios de CVS. Y aunque no necesitaba ser un reemplazo exacto de CVS, debía ser lo bastante similar para que cualquier usuario de CVS pudiera hacer el cambio con poco esfuerzo.

Después de catorce meses de codificación, Subversión pasó a ser “auto-hospedado” el 31 de agosto del 2001. Es decir, los desarrolladores de Subversión dejaron de usar CVS para la administración del propio código fuente de Subversión, y en su lugar empezaron a usar Subversión.

Si bien fue CollabNet quien inició el proyecto, y todavía financia una gran parte del trabajo (paga el salario de unos pocos desarrolladores a tiempo completo de Subversión), Subversión funciona como la mayoría de proyectos de código abierto, dirigido por un conjunto informal de reglas transparentes que fomentan el mérito. La licencia copyright de CollabNet es completamente compatible con las Directrices de Software Libre de Debian. En otras palabras, cualquier persona es libre de descargar, modificar, y redistribuir Subversión



### 1.2.2 Qué es “Subversión”

- Herramienta de código abierto, multiplataforma (Win32, Linux), para el control de versiones de ficheros electrónicos, como son el software o la documentación.
- Se basa en un repositorio central que actúa como un servidor de ficheros, con la capacidad de recordar todos los cambios que se hacen tanto en sus directorios como en sus ficheros.
- El repositorio incrementa un número global de revisión con cada conjunto de cambios enviados (commit) al mismo. Es posible copiar y renombrar ficheros.

### 1.3 Conceptos de Subversiones.

- Servidor: Mantiene el código oficial (repositorio)
- Cliente: Mantiene una copia de trabajo o borrador
- Módulo: Conjunto de archivos y directorios
- Repositorio: Contiene los módulos
- Usuario: Persona autorizada a trabajar con un módulo
- Tronco: Directorio donde se almacenan todos los archivos.
- Ramas: Línea de desarrollo que existe de forma independiente a otra, pero que comparte un historial común si se mira lo suficientemente atrás en el tiempo.

### 1.4 Subversión. Características.

- *Alta disponibilidad de clientes* Existen versiones para todos los sistemas operativos tanto de consola como gráficos.
- *Propiedades*: la meta información arbitraria o con significado especial que se asocia a un archivo o directorio. A través de estas propiedades subversion maneja algunas cosas
  - interesantes, como el tipo de fin de línea, si es un archivo binario y de qué tipo (para utilizar la herramienta de diffcorrecta), si es ejecutable, etc.



- El creador de ramas y etiquetas es una *operación eficiente*  $O(1)$
- Se envían *sólo las diferencias en ambas direcciones*.
- *Maneja eficientemente archivos binarios*.

### 1.5 Subversión proporcional.

- **Verdadero historial de versiones:**

- Subversión implementa sistema de ficheros versionado “virtual” sigue los cambios sobre árboles de directorios. Ficheros y directorios, se encuentran bajo el control de versiones.
- Con Subversión, se puede añadir, borrar, copiar, y renombrar ficheros y directorios. Cada
- fichero nuevo comienza con un historial nuevo, completamente suyo.

- **Envíos atómicos:**

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

- **Versionado de metadatos:**

- Cada fichero y directorio tiene un conjunto de propiedades claves y sus valores asociado a él. Se puede crear y almacenar cualquier par 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:**

- Subversión tiene una noción abstracta del acceso al repositorio, lo que facilita implementar nuevos mecanismos de red.
- Puede conectarse al servidor HTTP Apache como un módulo de extensión.
- También tiene disponible un servidor de Subversión independiente, y más ligero, que habla un protocolo propio, el cual puede ser encaminado



fácilmente a través de un túnelSSH.

### **Manipulación consistente de datos:**

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

### **Ramificación y etiquetado eficientes:**

El coste de ramificación y etiquetado no necesita ser proporcional al tamaño del proyecto. Subversión crea ramas y etiquetas simplemente copiando el proyecto, usando un mecanismo similar al enlace duro.

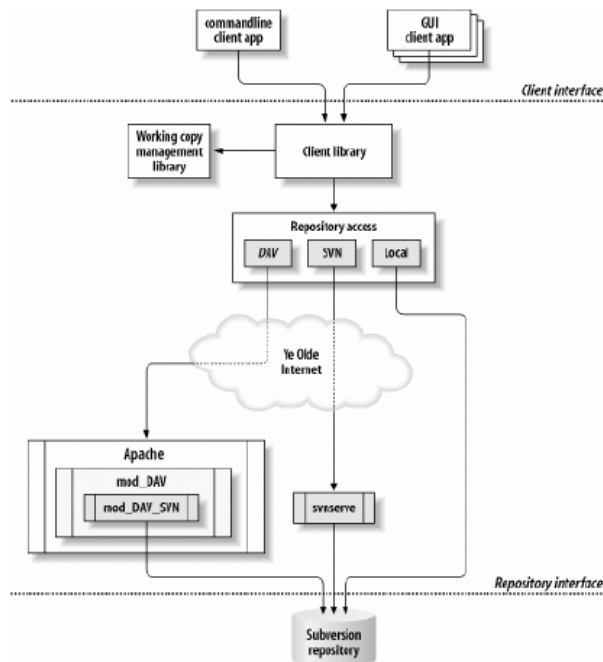
Estas operaciones toman solamente una cantidad de tiempo pequeña y constante.

### **Hackability:**

Subversión no tiene un equipaje histórico; está implementado como una colección de bibliotecas compartidas en C con APIs bien definidas. *Esto lo hace fácil de mantener y reutilizable por otras aplicaciones y lenguajes* figura(5.1.6.1)



## 1.6 Arquitectura de Subversión



**Figura (5.1.6.1)** Sistema de Control de Versiones Caso de estudio: Subversión

Subversión usa el modelo Copy-Modify-Merge y trata siempre de optimizar el uso de ancho de banda haciendo offline todas las operaciones posibles. Para ello guarda una copia intacta del estado del repositorio, y otra para que modifiquemos. Así, algunas operaciones que pueden hacerse offline. Entre ellas, la más importante es **diff**, en la que se envía al servidor solamente el **changeset** (y no archivos modificados enteros y hacer el diff en el servidor).

El uso de ancho de banda en subversión al aplicar un **changeset** es proporcional al tamaño de los cambios y no de los archivos enteros, como pasa en otros SCMs.

*Subversión, tiene un filesystem versionado*, esto quiere decir que es un sistema de archivo (conjunto organizado jerárquicamente de directorios y archivos, y sus contenidos, por supuesto).

Por eso provee las operaciones comunes de cualquier filesystem (crear un directorio, copiar un archivo, mover un archivo, borrar un archivo, etc), con algunas particularidades.



Subversión piensa en los repositorios como una línea de tiempo, al hacer una copia de un archivo A a B, en realidad no se copia el archivo sino que se dice que el archivo copiado B es una bifurcación en la línea de tiempo del archivo A. Es decir:

A esta característica se la llama "**cheapcopy**" (copia barata) y es la manera en la que subversión implementa los branches.

Suponiendo que en vez de ser archivos son directorios, lo que tenemos son 2 evoluciones distintas de la misma base de código.

En un determinado momento podemos aplicar los changesets de un branch en el otro a través de un merge.

### 1.7 Subversión trabaja replicando el modelo cliente/servidor: <sup>16</sup>

Uno o más clientes se conectan a un servidor central que tiene la última copia del proyecto, como también copias de sus versiones anteriores, dentro de lo que llamaremos un **repositorio**.

El cliente pide la última versión disponible del proyecto para poder trabajar. Los clientes hacen cambios y, por último, sincronizan sus copias con el repositorio.

Como se puede observar, su uso en equipos de trabajo está perfectamente justificado. De cualquier manera, subversión también es útil como una herramienta de backup incremental unipersonal. Imaginemos SVN como un sistema de históricos con capturas en el código fuente automáticas. Cada vez que enviemos

un archivo modificado al repositorio, éste creará una copia de la versión anterior. Ninguna de las copias anteriores se perderá, por lo que siempre

---

<sup>16</sup> webmasterscoordinando versiones de código fácilmente Rodrigo Galíndez .



podremos volver a una versión anterior de nuestros archivos si es necesario.

Es más, si estamos trabajando con un servidor de Subversion ubicado en algún lugar de Internet, digamos, un servidor alojado en los Estados Unidos, tenemos una buena posibilidad de no perder los archivos originales por un desperfecto en nuestras computadoras.

## **2.1 Linux**

### **2.1.1 ¿Qué es Linux?**

Linux, que se pronuncia de la misma manera en todos los idiomas del mundo, como el idioma Español, y *no* 'lay-nux', es un sistema operativo (así como lo es Windows, Solaris, Mac OS X); fue creado por LinusTorvalds en 1991 como una alternativa a los sistemas Unix de la época.

Acostumbrado a trabajar con sistemas operativos Unix, e insatisfecho con los ofrecimientos gratuitos por sus limitaciones y los comerciales por sus elevados precios, decidió entonces iniciar el proyecto que hoy se ha convertido en una auténtica alternativa a otros Sistemas Operativos.

### **2.1.2 Historia**

Al conocer de la existencia del Proyecto GNU, un grupo de desarrolladores de software que promueven la libre distribución de los programas fuente, para que los usuarios mismos contribuyan con sus conocimientos para el mejoramiento total del software, Linux decidió también adherirse a los lineamientos de esta organización, y publicar libremente los programas fuente de su Sistema Operativo. Inicialmente sólo contaba con las funciones más básicas, pero con la ayuda de colaboradores alrededor del mundo fue cobrando fuerza hasta convertirse en lo que es hoy en día: un Sistema Operativo basado en Unix con todas las características que se exigen en los sistemas comerciales.

Linus atrajo muchos desarrolladores desde un principio el concepto del 'Open SourceMovement' (Movimiento para la Liberación de los Programas Fuente) ha sido uno de los mayores atractivos. La libertad de desarrollo, la



proliferación de nuevas ideas y el mejoramiento de conceptos “clásicos” fue liderizado por entusiastas de distintas áreas aquellos con experiencia en redes contribuyeron con las secciones que se encargan de protocolos e implementación, mientras que otros con experiencia en manejo de memoria contribuyeron con las secciones de Linux que se encargan de esta tarea. Torvalds y otros desarrolladores de los primeros días de Linux adaptaron los componentes de GNU y de BSD, así como de otros muchos proyectos como Perl, Apache, Python, etc. para trabajar con el núcleo Linux, creando un sistema operativo completamente funcional procedente de muchísimas fuentes diferentes, la mayoría libres.

LinusTorvalds tiene generalmente la última palabra de qué se incorpora en cada nueva versión pero existen distribuidores que le incorporan distintas aplicaciones y herramientas a un precio razonable, junto con detallados libros y herramientas de instalación. Torvalds ha continuado publicando nuevas versiones del núcleo, consolidando aportes de otros programadores y haciendo cambios por su cuenta. Todas las versiones de Linux que tienen el número de sub-versión (el segundo número) par, pertenecen a la serie "estable", por ejemplo: 1.0.x, 1.2.x, 2.0.x, 2.2.x, 2.4.x y actualmente 2.6.x, mientras que las versiones con sub versión impar, como la serie 2.5.x, son versiones de desarrollo, es decir que no son consideradas de producción.

Mientras que Torvalds continúa publicando las últimas versiones de desarrollo, el mantenimiento de las rama "estables", siempre algo más viejas, ha sido delegada a otros programadores, incluyendo a David Weinehall (2.0), Alan Cox (2.2), Marcelo Tosatti (2.4) y Andrew Morton (2.6). Además de estas versiones "oficiales" del núcleo, es posible obtener versiones "alternativas" en otras fuentes. Los encargados de las Distribuciones de Linux, normalmente mantienen sus propias versiones del núcleo, que a veces incluyen por ejemplo, controladores que no han sido incorporados a la versión oficial.



### 2.1.3 Aplicaciones

Linux es un Sistema Operativo diseñado desde un principio para facilitar su uso y administración, especialmente remota. Incluso en ambientes gráficos, se puede utilizar un servidor Linux para ejecutar poderosas aplicaciones en terminales baratas (incluso equipos que ya pasaron hace mucho su “vida útil”, como por ejemplo Intel 486, Pentium I,II etc..).

Debido a su eficiente aprovechamiento de recursos, GNU/Linux tiene requisitos de hardware mínimos muy bajos: Una configuración mínima puede ser una 386 SX/16 con 1MB de RAM, y una diskettera (más teclado, placa de vídeo, monitor, etc.). Esto es suficiente para arrancar y entrar al sistema. Los bajos requisitos de hardware permiten hacer un sistema potente y útil de aquel 486 que algunos guardan en un armario. Esta misma característica permite aprovechar al máximo las capacidades de las computadoras más modernas.

#### 2.1.3.1 Características

<sup>17</sup>*Multitarea*: La palabra multitarea describe la habilidad de ejecutar varios programas al mismo tiempo. LINUX utiliza la llamada *multitarea preventiva*, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa.

*Multiusuario*: Muchos usuarios usando la misma máquina al mismo tiempo.

*Multiplataforma*: Las plataformas en las que en un principio se puede utilizar Linux son 386-, 486-. Pentium, Pentium Pro, Pentium II, Amiga y Atari, también existen versiones para su utilización en otras plataformas, como amd64, Alpha, ARM, MIPS, PowerPC y SPARC.

*Multiprocesador*: Soporte para sistemas con más de un procesador está disponible para Intel, AMD y SPARC.

Funciona en *modo protegido 386*.

*Protección de la memoria* entre procesos, de manera que uno de ellos no pueda colgar el sistema.

---

<sup>17</sup> [Fuente: Infosheet-Como. Autor: IvanCasado]



*Carga de ejecutables por demanda:* Linux sólo lee del disco aquellas partes de un programa que están siendo usadas actualmente.

Política de copia en escritura para la compartición de páginas entre ejecutables: esto significa que varios procesos pueden usar la misma zona de memoria para ejecutarse. Cuando alguno intenta escribir en esa memoria, la página (4Kb de memoria) se copia a otro lugar. Esta política de copia en escritura tiene dos beneficios: aumenta la velocidad y reduce el uso de memoria.

*Memoria virtual usando paginación* (sin intercambio de procesos completos) a disco: A una partición en el sistema de archivos, con la posibilidad de añadir más áreas de intercambio sobre la marcha.

La memoria se gestiona como un *recurso unificado* para los programas de usuario y para el caché de disco, de tal forma que toda la memoria libre puede ser usada para caché y ésta puede a su vez ser reducida cuando se ejecuten grandes programas.

*Librerías compartidas* de carga dinámica (DLL's) y *librerías estáticas*.

Se realizan *volcados de estado* (coredumps) para posibilitar los análisis post-mortem, permitiendo el uso de depuradores sobre los programas no sólo en ejecución sino también tras abortar éstos por cualquier motivo.

Compatible con *POSIX*, System V y BSD a nivel fuente.

Emulación de *iBCS2*, casi completamente compatible con SCO, SVR3 y SVR4 a nivel binario.

Todo el *código fuente* está *disponible*, incluyendo el núcleo completo y todos los drivers, las herramientas de desarrollo y todos los programas de usuario; además todo ello se puede distribuir libremente. Hay algunos programas comerciales que están siendo ofrecidos para Linux actualmente sin código fuente, pero todo lo que ha sido gratuito sigue siendo gratuito.



*Pseudo-terminales (pty's).*

*Emulación de 387* en el núcleo, de tal forma que los programas no tengan que hacer su propia emulación matemática. Cualquier máquina que ejecute Linux parecerá dotada de coprocesador matemático. Por supuesto, si el ordenador ya tiene una FPU (unidad de coma flotante), esta será usada en lugar de la emulación, pudiendo incluso compilar tu propio kernel sin la emulación matemática y conseguir un pequeño ahorro de memoria.

Soporte para muchos teclados nacionales o adaptados y es bastante fácil añadir nuevos dinámicamente.

*Consolas virtuales múltiples:* varias sesiones de login a través de la consola entre las que se puede cambiar con las combinaciones adecuadas de teclas (totalmente independiente del hardware de video). Se crean dinámicamente y puedes tener hasta 64.

Soporte para *varios sistemas de archivo* comunes, incluyendo minix-1, Xenix y todos los sistemas de archivo típicos de System V, y tiene un avanzado sistema de archivos propio con una capacidad de hasta 4 Tb y nombres de archivos de hasta 255 caracteres de longitud.

Acceso transparente a particiones MS-DOS (o a particiones OS/2 FAT) mediante un sistema de archivos especial: no es necesario ningún comando especial para usar la partición MS-DOS, esta parece un sistema de archivos normal de Unix (excepto por algunas restricciones en los nombres de archivo, permisos, y esas cosas). Las particiones comprimidas de MS-DOS 6 no son accesibles en este momento, y no se espera que lo sean en el futuro. El soporte para VFAT, FAT32 (WNT, Windows 95/98) se encuentra soportado desde la version 2.0 del nucleo y el NTFS de WNT desde la version 2.2 (Este último solo en modo lectura).

Soporte en sólo lectura de HPFS-2 del OS/2 2.1

Sistema de archivos de CD-ROM que lee todos los formatos estándar de CD-ROM.

*TCP/IP*, incluyendo ssh, ftp, telnet, NFS, etc.

*Appletalk.*

Software cliente y servidor *Netware*.

*Lan Manager / Windows Native (SMB)*, software cliente y servidor.



Diversos *protocolos de red* incluidos en el kernel: TCP, IPv4, IPv6, AX.25, X.25, IPX, DDP, Netrom, etc

#### **2.1.4 Distribuciones de Linux**

Linux es un sistema de libre distribución por lo que se puede encontrar todos los ficheros y programas necesarios para su funcionamiento en multitud de servidores conectados a Internet. La tarea de reunir todos los ficheros y programas necesarios, así como instalarlos en tu sistema y configurarlo, puede ser una tarea bastante complicada y no apta para muchos. Por esto mismo, nacieron las llamadas distribuciones de Linux, empresas y organizaciones que se dedican a hacer el trabajo "sucio" para nuestro beneficio y comodidad.

Una distribución no es otra cosa, que una recopilación de programas y ficheros, organizados y preparados para su instalación. Estas distribuciones se pueden obtener a través de Internet, o comprando los CDs de las mismas, los cuales contendrán todo lo necesario para instalar un sistema Linux bastante completo y en la mayoría de los casos un programa de instalación que nos ayudara en la tarea de una primera instalación. Casi todos los principales distribuidores de Linux, ofrecen la posibilidad de bajarse sus distribuciones, vía FTP (sin cargo alguno).

Existen muchas y variadas distribuciones creadas por diferentes empresas y organizaciones a unos precios bastantes asequibles (si se compran los CDs, en vez de bajársela via FTP)

A continuación algunas de las distribuciones más importantes de Linux (aunque no las únicas).

#### **UBUNTU**

Distribución basada en Debian, con lo que esto conlleva y centrada en el usuario final y facilidad de uso. Muy popular y con mucho soporte en la comunidad. El entorno de escritorio por defecto es GNOME.





## **REDHAT ENTERPRISE**

Esta es una distribución que tiene muy buena calidad, contenidos y soporte a los usuarios por parte de la empresa que la distribuye. Es necesario el pago de una licencia de soporte. Enfocada a empresas.

## **FEDORA**

Esta es una distribución patrocinada por RedHat y soportada por la comunidad. Fácil de instalar y buena calidad.

## **DEBIAN**

Otra distribución con muy buena calidad. El proceso de instalación es quizás un poco más complicado, pero sin mayores problemas. Gran estabilidad antes que últimos avances.

## **OpenSuSE**

Otra de las grandes. Fácil de instalar. Versión libre de la distribución comercial SuSE.

## **SuSE LINUX ENTERPRISE**

Otra de las grandes. Muy buena calidad, contenidos y soporte a los usuarios por parte de la empresa que la distribuye, Novell. Es necesario el pago de una licencia de soporte. Enfocada a empresas.

## **SLACKWARE**

Esta distribución es de las primeras que existió. Tuvo un periodo en el cual no se actualizó muy a menudo, pero eso es historia. Es raro encontrar usuarios de los que empezaron en el mundo linux hace tiempo, que no hayan tenido esta distribución instalada en su ordenador en algún momento.



## **GENTOO**

Esta distribución es una de las únicas que incorporaron un concepto totalmente nuevo en Linux. Es una sistema inspirado en BSD-ports. Se puede compilar/optimizar vuestro sistema completamente desde cero. No es recomendable adentrarse en esta distribución sin una buena conexión a internet, un ordenador medianamente potente (si quiere terminar de compilar en un tiempo prudencial) y cierta experiencia en sistemas Unix.

## **KUBUNTU**

Distribución basada en Ubuntu, con lo que esto conlleva y centrada en el usuario final y facilidad de uso. La gran diferencia con Ubuntu es que el entorno de escritorio por defecto es KDE.

## **MANDRIVA**

Esta distribución fue creada en 1998 con el objetivo de acercar el uso de Linux a todos los usuarios, en un principio se llamó Mandrake Linux. Facilidad de uso para todos los usuarios.

### **3.1 Definición Arquitectura Cliente Servidor**

#### **3.1.1 Introducción**

En vista del aprendizaje que tenemos diariamente en el aula de clases, nos vemos desafiados por un mundo lleno de conocimientos que invoca a la investigación.

Este trabajo fue realizado precisamente para llenar las expectativas y ansias de intelectualidad que nos brinda la carrera, desde bases de datos, vemos la importancia de la arquitectura cliente servidor.

Es exactamente lo que se plasmara en el siguiente trabajo, la forma de conocer una arquitectura que en este momento es una de las más importantes y utilizadas en el ámbito de enviar y recibir información, también



es una herramienta potente para guardar los datos en una base de datos como servidor.

Con respecto a la definición de arquitectura cliente/servidor se encuentran las siguientes definiciones:

Cualquier combinación de sistemas que pueden colaborar entre si para dar a los usuarios toda la información que ellos necesiten sin que tengan que saber dónde está ubicada.

Es una arquitectura de procesamiento cooperativo donde uno de los componentes pide servicios a otro.

Es un procesamiento de datos de índole colaborativo entre dos o más computadoras conectadas a una red.

El término cliente/servidor es originalmente aplicado a la arquitectura de software que describe el procesamiento entre dos o más programas: una aplicación y un servicio soportante.

IBM define al modelo Cliente/Servidor. "Es la tecnología que proporciona al usuario final el acceso transparente a las aplicaciones, datos, servicios de cómputo o cualquier otro recurso del grupo de trabajo y/o, a través de la organización, en múltiples plataformas. El modelo soporta un medio ambiente distribuido en el cual los requerimientos de servicio hechos por estaciones de trabajo inteligentes o "clientes", resultan en un trabajo realizado por otros computadores llamados servidores".

"Es un modelo para construir sistemas de información, que se sustenta en la idea de repartir el tratamiento de la información y los datos por todo el sistema informático, permitiendo mejorar el rendimiento del sistema global de información"

### **3.3.1 Elementos principales**

"Los elementos principales de la arquitectura cliente servidor son justamente el elemento llamado cliente y el otro elemento llamado servidor".

Por ejemplo dentro de un ambiente multimedia, el elemento cliente



sería el dispositivo que puede observar el vídeo, cuadros y texto, o reproduce el audio distribuido por el elemento servidor.

Por otro lado el cliente también puede ser una computadora personal o una televisión inteligente que posea la capacidad de entender datos digitales. Dentro de este caso el elemento servidor es el depositario del vídeo digital, audio, fotografías digitales y texto y los distribuye bajo demanda de ser una máquina que cuenta con la capacidad de almacenar los datos y ejecutar todo el software que brinda éstos al cliente:

C/S es una relación entre procesos corriendo en máquinas separadas

El servidor (S) es un proveedor de servicios.

El cliente (C) es un consumidor de servicios.

C y S Interactúan por un mecanismo de pasaje de mensajes:

Pedido de servicio.

Respuesta

### **3.1.3 Algunos antecedentes, ¿por qué fue creado?**

Existen diversos puntos de vista sobre la manera en que debería efectuarse el procesamiento de datos, aunque la mayoría que opina, coincide en que nos encontramos en medio de un proceso de evolución que se prolongará todavía por algunos años y que cambiará la forma en que obtenemos y utilizamos la información almacenada electrónicamente.

El principal motivo detrás de esta evolución es la necesidad que tienen las organizaciones (empresas o instituciones públicas o privadas), de realizar sus operaciones más ágil y eficientemente, debido a la creciente presión competitiva a la que están sometidas, lo cual se traduce en la necesidad de que su personal sea más productivo, que se reduzcan los costos y gastos de operación, al mismo tiempo que se generan productos y servicios más rápidamente y con mejor calidad.

En este contexto, es necesario establecer una infraestructura de procesamiento de información, que cuente con los elementos requeridos para proveer información adecuada, exacta y oportuna en la toma de decisiones y para proporcionar un mejor servicio a los clientes.



El modelo Cliente/Servidor reúne las características necesarias para proveer esta infraestructura, independientemente del tamaño y complejidad de las operaciones de las organizaciones públicas o privadas y, consecuentemente desempeña un papel importante en este proceso de evolución.

### **3.1.4 Evolución de la arquitectura cliente servidor**

#### **3.1.4.1 La era de la computadora central**

“Desde sus inicios el modelo de administración de datos a través de computadoras se basaba en el uso de terminales remotas, que se conectaban de manera directa a una computadora central”. Dicha computadora central se encargaba de prestar servicios caracterizados por que cada servicio se prestaba solo a un grupo exclusivo de usuarios.

#### **3.1.4.2 La era de las computadoras**

Esta es la era en la que cada servicio empleaba su propia computadora que permitía que los usuarios de ese servicio se conectaran directamente. Esto es consecuencia de la aparición de computadoras pequeñas, de fácil uso, más baratas y más poderosas de las convencionales.

#### **3.1.4.3 La era de la conexión libre**

Hace más de 10 años que la computadoras escritorio aparecieron de manera masiva. Esto permitió que parte apreciable de la carga de trabajo de cómputo tanto en el ámbito de cálculo como en el ámbito de la presentación se lleven a cabo desde el escritorio del usuario. En muchos de los casos el usuario obtiene la información que necesita de alguna computadora de servicio. Estas computadoras de escritorio se conectan a las computadoras de servicio empleando software que permite la emulación de algún tipo de terminal. En otros de los casos se les transfiere la información haciendo uso de recursos magnéticos o por transcripción.



#### **3.1.4.4 La era del cómputo a través de la redes**

Esta es la era que está basada en el concepto de redes de computadoras, en la que la información reside en una o varias computadoras, los usuarios de esta información hacen uso de computadoras para laborar y todas ellas se encuentran conectadas entre sí. Esto brinda la posibilidad de que todos los usuarios puedan acceder a la información de todas las computadoras y a la vez que los diversos sistemas intercambien información.

La era de la arquitectura cliente servidor

“En esta arquitectura la computadora de cada uno de los usuarios, llamada cliente, produce una demanda de información a cualquiera de las computadoras que proporcionan información, conocidas como servidores, estos últimos responden a la demanda del cliente que la produjo.

Los clientes y los servidores pueden estar conectados a una red local o una red amplia, como la que se puede implementar en una empresa o a una red mundial como lo es la Internet.

Bajo este modelo cada usuario tiene la libertad de obtener la información que requiera en un momento dado proveniente de una o varias fuentes locales o distantes y de procesarla como según le convenga. Los distintos servidores también pueden intercambiar información dentro de esta arquitectura.

#### **3.1.4.5 Que es una arquitectura**

Una arquitectura es un entramado de componentes funcionales que aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos, de manera que pueden ser utilizados eficazmente dentro de la organización.

Debemos señalar que para seleccionar el modelo de una arquitectura, hay que partir del contexto tecnológico y organizativo del momento y, que la arquitectura Cliente/Servidor requiere una determinada especialización de cada uno de los diferentes componentes que la integran.

### 3.1.1 Que es un cliente

Es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes LAN o WAN. La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente.

### 3.1.2 Que es un servidor

Es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LANs o WANs, para proveer de múltiples servicios a los clientes y ciudadanos tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, etc ver figura (5.3.1.2.1).

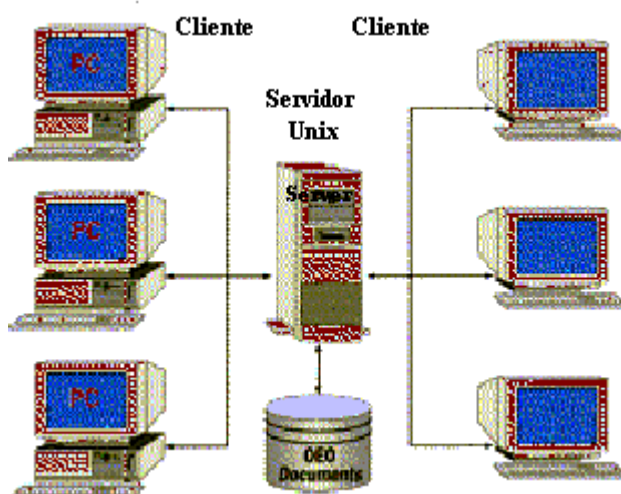


Figura 5.3.1.2.1<sup>18</sup> Este es el ejemplo gráfico de la arquitectura cliente servidor.

<sup>18</sup><http://www.google.com.ec/servidores>

### 3.1.3 Elementos de la arquitectura de un servidor

En esta aproximación, y con el objetivo de definir y delimitar el modelo de referencia de una arquitectura Cliente/Servidor, debemos identificar los componentes que permitan articular dicha arquitectura, considerando que toda aplicación de un sistema de información está caracterizada por tres componentes básicos:

- Presentación/Captación de Información
- Procesos
- Almacenamiento de la Información

Los cuales se suelen distribuir tal como se presenta en la figura (5.3.1.2.2)

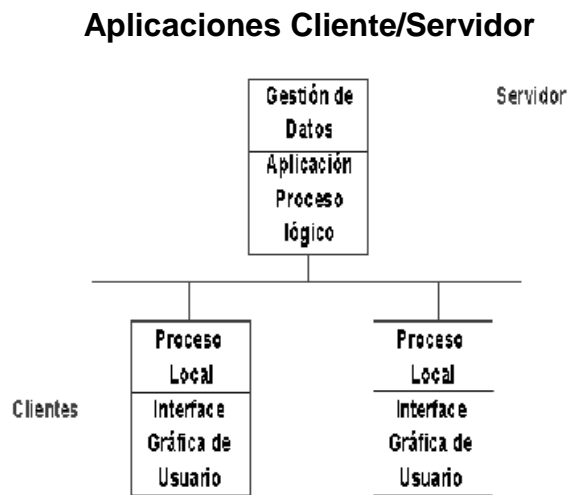


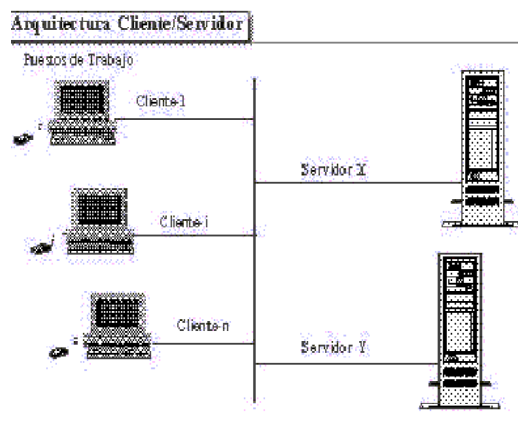
Figura (5.3.1.2.2)

Y se integran en una arquitectura Cliente/Servidor en base a los elementos que caracterizan dicha arquitectura, es decir:

- Puestos de Trabajo
- Comunicaciones
- Servidores

Tal como se presenta en la figura: (5.3.1.2.2)





**Figura (5.3.1.2.2) <sup>19</sup>Arquitectura Cliente/Servidor**

De estos elementos debemos destacar:

### **El Puesto de Trabajo o Cliente**

Una Estación de trabajo o microcomputador (PC: Computador Personal) conectado a una red, que le permite acceder y gestionar una serie de recursos» el cual se perfila como un puesto de trabajo universal. Nos referimos a un microcomputador conectado al sistema de información y en el que se realiza una parte mayoritaria de los procesos.

Se trata de un fenómeno en el sector informático. Aquellos responsables informáticos que se oponen a la utilización de los terminales no programables, acaban siendo marginados por la presión de los usuarios. Debemos destacar que el puesto de trabajo basado en un microcomputador conectado a una red, favorece la flexibilidad y el dinamismo en las organizaciones. Entre otras razones, porque permite modificar la ubicación de los puestos de trabajo, dadas las ventajas de la red.

### **Los Servidores o Back-end**

Una máquina que suministra una serie de servicios como Bases de Datos, Archivos, Comunicaciones,...).

Los Servidores, según la especialización y los requerimientos de los servicios que debe suministrar pueden ser:

<sup>19</sup><http://images.google.com.ec/images?hl=es&q=cliente%20servidor&um=1&ie=UTF-8&sa=N&tab=wi>



- Mainframes
- Miniordenadores

Especializados (Dispositivos de Red, Imagen, etc.)

Una característica a considerar es que los diferentes servicios, según el caso, pueden ser suministrados por un único Servidor o por varios Servidores especializados.

## **Las Comunicaciones**

En sus dos vertientes:

- Infraestructura de redes
- Infraestructura de comunicaciones

### **Infraestructura de redes**

Componentes Hardware y Software que garantizan la conexión física y la transferencia de datos entre los distintos equipos de la red.

### **Infraestructura de comunicaciones**

Componentes Hardware y Software que permiten la comunicación y su gestión, entre los clientes y los servidores.

La arquitectura Cliente/Servidor es el resultado de la integración de dos culturas. Por un lado, la del Mainframe que aporta capacidad de almacenamiento, integridad y acceso a la información y, por el otro, la del computador que aporta facilidad de uso (cultura de PC), bajo costo, presentación atractiva (aspecto lúdico) y una amplia oferta en productos y aplicaciones.

#### **3.1.5 Características del modelo cliente servidor**

En el modelo CLIENTE/SERVIDOR podemos encontrar las siguientes características:



1. El Cliente y el Servidor pueden actuar como una sola entidad y también pueden actuar como entidades separadas, realizando actividades o tareas independientes.
2. Las funciones de Cliente y Servidor pueden estar en plataformas separadas, o en la misma plataforma.
3. Un servidor da servicio a múltiples clientes en forma concurrente.
4. Cada plataforma puede ser escalable independientemente. Los cambios realizados en las plataformas de los Clientes o de los Servidores, ya sean por actualización o por reemplazo tecnológico, se realizan de una manera transparente para el usuario final.
5. La interrelación entre el hardware y el software están basados en una infraestructura poderosa, de tal forma que el acceso a los recursos de la red no muestra la complejidad de los diferentes tipos de formatos de datos y de los protocolos.
6. Un sistema de servidores realiza múltiples funciones al mismo tiempo que presenta una imagen de un solo sistema a las estaciones Clientes. Esto se logra combinando los recursos de cómputo que se encuentran físicamente separados en un solo sistema lógico, proporcionando de esta manera el servicio más efectivo para el usuario final.

También es importante hacer notar que las funciones Cliente/Servidor pueden ser dinámicas. Ejemplo, un servidor puede convertirse en cliente cuando realiza la solicitud de servicios a otras plataformas dentro de la red. Su capacidad para permitir integrar los equipos ya existentes en una organización, dentro de una arquitectura informática descentralizada y heterogénea.

7. Además se constituye como el nexo de unión más adecuado para reconciliar los sistemas de información basados en mainframes o minicomputadores, con aquellos otros sustentados en entornos informáticos pequeños y estaciones de trabajo.
8. Designa un modelo de construcción de sistemas informáticos de carácter distribuido.



1. Su representación típica es un centro de trabajo (PC), en donde el usuario dispone de sus propias aplicaciones de oficina y sus propias bases de datos, sin dependencia directa del sistema central de información de la organización, al tiempo que puede acceder a los
2. recursos de este host central y otros sistemas de la organización ponen a su servicio.

En conclusión, Cliente/Servidor puede incluir múltiples plataformas, bases de datos, redes y sistemas operativos. Estos pueden ser de distintos proveedores, en arquitecturas propietarias y no propietarias y funcionando todos al mismo tiempo. Por lo tanto, su implantación involucra diferentes tipos de estándares: APPC, TCP/IP, OSI, NFS, DRDA corriendo sobre DOS, OS/2, Windows o PC UNIX, en TokenRing, Ethernet, FDDI o medio coaxial, sólo por mencionar algunas de las posibilidades.

### **Tipos de clientes**

#### **“cliente flaco”:**

Servidor rápidamente saturado.

Gran circulación de datos de interface en la red.

#### **“cliente gordo”:**

Casi todo el trabajo en el cliente.

No hay centralización de la gestión de la BD.

Gran circulación de datos inútiles en la red. Ver **figura (3.1.4.6.1)**

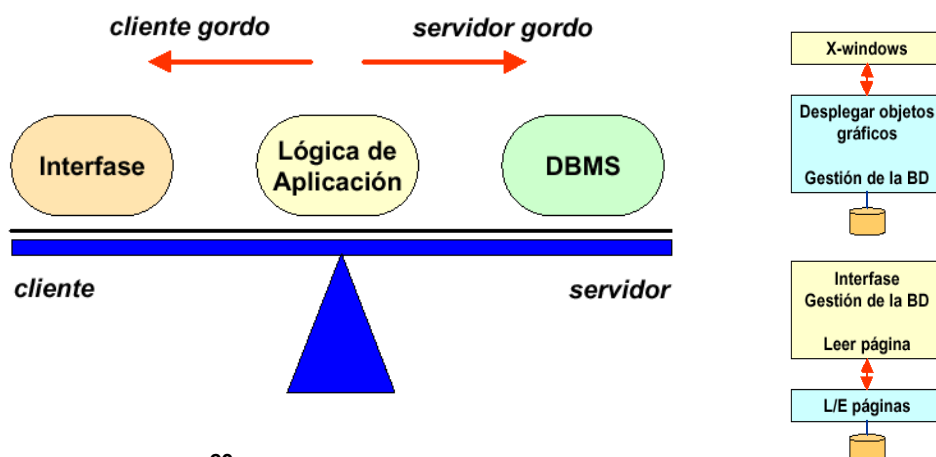


Figura 3.1.4.6.1 <sup>20</sup>Imagen cliente servidor

### 3.1.4.6 Tipos de Servidor

#### Servidores de archivos

Servidor donde se almacena archivos y aplicaciones de productividad como por ejemplo procesadores de texto, hojas de cálculo, etc.

#### Servidores de bases de datos

Servidor donde se almacenan las bases de datos, tablas, índices. Es uno de los servidores que más carga tiene.

#### Servidores de transacciones

Servidor que cumple o procesa todas las transacciones. Valida primero y recién genera un pedido al servidor de bases de datos.

#### Servidores de Groupware

Servidor utilizado para el seguimiento de operaciones dentro de la red.

<sup>20</sup><http://images.google.com.ec/images?hl=es&q=cliente%20servidor&um=1&ie=UTF-8&sa=N&tab=wi>



## **Servidores de objetos**

Contienen objetos que deben estar fuera del servidor de base de datos. Estos objetos pueden ser videos, imágenes, objetos multimedia en general.

## **Servidores Web**

Se usan como una forma inteligente para comunicación entre empresas a través de Internet.

Este servidor permite transacciones con el acondicionamiento de un browser específico.

### **3.1.5 Estilo de modelo cliente servidor**

#### *PRESENTACIÓN DISTRIBUIDA*

Se distribuye la interfaz entre el cliente y la plataforma servidora.

La aplicación y los datos están ambos en el servidor.

Similar a la arquitectura tradicional de un Host y Terminales.

El PC se aprovecha solo para mejorar la interfaz gráfica del usuario

#### **3.1.5.1 Ventajas**

Revitaliza los sistemas antiguos.

Bajo costo de desarrollo.

No hay cambios en los sistemas existentes

#### **3.1.5.2 Desventajas**

El sistema sigue en el Host.

No se aprovecha la GUI y/o LAN.



La interfaz del usuario se mantiene en muchas plataformas

### **3.1.6 Lógica remota**

La interfaz para el usuario está completamente en el cliente.

La aplicación y los datos están en el servidor.

#### **3.1.6.1 Ventaja**

La interfaz del usuario aprovecha bien la GUI y la LAN.

La aplicación aprovecha el Host.

Adecuado para algunos tipos de aplicaciones de apoyo a la toma de decisiones.

#### **3.1.6.2 Desventaja**

- Las aplicaciones pueden ser complejas de desarrollar.
- Los programas de la aplicación siguen en el Host.
- El alto volumen de tráfico en la red puede hacer difícil la operación de aplicaciones muy pesadas.

### **3.1.7 Funciones de un cliente servidor**

Espera las solicitudes de los clientes.

Ejecuta muchas solicitudes al mismo tiempo.

Atiende primero a los clientes VIP.

Emprende y opera actividades de tareas en segundo plano.

Se mantiene activa en forma permanente



## **6 METODOLOGIA**





## 6.1 MATRIZ DE CONSISTENCIA GENERAL

### PROBLEMA DE INVESTIGACIÓN

“En la actualidad no existe un control en los cambios de actualizaciones de los proyectos en desarrollo utilizando un almacenamiento de información de sus datos o código fuente centralizado SVN, en la Unidad de Desarrollo Informático de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.”

TEMA	OBJETO DE INVESTIGACIÓN	OBJETIVO DE LA INVESTIGACIÓN	HIPÓTESIS DE INVESTIGACIÓN
<b>Configuración e implementación de un servidor en Linux. Para repositorio de versiones utilizando SVN, para la unidad de desarrollo informático del A.E.I.R.N.N.R de la universidad nacional de Loja</b>	Sistema Repositorio centralizado de versiones SVN en Linux en la Unidad Desarrollo Informático de la carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.	Implementar un sistema informático cliente-servidor de repositorios de versiones SVN que permitirá el seguimiento de proyectos en desarrollo, para la Unidad de Desarrollo Informático de la carrera en Ingeniería en Sistema del A.E.I.R.N.N.R. De “La Universidad Nacional de Loja” mejorando los procesos de desarrollo en software	Con Implementación del sistema de versiones de repositorios SVN centralizado se podrá, optimizar los procesos de software en desarrollo en el Departamento de Desarrollo Informático de la carrera en Ingeniería en Sistema del A.E.I.R.N.N.R. De “La Universidad Nacional de Loja”



## **PROBLEMAS ESPECÍFICOS:**

1. Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones SVN en Linux para el desarrollo de proyectos de sistemas informáticos.
2. Actualmente el departamento de desarrollo Informático no cuenta con un repositorio de versiones de proyectos en desarrollo de los egresados de la carrera de ingeniería en sistemas.
3. Actualmente el departamento de desarrollo no cuenta con un manual de repositorio de versiones SVN para la administración de los proyectos en el departamento de desarrollo informático de la carrera de ingeniería en sistemas
4. Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones SVN cliente servidor en Linux para el seguimiento y control de proyectos mediante accesos a los clientes en forma visual
5. Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones para el seguimiento y control de proyectos mediante en el departamento de desarrollo informático de la carrera de ingeniería en sistemas

## **6.2 Materiales, métodos y técnicas de trabajo**

### **Los materiales de trabajo son:**

- Lapiceros
- Copias
- Resma de papel
- Impresora
- Recarga de tóner
- Recarga de tinta negra
- Recarga de tinta a color
- Transporte
- Computador



- Memoria de 4GB

### Diseño metodológico del proyecto de la investigación

La presente investigación se realizará la observación de todos los elementos que conforman el objeto de estudio compuesto por funciones y ámbitos de análisis; es descriptiva porque facilitará el análisis del comportamiento de los estándares e indicadores de calidad que se utilizan para evaluar el proceso de actual que cumple el departamento del área.

Además como tipo de estudio, se usará el **método exploratorio**, debido a que este nos permitirá familiarizarnos con los problemas de promoción y difusión del Departamento de “DESARROLLO DE SOFTWARE” , además se podrá identificar conceptos o variables relevantes que estén relacionadas con los problemas a investigar.

**Método cualitativo y cuantitativo**, el *cualitativo* se lo usará en las observaciones que se realizará de los elementos de observación y al *cuantitativo* se utilizará en las tabulaciones respectivas y en procesos donde se trabaje con cantidades, cifras numéricas, etc.

Para la recolección de la información relacionada a las actividades, problemas, causas y posibles alternativas de solución referentes a los de elementos de observación nos ayudaremos del **método deductivo**, junto con la observación fortalecerá los datos obtenidos.

### Metodología para la ejecución de la Investigación

La herramienta que se plantea permitirá realizar el control del subversiones SVN que tienen sistema operativo Linux, cuya versión de kernel se 2.4, 2.6 o superiores, permitirá realizar una configuración de la red, para de este modo realice las subversiones de proyectos en la carrera de ingeniería en sistemas.

La investigación exploratoria tiene un diseño especial, que aplicado a esta disciplina científica nos servirá de mucho, puesto que su objetivo es realizar una investigación preliminar mediante la cual se realiza la observación



inmediata del área y de los elementos constitutivos del objeto que va ser investigado, permitirá formular los problemas, aclarar conceptos, reunir información y familiarizarnos con el fenómeno que deseamos investigar.

Para esto utiliza el método científico porque nos permitirá establecer una relación teórica y práctica dentro de este proceso. Este método es el proceso más objetivo, ya que nos ayudara a entender el proceso de los métodos inductivo y deductivo.

El **método inductivo**, crea leyes a partir de la observación de los hechos, mediante la generalización del comportamiento observado, nos servirá para la recolección de la información relacionada a las actividades, problemas, causas y posibles alternativas de solución referentes a los elementos de observación, **método deductivo** aspira a demostrar, mediante la lógica pura, la conclusión en su totalidad a partir de unas premisas, de manera que se garantiza la veracidad de las conclusiones.

Para realizar la observación de los elementos más importantes del objeto que se investiga, la investigación exploratoria necesita ayuda de una actividad de campo, en donde el mismo objeto de estudio nos servirá como fuente de información utilizando técnicas como:

- 2) La observación directa que consiste en la inspección y estudio por medio de la observación de las características más sobresalientes, logrando la captación de la realidad natural, que implica, procesos de cada departamento, área e identificar y acudir a personas que tengan un conocimiento completo del funcionamiento, con la finalidad de recoger información válida para el análisis descriptivo de los hechos que nos confieren.
- 3) La entrevista dirigida a empleados, usuarios y administrativos para obtener una información oral valida, clara y concisa como aporte en busca del correcto y normal funcionamiento.



## **Técnicas e instrumentos**

La implementación y configuración de control de subversiones " conlleva la recolección y procesamiento de la información de datos. Esta información se obtiene de la aplicación de técnicas e instrumentos que se realizara en el transcurso del proyecto.

La recopilación de información se realizará utilizando las siguientes técnicas:

### **Técnica entrevista**

Está técnica nos ayudará a una comunicación directa con los usuarios, con la finalidad de conocer el funcionamiento actual de las transacciones, procesos, sus seguridades, sus ventajas y desventajas, permitiéndonos mejorar el funcionamiento y el servicio que realizan en la actualidad.

Esta técnica de recopilación de información sirve para evaluar indicadores cualitativos que requieren de informantes denominados individuales.

La entrevista puede ser estructurada y no estructurada. La técnica estructurada es la que contempla un guión con las preguntas a ser aplicadas al entrevistado por parte del entrevistador.

La entrevista no estructurada en cambio es aquella que no tiene un libreto prediseñado con la preguntas. En investigación evaluativa se recomienda siempre utilizar la entrevista estructurada.

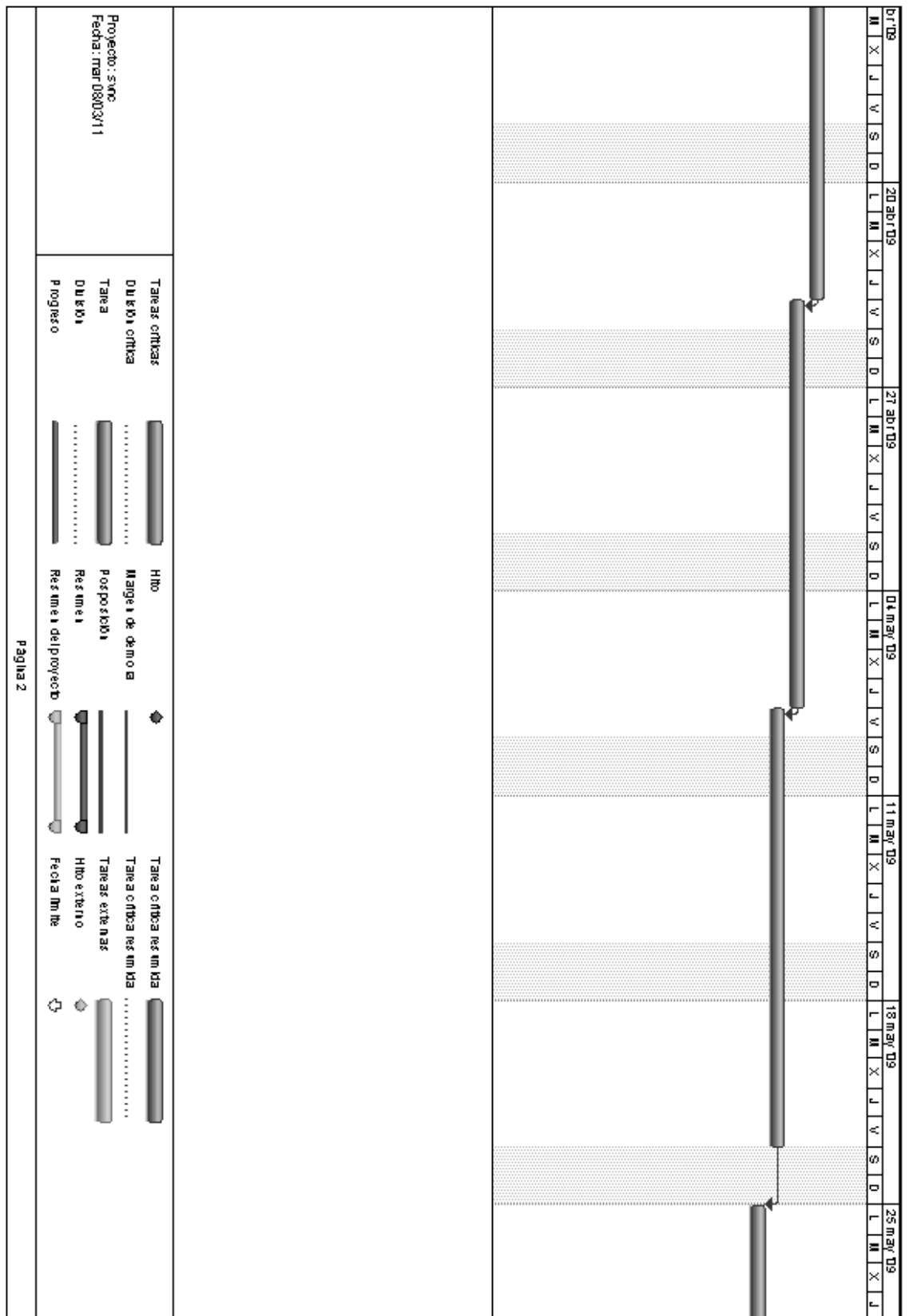
### **La observación**

La información recogida de las entrevistas nos servirá para complementar toda la información recopilada a través de la técnica de la **observación**, que nos ayudará a reforzar el proyecto de investigación. Mediante los catálogos, formularios, procesos que se realicen en cada uno de ellos con sus respectivos requerimientos

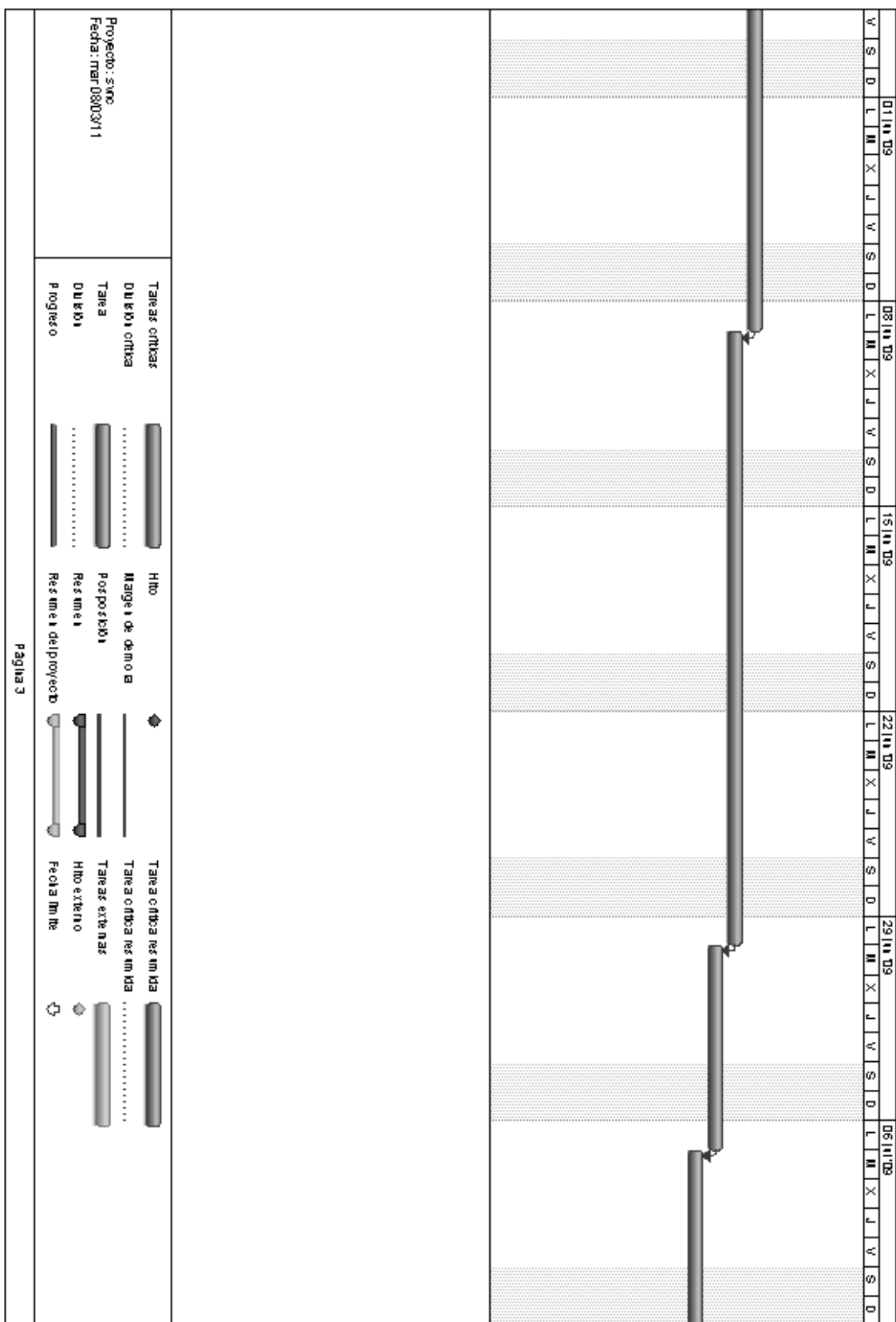


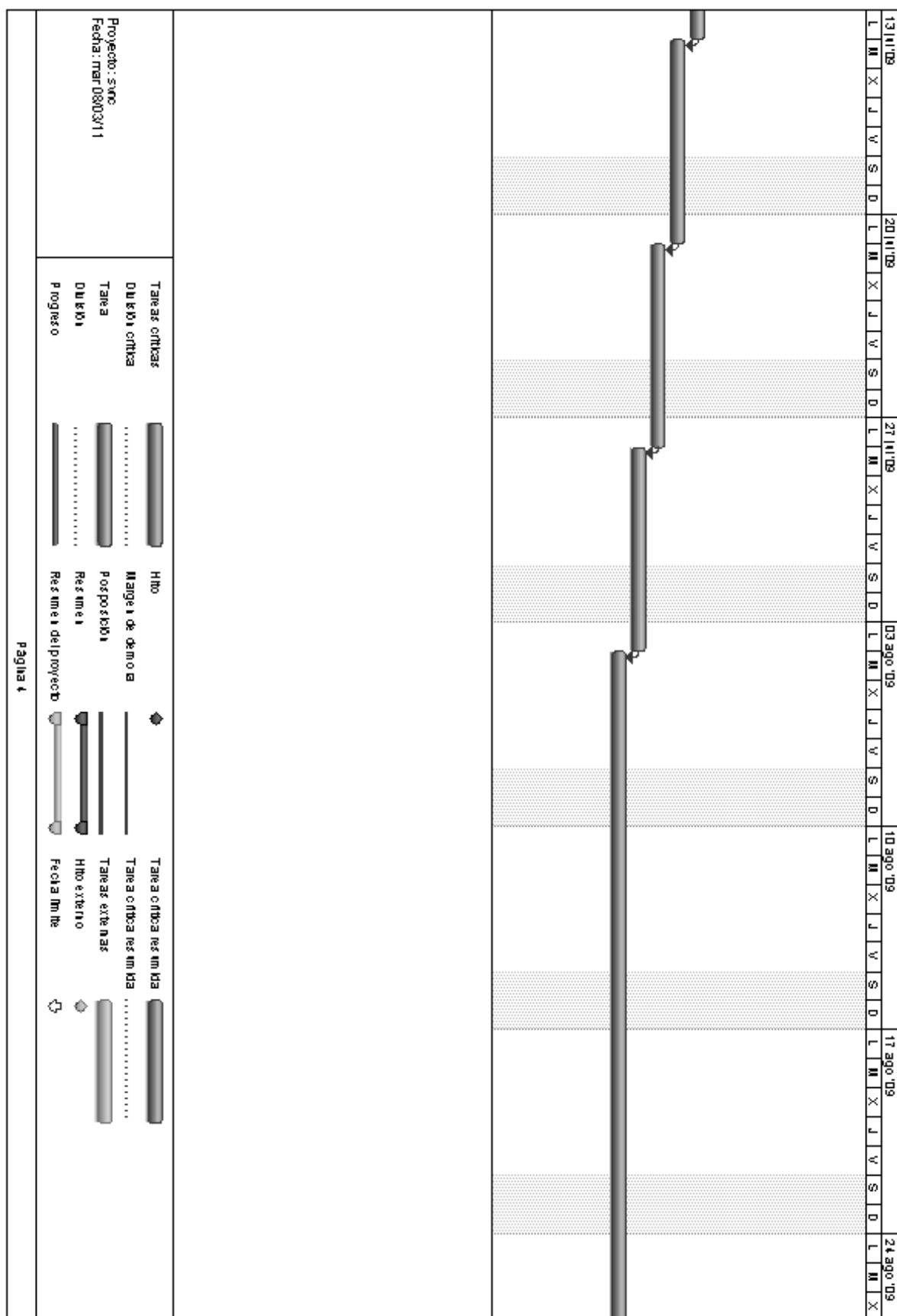
## **7 CRONOGRAMA**

[illegible]



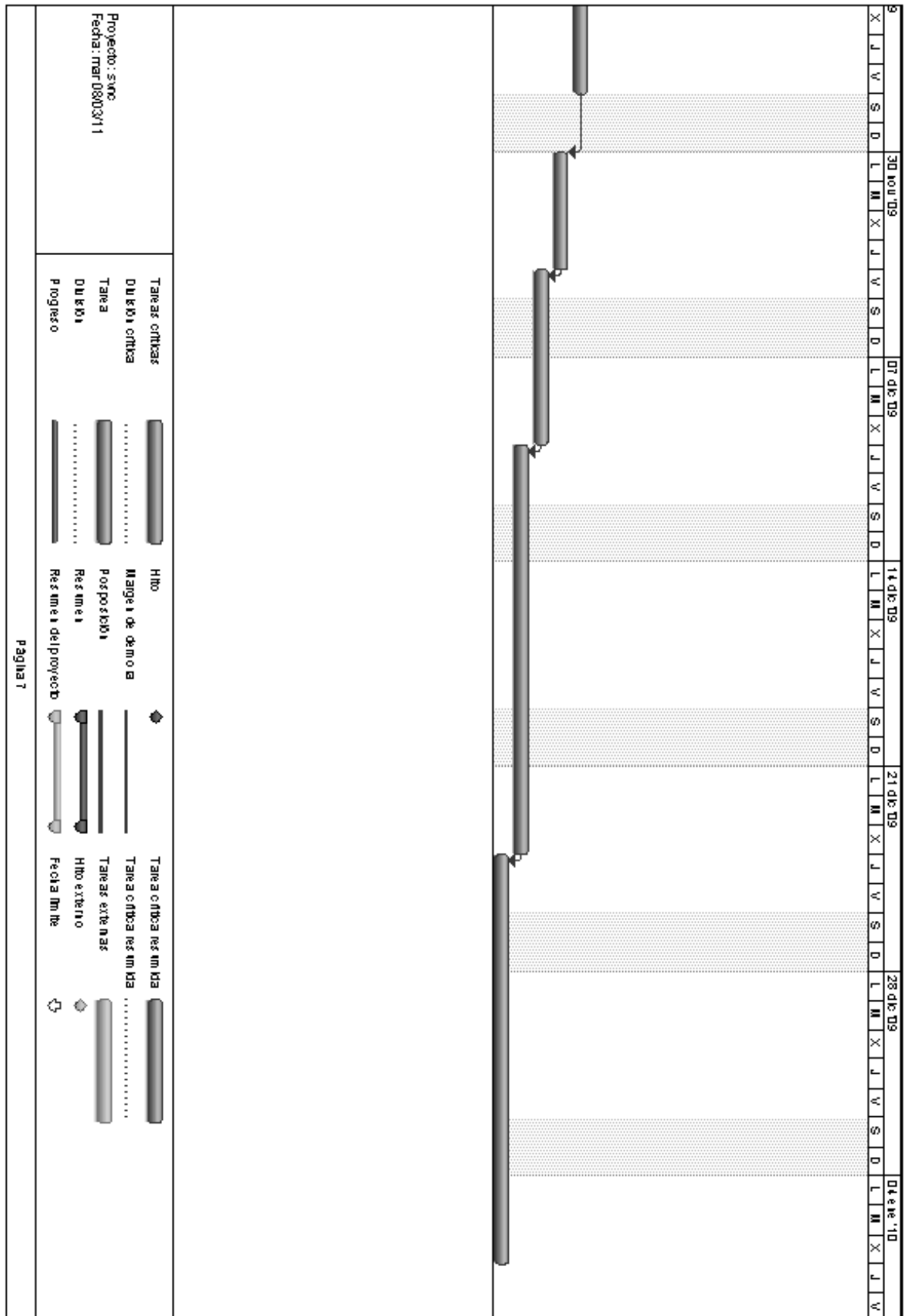






[illegible]







## **8 PRESUPUESTO Y FINANCIAMIENTO**



DESCRIPCIÓN	CANTIDAD	# DE HORAS	V/u \$	V/t \$
<b>RECURSOS HUMANOS</b>				
Investigadores	1	0	0.00	0.00
Unida de Desarrollo Informático	1	2	0.00	0.00
Asesor	1	3	250.00	250.00
Director de tesis	1	0	0.00	0.00
<b>RECURSOS TÉCNICOS</b>				
<b>HARDWARE</b>				
Computador	2		1200	2400
Impresora	1		50	50
Dispositivo de almacenamiento 8GB	1	0	35	35
<b>SOFTWARE</b>				
Microsoft office XP	1	--	220	220
Linux	1	0	0	0
Java	1	--	---	--
Open office 3.0	1	--	--	--
Base de Datos	1	--	--	--
Mysql		---	--	--



Poseidón	1	0	250	250
Net Beans 6.5	--	---	----	----

### RECURSOS MATERIALES

<i>Descripción</i>	<i>Cantidad</i>	<i>Valor Unitario</i>	<i>Valor Total</i>
Copias	-	\$30.00	\$30.00
Resma de papel	3	\$2.80	\$8.40
Recarga de toner	2	\$25.00	\$50.00
Recarga de tinta negra	4	\$5.00	\$20.00
Recarga de tinta a color	2	\$12.00	\$24.00
Transporte	--	\$100.00	\$300.00
Imprevistos	--	\$150.00	\$250.00
Internet	50	\$0.80	\$40.00
<b>TOTAL</b>			<b>\$ 722.40</b>





## 9 BIBLIOGRAFIA



### **Páginas Web:**

1. **Software Libre, Programación y algo mas... Blog dedicado en su mayoría a Linux, software libre y programación;**  
[<http://estrada-david.blogspot.com/2009/01/netbeanssvn.html>],  
[Consulta: 14enero 2009].
2. **Distribuciones Linux**  
[[http://es.wikipedia.org/wiki/Lista\\_de\\_Distribuci%C3%B3n\\_Linux](http://es.wikipedia.org/wiki/Lista_de_Distribuci%C3%B3n_Linux)],  
[Consulta: 16 enero]
3. **eAthena Support Board latest news** Blog dedicado en su mayoría a SVN  
[<http://www.eathena.ws/board/index.php?showtopic=124056>][Consulta : 24 enero 2009]
4. **Rafael Martínez. Copyright 1998-2009 El rincón de Linux -**  
[[http://www.linux-es.org/sobre\\_linux](http://www.linux-es.org/sobre_linux)], [Consulta: 6enero2009].
5. **Rafael Martínez El rincón de Linux**  
[<http://www.linux-es.org/distribuciones>], [Consulta: 15 enero 2009].
6. [<http://www.monografias.com/trabajos14/linux/linux.shtml>], [Consulta: 19 enero 2009].
7. **Open Source Software Engineering Tools**  
[<http://subversion.tigris.org/>], [Consulta: 22 enero 2009].
8. **GreenstoneWiki.-Welcome to GreenstoneWiki: Documentation for Greenstone**  
[[http://wiki.greenstone.org/wiki/index.php/Install\\_SVN\\_on\\_Linux](http://wiki.greenstone.org/wiki/index.php/Install_SVN_on_Linux)][Consulta: 27 enero]



## 10 ANEXOS

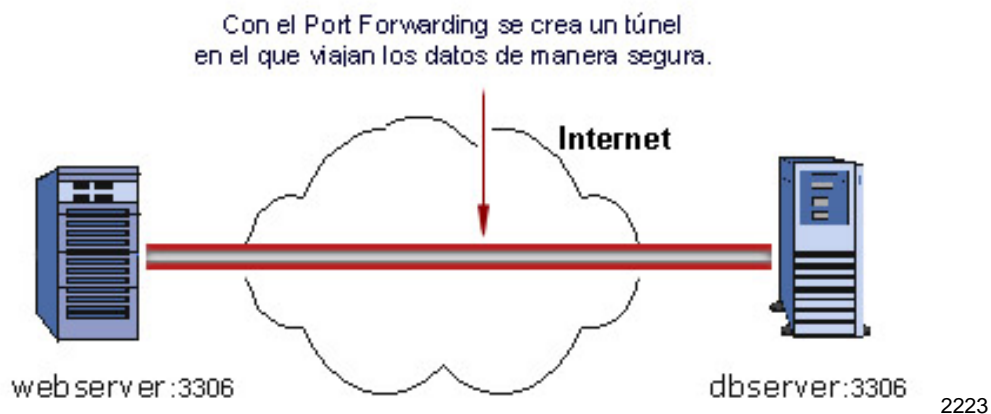
## 10. ANEXOS

Figura 5.3.1.1



21

Figura 5.3.1.2

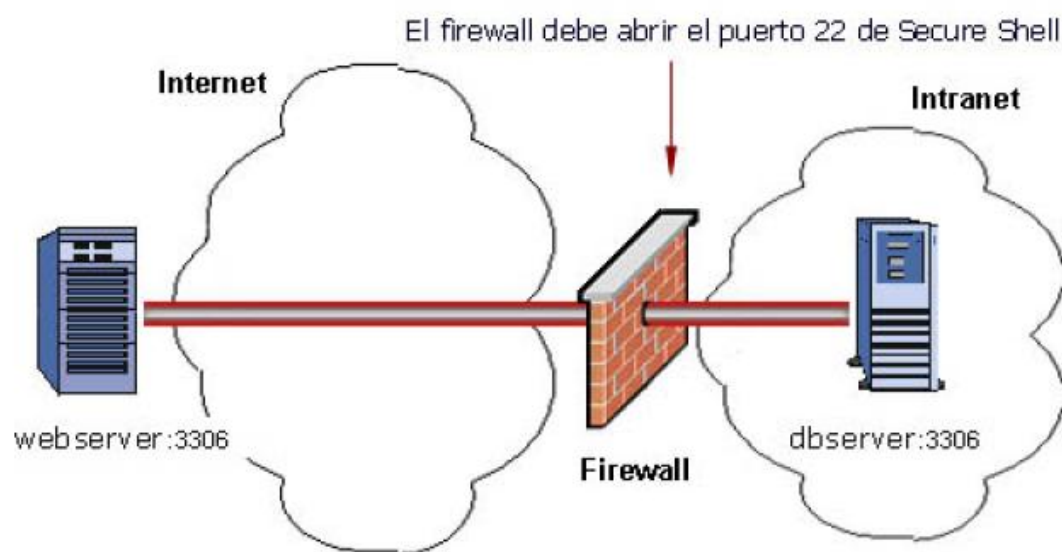


<sup>21</sup><http://images.google.com/ec/images?hl=es&q=cliente%20servidor&um=1&ie=UTF-8&sa=N&tab=wi>

<sup>22</sup><http://images.google.com/ec/images?hl=es&q=cliente%20servidor&um=1&ie=UTF-8&sa=N&tab=wi>

<sup>23</sup><http://images.google.com/ec/images?hl=es&q=cliente%20servidor&um=1&ie=UTF-8&sa=N&tab=wi>

**Figura 5.3.1.3**



24

<sup>24</sup><http://images.google.com.ec/images?hl=es&q=cliente%20servidor&um=1&ie=UTF-8&sa=N&tab=wi>



## 6.2 Matriz de consistencia específica10.1 Matriz De Consistencia Específica

**PROBLEMA ESPECÍFICO:** Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones SVN para el seguimiento y control de proyectos en el departamento de desarrollo informático de la carrera de ingeniería en sistemas.

OBJETIVO ESPECÍFICO	HIPÓTESIS ESPECÍFICA	UNIDAD DE OBSERVACIÓN	SISTEMA CATEGORIAL
<ul style="list-style-type: none"><li>• Instalación y configuración del servicio subversión SVN, con sus dependencias .</li></ul>	A través de la Instalación y configuración de un software de repositorio centralizado de SVN bajo Linux se podrá tener un mejor control y seguimientos de proyectos en sus actualizaciones de versiones.	Recopilación información bibliográfica acerca de cada uno de los modelos de versiones de repositorios SVN. Para la distribución Linux. Para una mejor eficiencia de actualizaciones de versiones de proyectos.	<ul style="list-style-type: none"><li>• Introducción a subversión</li><li>• Qué es Control de Versiones.</li><li>• Que es el control de subversiones</li><li>• Qué es Subversion</li></ul>



## 10.2 Matriz de operatividad de objetivos

<b>PROBLEMA ESPECÍFICO:</b> Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones SVN cliente servidor en Linux para el seguimiento y control de proyectos mediante accesos a los clientes en forma visual.			
<b>OBJETIVO ESPECÍFICO</b>	<b>HIPÓTESIS ESPECÍFICA</b>	<b>UNIDAD DE OBSERVACIÓN</b>	<b>SISTEMA CATEGORIAL</b>
<ul style="list-style-type: none"> <li>Instalación y configuración del servicio subversión SVN visual, para clientes.</li> </ul>	<p>Mediante la Instalación y configuración de un software de repositorio de SVN bajo Linux y Windows visual, se podrá tener una mejor facilidad de utilización de esta herramienta para los estudiantes de la Carrera de Ingeniería en Sistemas o interesados en el uso de la herramienta.</p>	<p>Recopilación información bibliográfica acerca de cada uno de los modelos de versiones de repositorios SVN visuales para un mejor control de acceso a usuarios para la actualización de versiones. Para la distribución Linux.</p>	<ul style="list-style-type: none"> <li>Introducción a subversión</li> <li>Qué es Control de Versiones.</li> <li>Que es el control de subversiones</li> <li>Qué es Subversion</li> <li>Herramientas de desarrollo Qué es Linux?</li> <li>Historia</li> <li>Aplicaciones</li> <li>Características</li> <li>Distribuciones de Linux</li> <li>Definición Arquitectura Cliente</li> </ul>



			Servidor
--	--	--	----------

**PROBLEMA ESPECÍFICO:** Actualmente el departamento de desarrollo Informático no cuenta con un repositorio de versiones de proyectos en desarrollo de los egresados de la carrera de ingeniería en sistemas.

OBJETIVO ESPECÍFICO	HIPÓTESIS ESPECÍFICA	UNIDAD DE OBSERVACIÓN	SISTEMA CATEGORIAL
▪ Construir los repositorios SVN de los proyectos desarrollados y en desarrollo por los egresados de la carrera con sus respectivos controles de accesos.	Realizando la Implantación de los repositorios de los proyectos que se encuentran en desarrollo en la carrera de Ingeniería en Sistemas al servidor del Departamento de la Unidad de Desarrollo Informáticos con sus respectivas claves de acceso, les permitirá tener un mayor	Recopilación información de todos los proyectos en desarrollo de tesis de la carrera de ingeniería en sistemas	<ul style="list-style-type: none"><li>• Qué es Control de Versiones.</li><li>• Que es el control de subversiones</li><li>• Qué es Subversion</li><li>• Herramientas de desarrollo Qué es Linux?</li><li>• Historia</li><li>• Aplicaciones</li><li>• Características</li><li>• Distribuciones de Linux</li></ul> Definición Arquitectura Cliente Servidor





	control de los proyectos en sus actualizaciones y reversiones a sus históricas. Y sus versiones siguientes		
--	---	--	--



**PROBLEMA ESPECÍFICO:** Actualmente el departamento de desarrollo no cuenta con un manual de repositorio de versiones SVN para la administración de los proyectos en el departamento de desarrollo informático de la carrera de ingeniería en sistemas.

OBJETIVO ESPECÍFICO	HIPÓTESIS ESPECÍFICA	UNIDAD DE OBSERVACIÓN	SISTEMA CATEGORIAL
Desarrollar un manual de administrador para el control de versiones.	Desarrollando un manual del administrador de la creación de proyectos. Con sus respectivos accesos de usuarios el cual nos permitirá tener un mejor conocimiento de esta herramienta, el cual permitirá que el estudiantado de la carrera de Ingeniería en Sistemas y las demás carreras puedan utilizar dicha herramienta, para desenvolverse dentro de la sociedad que los rodea.	Encargados de la Unidad de Desarrollo Informático de la carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.	<ul style="list-style-type: none"><li>• Qué es Control de Versiones.</li><li>• Que es el control de subversiones</li><li>• Qué es Subversion</li><li>• Herramientas de desarrollo Qué es Linux?</li><li>• Características</li><li>• Distribuciones de Linux</li><li>• Definición Arquitectura Cliente Servidor</li><li>• <b>Tipos de clientes</b></li><li>• <b>Tipos de Servidor</b></li><li>• <b>Estilo de modelo cliente servidor</b></li></ul>



**PROBLEMA ESPECÍFICO:** Actualmente el departamento de desarrollo no cuenta con un repositorio de versiones para el seguimiento y control de proyectos mediante en el departamento de desarrollo informático de la carrera de ingeniería en sistemas.

OBJETIVO ESPECÍFICO	HIPÓTESIS ESPECÍFICA	UNIDAD DE OBSERVACIÓN	SISTEMA CATEGORIAL
Aplicar el plan de validación del repositorio de versiones en los procesos de investigación modular de la carrera de Ingeniería en sistemas.	<ul style="list-style-type: none"><li>• Capacitando a los usuarios y administradores, programadores del software para que adquieran nuevas formas de programación permitiendo utilizar la herramienta de control de versión SVN en los procesos de investigación modular de la carrera de</li></ul>	Validación de repositorios en los procesos de investigación modular.	<ul style="list-style-type: none"><li>• <b>SVN.</b></li><li>• <b>Que es el control de subversiones</b></li><li>• <b>Actualizaciones de versiones</b></li></ul>



	Ingeniería en Sistemas		
--	------------------------	--	--



## 10.2 Matriz De Operatividad De Objetivos

### ▪ OBJETIVO ESPECÍFICO

Instalación y configuración del servicio subversión SVN, con sus dependencias

ACTIVIDAD O TAREA	METODOLOGÍA	FECHA		RESPONSABLE	PRESUPUESTO	RESULTADOS ESPERADOS
		INICIO	FIN			
Búsqueda de información	Realizar la búsqueda de información necesaria, que detalle las topologías implementadas en la red de la AEIRNNR	16/04/09	19/05/09	Diego Genaro Achupallas España	\$ 50	Buscar la mayor cantidad de documentos certificados referentes a las topologías implementadas en la red de la AEIRNNR.
Lectura y síntesis de información	Realizar la síntesis de la información buscada, las topologías implementadas en la red de la AEIRNNR	22/05/09	26/06/09	Diego Genaro Achupallas España	\$ 0.0	Validar y contraponer la información que se buscó para poder tener un criterio de las ventajas y la posterior utilización de las



						topologías implementadas en la red de
Implementar	Se consultará en Internet, manuales, reportes y libros relacionados con la SVN, e información sobre el manejo de las subversiones con sus respectivas configuraciones y manejo.	27/06/09	27/07/09	Diego Genaro Achupallas España	\$ 15.00	Información de modelo de repositorio SVN
configurar	Se consultará en Internet, manuales, reportes y libros relacionados con la SVN, e información sobre el manejo de las subversiones con sus respectivas configuraciones y manejo.	03/08/09	17/08/09	Diego Genaro Achupallas España	\$ 50	<ul style="list-style-type: none"><li>- Implementación de la herramienta de repositorio SVN cliente servidor SVN.</li><li>- Pruebas.</li></ul>



- **OBJETIVO ESPECÍFICO** Instalación y configuración del servicio subversión SVN visual, para clientes.

ACTIVIDAD O TAREA	METODOLOGÍA	FECHA		RESPONSABLE	PRESUPUESTO	RESULTADOS ESPERADOS
		INICIO	FIN			
Análisis de Datos	Recopilación bibliográfica, búsquedas en Internet sobre medidas de seguridad y acceso	10/08/09	12/08/09	<ul style="list-style-type: none"> <li>Diego Genaro Achupallas España</li> </ul>	\$ 40	Información detallada de distintas formas de seguridad y acceso a datos.
Diseño	Definir las seguridades de acceso correctas cliente servidor.	13/08/09	19/10/09	<ul style="list-style-type: none"> <li>Diego Achupallas</li> </ul>	\$ 40	Establecer que seguridades tendrá el sistema.



Instalación y Configuración	Codificación de seguridades de acceso a información en el sistema visual.	20/10/09	24/11/09	<ul style="list-style-type: none"><li>Diego Achupallas</li></ul>	\$ 350	Aplicación con seguridades correctas y funcionando.
Prueba	Verificación de seguridad de datos	01/12/09	22/12/09	<ul style="list-style-type: none"><li>Diego Achupallas</li></ul>	\$ 200	Un sistema de gran solidez y consistencia.





- **OBJETIVO ESPECÍFICO** Construir los repositorios SVN de los proyectos desarrollados y en desarrollo por los egresado de la carrera con sus respectivos controles de accesos.

ACTIVIDAD O TAREA	METODOLOGÍA	FECHA		RESPONSABLE	PRESUPUESTO	RESULTADOS ESPERADOS
		INICIO	FIN			
Pedir una entrevista al director del departamento de desarrollo	Elaborar un oficio para solicitar una entrevista	06/01/10	10/01/10	Diego Genaro Achupallas España	\$ 5	Atiendan a la solicitud y establezcan una fecha para la entrevista
Construcción de repositorios de los proyectos en desarrollo con sus respectivas claves de acceso.	Recopilación u obtención de Datos que servirán para la creación de repositorio centralizados SVN.	11/01/10	12/02/10	Diego Genaro Achupallas España	\$ 100	- - Ingresar todos los temas en desarrollo del tesis en ejecución de la carrera de ingeniería en sistemas en repositorio SVN.



pruebas y depuración del sistema	Verificación de seguridad de datos, y solides del sistema	14/02/10	24/02/10	<ul style="list-style-type: none"><li>• Diego Achupallas</li><li>•</li></ul>	\$ 200	Un sistema de gran solidez y consistencia.
----------------------------------	---	----------	----------	--	--------	--



▪ **OBJETIVO ESPECÍFICO**

Desarrollar un manual de administrador para el control de versiones

ACTIVIDAD O TAREA	METODOLOGÍA	FECHA		RESPONSABLE	PRESUPUESTO	RESULTADOS ESPERADOS
		INICIO	FIN			
Desarrollo de un manual de usuario servidor para la creación de repositorio	<ul style="list-style-type: none"><li>▪ Diseñar mediante diagramas, graficas.</li><li>▪ Desarrollo de componentes de la aplicación.</li><li>▪ Integrar los componentes a la aplicación.</li></ul>	26/02/10	27/03/10	Diego Genaro Achupallas España	\$ 20	Desarrollo de un manual fácil de comprender para el usuario final.



▪ **OBJETIVO ESPECÍFICO**

Aplicar el plan de validación de las subversiones en la carrera de Ingeniería en sistemas.

ACTIVIDAD O TAREA	METODOLOGÍA	FECHA		RESPONSABLE	PRESUPUESTO	RESULTADOS ESPERADOS
		INICIO	FIN			
pruebas y depuración del sistema implementado en su totalidad	Verificación de seguridad de datos	02/04/10	27/04/10	Diego Genaro Achupallas España	\$ 200	Un sistema de gran solidez y consistencia para desarrollo de sistemas en la Carrera de ingeniería en Sistemas.



### 10.3 Matriz De Control De Resultados

Nº	RESULTADOS	FECHA		FIRMA DIRECTOR DE TESIS
1	Atiendan a la solicitud y establezcan una fecha para la entrevista	03/05/09	04/05/09	
2	<ul style="list-style-type: none"><li>- Estructura del departamento</li><li>- Recopilación de información de la planificación.</li><li>- Información de la experiencia de profesionales.</li></ul>	05/05/09	10/05/09	
3	Haber obtenido toda la información suficiente para la implantación o del proyecto.	11/05/09	20/05/09	
4	Tener conocimiento de la mayoría de distribuciones Linux para su utilización y beneficio.  E instalación	22/05/09	01/06/09	
5	Buscar la mayor cantidad de documentos certificados referentes a las topologías implementadas en la red de la AEIRNNR.	03/06/09	09/06/09	
	Validar y contraponer la información que se buscó para poder tener un criterio de las ventajas y la posterior utilización de las topologías			



	implementadas en la red de	10/06/09	15/06/09	
	Información de modelo de repositorio SVN	16/06/09	16/07/09	
	- Implementación de la herramienta de repositorio SVN cliente servidor SVN. - Pruebas.	24/07/09	24/10/09	
	Información detallada de distintas formas de seguridad y acceso a datos.	28/10/09	9/11/09	
	Establecer que seguridades tendrá el sistema.	13/11/09	11/11/09	
	Aplicación con seguridades correctas y funcionando.	05/01/10	28/01/10	
	Un sistema de gran solidez y consistencia.	02/02/10	03/03/10	
	Desarrollo de un manual fácil de comprender para el usuario final	05/03/10	05/04/10	



	Un sistema de gran solidez y consistencia para desarrollo de sistemas en la Carrera de ingeniería en Sistemas.	06/04/10	27/04/10	
--	--	----------	----------	--



## Anexo 2. Glosario de Términos

- **Commit dat.\_** enviar cambios al repositorio. No hay traducción directa de "commit". Es habitual leer comentarios del tipo "Voy a hacer un commit para que veas los cambios". Dado que tanto en CVS como Subversion se trata de un programa cliente-servidor, realizar "un commit", es equivalente a enviar los cambios al repositorio o hacerlos públicos/visibles. Nunca traducir como sustantivo. Dependiendo del contexto, usar estas expresiones o variación aceptable:
  - ...enviar/subir cambios al repositorio...
  - ...guardar los cambios en el servidor...
  - ...hacer los cambios públicos/visibles...
- **branching and merging.\_** crear ramas y fusionarlas Aunque branch está bien traducido como rama o rama de desarrollo (en su versión "verbose"), no hay forma de hacer de un sustantivo un verbo. Por lo tanto, se crean, borran y fusionan ramas.
- **Browse.** – Navegar, usado con frecuencia para navegar por directorios o repositorios.
- **backup.** - copia de seguridad.
- **Command line.\_** (línea de comandos) lugar donde se escribe los comandos de la shell.
- **Command.-** (comandos) instrucción dada por el computador, en la mayoría de los casos con un teclado o mouse.
- **DAV share.\_** recurso DAV compartido.
- **file path.-** ruta del fichero.
- **Graphical user interface (gui):** pantalla gráfica, con íconos, menú y paneles, que el usuario pulsa para iniciar funciones.
- **Gnome (the gnu network object model environment):** gui incluidos en gnu/linux
- **Kde (common desktop environment):** gui incluidos en gnu/linux
- **open source .\_** código fuente abierto.
- **Panel.-** una barra de herramientas en entorno gráfico habitualmente localizada en la parte inferior de la pantalla.
- **plugins.-** módulos.





- **root.-** la cuenta root se crea durante la instalación y tiene acceso completo al sistema, es el super usuario.
- **su.-** el comando **su** le da acceso a la cuenta **root** u otras cuentas del sistema.
- **SHELL** .- Llamada también consola (Pantalla de comandos), interfaz igual al dos de Windows.
- **namespace.-** espacio de nombrado.
- **merge** - fusionar cambios, fusión Obtenida referencia del manual de CVS en castellano, la otra alternativa obvia para traducir merge es mezclar. No obstante, mezclar tiene una connotación de azar al realizar la mezcla. Se mezclan líquidos, ingredientes, etc. En cambio, un "merge" es cualquier cosa menos un proceso realizado al azar (especialmente si ha habido conflictos). En caso de traducir como sustantivo, fusión vuelve a "sonar mejor" que mezcla, que por alguna razón me suena a un combustible especial usado en vehículos de transporte.
- **Subversion.-** permite que ciertas opciones almacenen patrones con los que se ignoran ficheros no versionados.
- **location (repository).-** ubicación (del repositorio).
- **tag, tagging.\_** etiqueta, etiquetar.
- **working copy** .\_ copia (de trabajo) local/activa traducción similar a la de "commit data" (razonamiento cliente-servidor).



## Anexo 3 Modelo de la Encuesta



### UNIVERSIDAD NACIONAL DE LOJA

#### Área De Energía Industria Y Recursos Naturales No Renovables CARRERA DE INGENIERÍA EN SISTEMAS

#### ENCUESTA APLICADA A LOS USUARIOS DEL SERVICIO DE REPOSITORIOS DE SUBVERSION

1.- Considera usted que el servicio de repositorio subversion, implementado en el departamento de la Unidad de Desarrollo Informático de la Universidad Nacional de Loja es útil.

Si ( ) No ( )

Porque.....  
.....

.....  
....

2.- El manejo del Software TortoiseSVN es:

Fácil ( )      Difícil ( )      Confuso ( )

3.- La rapidez de actualización al recibir o realizar una actualización es:

Buena ( )      Excelente ( )      Mala ( )

4.- Ha tenido problemas con el servicio de subversion.

Si ( ) No ( )

Porque.....  
.....

.....  
.....



## Anexo 4 Certificaciones

### OFICIO PARA CERTIFICACIÓN DEL PROYECTO POR SECRETARIO DEL ÁREA

Fecha viernes 7 de enero de 2011

Señor Ingeniero

José Ochoa

**DIRECTOR DEL ÁREA DE ENERGÍA, INDUSTRIAS Y  
RECURSOS NATURALES NO RENOVABLES**  
Ciudad.

De mi consideración:

De conformidad a lo dispuesto en el numeral seis de la resolución administrativa emitida por el Consejo Académico Administrativo Superior en sesión ordinaria de 17 de septiembre de 2002, y en virtud de haber sido **aprobado** por su autoridad, el **proyecto de tesis** cuyo tema versa sobre " **CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMATICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA**", previo a optar el **título de Ingeniero** en Sistemas en la Carrera de Ingeniería en Sistemas de la Unidad Académica que acertadamente regenta; solicito a su autoridad, que en la continuación del trámite, se digne **conceder la autorización** respectiva, a fin de que el Secretario-Abogado del Área, **certifique la aprobación** del referido plan de investigación, para cuyo efecto **adjunto** a la presente los **derechos arancelarios** correspondientes.

En la seguridad de que su autoridad, se dignará atenderme favorablemente, me anticipo en presentarle mis cumplidos agradecimientos.

Atentamente,

.....

Egr. Diego Genaro Achupallas España



## OFICIO DE SOLICITUD DE TRIBUNAL CALIFICADOR DE TESIS

Fecha viernes 7 de enero de 2011

Señor Ingeniero

José Ochoa

COORDINADOR DE LA CARRERA DE

Ciudad

De mi consideración:

En virtud de haber obtenido la **declaratoria de aptitud legal** por parte del H. Consejo Académico del Área, y una vez que ha sido **aprobado el desarrollo de la tesis**, cuya problemática versa sobre el tema: "**CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMATICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA**", previo a optar el grado de Ingeniero en Sistemas en la Carrera de Ingeniería en Sistemas de esta Unidad Académica, conforme lo acredito con el informe del Director de la tesis que autoriza la presentación del mismo, y con la certificación que adjunto como documentos habilitantes; en la continuación del trámite respectivo, de conformidad a lo estipulado en el **NORMATIVO PARA LA ESTRUCTURACIÓN DE LAS ÁREAS** y en el **REGLAMENTO GENERAL PARA LA CONCESIÓN DE GRADOS Y TÍTULOS EN LA UNIVERSIDAD NACIONAL DE LOJA** en actual vigencia, solicito a usted, se digne designar el tribunal para que examine el trabajo, lo califique en sesión reservada e intervenga en la exposición respectiva, para cuyo efecto, adjunto cinco ejemplares en carpeta folder, conjuntamente con el proyecto aprobado.

Por la atención favorable que se digne dispensarme, le anticipo mis debidos agradecimientos.

Atentamente,

.....

Diego Genaro Achupallas España

Egresado.



## OFICIO PARA CERTIFICACIÓN DEL PROYECTO POR SECRETARIO DEL ÁREA

Fecha viernes 21 de enero de 2011

Señor Ingeniero

José Ochoa

DIRECTOR DEL ÁREA DE ENERGÍA, INDUSTRIAS Y  
RECURSOS NATURALES NO RENOVABLES

Ciudad.

De mi consideración:

De conformidad a lo dispuesto en el numeral seis de la resolución administrativa emitida por el Consejo Académico Administrativo Superior en sesión ordinaria de 17 de septiembre de 2002, y en virtud de haber sido **aprobado** por su autoridad, el **proyecto de tesis** cuyo tema versa sobre " **CONFIGURACION E IMPLEMENTACION DE UN SERVIDOR DE REPOSITORIO DE VERSIONES UTILIZANDO SVN EN LINUX, PARA LA UNIDAD DE DESARROLLO INFORMATICO DEL A.E.I.R.N.N.R DE LA UNIVERSIDAD NACIONAL DE LOJA**", previo a optar el **título de Ingeniero** en Sistemas en la Carrera de Ingeniería en Sistemas de la Unidad Académica que acertadamente regenta; solicito a su autoridad, que en la continuación del trámite, se digne **conceder la autorización** respectiva, a fin de que el Secretario-Abogado del Área, **certifique la aprobación** del referido plan de investigación, para cuyo efecto **adjunto** a la presente los **derechos arancelarios** correspondientes.

En la seguridad de que su autoridad, se dignará atenderme favorablemente, me anticipo en presentarle mis cumplidos agradecimientos.

Atentamente,

.....  
Egr. Diego Genaro Achupallas España



---

---