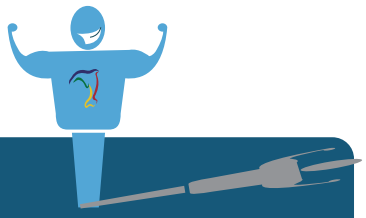
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

¿**Qué ofrece** Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

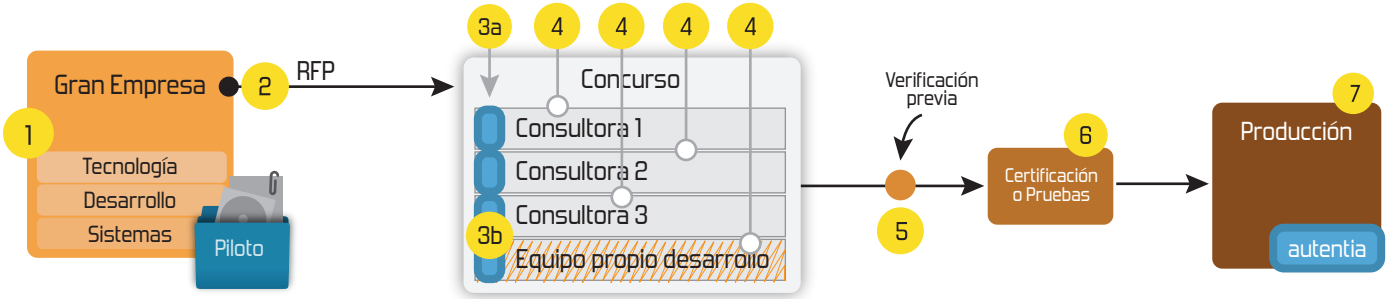
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

- Spring MVC, JSF-PrimeFaces /RichFaces, HTML5, CSS3, JavaScript-jQuery
- Gestor portales (Liferay)
- Gestor de contenidos (Alfresco)
- Aplicaciones híbridas
- Control de autenticación y acceso (Spring Security)
- UDDI
- Web Services
- Rest Services
- Social SSO
- SSO (Cas)
- JPA-Hibernate, MyBatis
- Motor de búsqueda empresarial (Solr)
- ETL (Talend)
- Dirección de Proyectos Informáticos.
- Metodologías ágiles
- Patrones de diseño
- TDD
- Tareas programadas (Quartz)
- Gestor documental (Alfresco)
- Inversión de control (Spring)
- BPM (jBPM o Bonita)
- Generación de informes (JasperReport)
- ESB (Open ESB)

Compartimos nuestro conociemiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

Tutorial desarrollado por: Alejandro Perez García 2003-2005, nuestro experto en J2EE, Linux y optimización de aplicaciones empresariales.

Si te gusta lo que ves, **puedes contratarme** para impartir **cursos presenciales** en tu empresa o para ayudarte en proyectos (Madrid).

Contacta: alejandropg@autentia.com.

Descargar este documento en formato PDF subversion.pdf

Firma en nuestro libro de Visitas

Subversion, sistema de control de versiones, en Debian GNU/Linux

Creación: 09-08-2006

Índice de contenidos

1. Introducción

En este tutorial vamos a hacer una introducción a Subversion, comentando como instalarlo y configurarlo en Debian GNU/Linux.

Subversion es un sistema de control de versiones. Es decir, nos va a permitir guardar todo el histórico de cambios de nuestros ficheros.

Es muy similar al CVS, pero ofreciendo ciertas ventajas sobre este. Para mi las principales ventajas frente al CVS son:

- Gestión de metadatos: en CVS si cambiamos el nombre a un fichero, es como si el fichero fuera borrado y se creara uno nuevo, no tienen conexión. Pasa lo mismo si el fichero se mueve a otro directorio. Subversión es capaza de gestionar estas situaciones, de forma que no perdemos el histórico.
- Commits transaccionales: es decir o se suben todos los cambios o no se sube ninguno. De esta forma el repositorio siempre tiene un estado consistente.
- Número de revisión por repositorio en vez de por fichero: de esta forma sabiendo el número de revisión tenemos una fot completa de todo el repositorio en un momento dado.

Subversion es un producto open source de la comunidad Tigris.org Open Source Software Engineering Tools (http://subversion.tigris.org/).

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil Ahtec Signal 259M MXM (Sonoma 2.1 GHz, 2048 MB RAM, 100 GB HD).
- Sistema Operativo: GNU / Linux, Debian Sid (unstable), Kernel 2.6.17, KDE 3.5
- Máquina Virtual Java: JDK 1.5.0_07 de Sun Microsystems
- Subversion 1.3.2-5

3. Instalación de Subversion

Para instalar Subversion en Debian basta con ejecutar:

```
# apt-get install subversion subversion-tools
```

4. Configuración de Subversión en el servidor

La configuración que se plantea a continuación no es más que un ejemplo, y por supuesto no es la única forma de configurar Subversion (y posiblemente tampoco sea la mejor, aunque a mi no me parece mal ;)

4.1. Usuario administrador del repositorio

Es buena práctica crear un usuario svn. De esta forma todas las operaciones sobre el repositorio se harán usando este usuario. En ningún caso se harán las operaciones como root.

También debe existir en el sistema el grupo svn. Por supuesto el usuario svn debe pertenecer al grupo svn.

Se debe asegura que el usuario svn tiene definido correctamente el umask. Para ello modificamos el fichero /home/svn/.bash_profile y añadimos la línea:

```
umask 027
```

Esto nos garantiza que todos los ficheros que cree este usuario sólo serán visibles por él mismo y por el grupo svn, pero otros usuarios no podrán acceder ni en lectura ni en escritura.

4.2. "Home" de los repositorios de Subversion

La home de Subversion, es decir, el directorio del cual colgaran todos los repositorio será:

/var/lib/svn

Este directorio pertenecerá al usuario root y al grupo svn. Y tendrá los permisos: drwxrws--- (nótese como el directorio está marcado para que se fije le ID del grupo al crear nuevos directorios, y como no hay permisos concedidos para "others").

4.3. Configuración del servicio como demonio

Como servidor se utilizará svnserve. Se utiliza esta opción en vez de apache2 por:

- Mejor velocidad.
- Simplicidad de instalación y configuración, sin perder las prestaciones necesarias (usuarios independientes, configuraciór de acceso de sólo lectura o escritura para los usuarios, ...).

Se debe crear el fichero /etc/init.d/svnserve con el siguiente contenido:

```
#!/bin/sh
#
# start/stop svn (Subversion) server.

set -e

NAME=svnserve
DESC="Subversion server"
DAEMON=/usr/bin/$NAME
PARAMS="-d -T -r /var/lib/svn"
DAEMONUSER=svn

test -x $DAEMON || exit 0

. /lib/lsb/init-functions

start_it_up()
{
log_daemon_msg "Starting $DESC" "$NAME"
start-stop-daemon --start --quiet --chuid $DAEMONUSER:$DAEMONUSER --exec $DAEMON -- $PARAMS
log_end_msg $?
}

shut_it_down()
{
log_daemon_msg "Stopping $DESC" "$NAME"
start-stop-daemon --stop --retry 60 --quiet --oknodo --exec $DAEMON
log_end_msg $?
}

case "$1" in
start)
start_it_up
;;
stop)
shut_it_down
;;
restart)
shut_it_down
start_it_up
;;
*)
echo "Usage: /etc/init.d/$NAME {start|stop|restart}" >&2
exit 1
;;
esac
```

```
exit 0
```

Nótese como la variable PARAMS define los parámetros con los que se arrancará svnserve:

- -d ejecuta svnserve como un demonio.
- -T para atender las peticiones se usarán hilos, en vez de procesos.
- -r permite a los clientes no tener que escribir la ruta completa. Esto resulta más cómodo al cliente y aporta cierta seguridad al ocultar los detalles de donde están realmente los repositorios.

Ahora sólo nos queda crear los enlaces simbólicos necesarios en la estructura /etc/rcx.d para que el servicio se arranque y se pare automáticamente al arrancara y parar el servidor. Para ello se hará:

```
# update-rc.d svnserve defaults
```

4.4. Configuración del servicio con inetd

Atención, esta opción ha sido descartada por problemas de rendimiento y estabilidad. Sólo se muestra aquí para que quede documentado como se tendría que hacer.
inetd crea un nuevo proceso por cada petición que llega al servidor. Además, tras varias pruebas se ha visto que cuando se hacen varias peticiones seguidas inetd piensa que se está haciendo un ataque (looping) y deja de dar servicio. Podría configurarse inetd para permitir más conexiones por segundo, pero no interesa.

En el fichero /etc/services deben añadirse las siguientes dos líneas (si todavía no existen):

```
svn 3690/tcp subversion # Subversion protocol
svn 3690/udp subversion
```

Para que funcione svnserve se añadirá al fichero /etc/inetd.conf la siguiente línea:

```
svn stream tcp nowait svn.svn /usr/bin/svnserve svnserve -i -r /var/lib/svn
```

La opción -r permite a los clientes no tener que escribir la ruta completa. Esto resulta más cómodo al cliente y aporta cierta seguridad al ocultar los detalles de donde están realmente los repositorios.

Para que se cojan los cambios se puede ejecutar:

```
# /etc/init.d/inetd restart
```

esto reinicio el demonio inetd.

5. Configuración de los repositorios

5.1. Creación de repositorios

Para crear un repositorio se usará el comando:

```
$ svnadmin create /var/lib/svn/nombreRepositorio
```

Donde "nombreRepositorio" es el nombre del repositorio que se está creando.

Los repositorios se crearán usando FSFS (versioned filesystem) como sistema de almacén de los datos. Se prefiere este sistema antes que utilizará Berkeley DB por posibles problemas de portabilidad entre plataformas y otros problemas que se pueden dar con Berkeley DB.

5.2. Configuración de la seguridad de los repositorios

Todos los repositorios tendrán permiso de lectura para usuarios anónimos y sólo permiso de escritura para los usuario registrados. Con esto conseguimos que cualquier miembro del equipo pueda acceder en consulta a todos los proyectos, pero sól el personal autorizado puede modificarlos.

Esto se consigue poniendo las siguiente opciones en el fichero svnserve.conf que se encuentra en el directorio conf dentro del repositorio:

```
anon-access = read
auth-access = write
password-db = passwd
realm = Mi Repositorio
```

Donde passwd es el nombre del fichero donde se dan de alta los usuarios que tendrán permiso de escritura en el repositorio. Oj con los permisos de este fichero, ya que las claves se almacenan en texto claro. En ningún caso debería tener permiso de lectur para el grupo ni para "others".

Donde realm define el prompt que verá el usuario para autenticarse, y también se usa por Subversion como identificador del "dominio" de autenticación.

6. Trabajando con los proyectos

6.1. Estructura de los proyectos

Los proyectos seguirán la estructura habitual de Subversion, es decir tendrán los siguientes tres directorios:

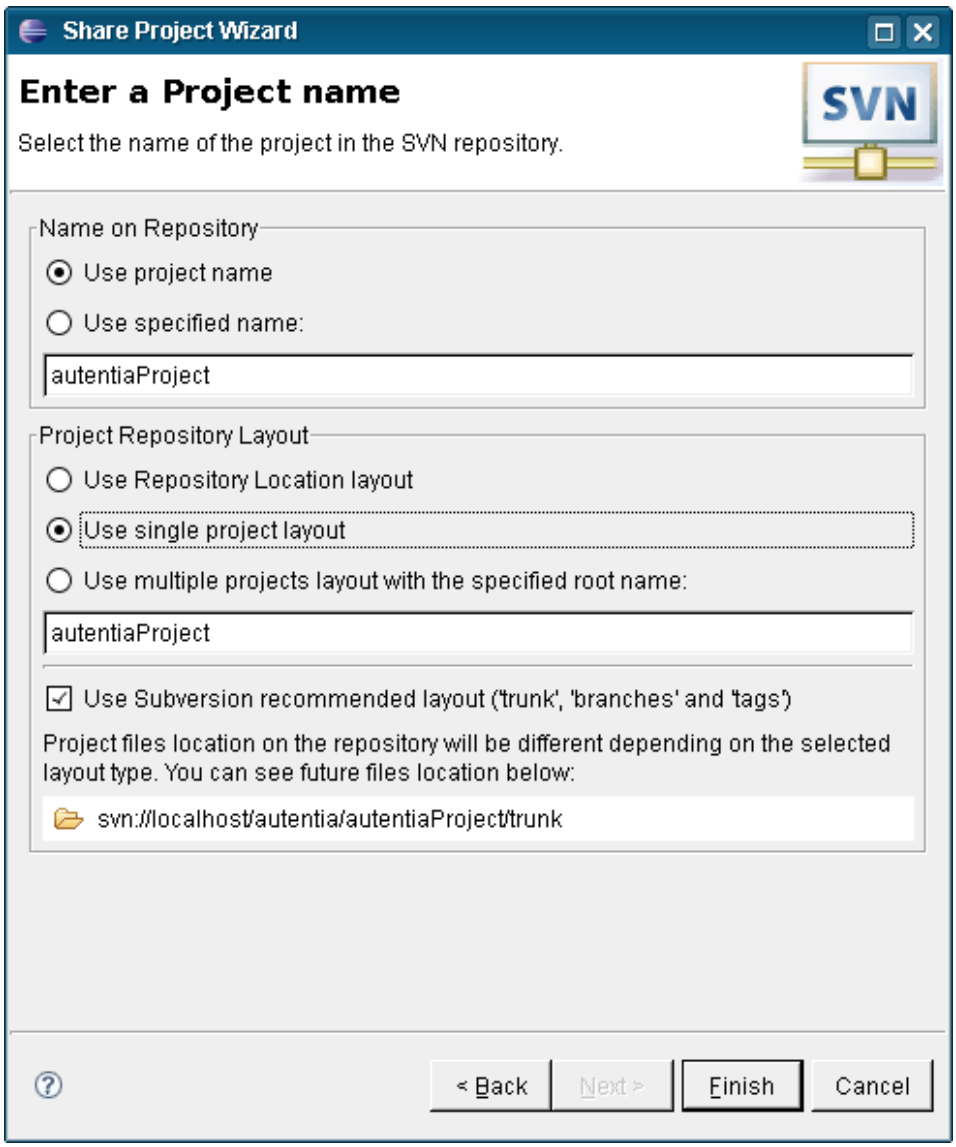
- **trunk**: Directorio con el HEAD del desarrollo. Es donde se realiza el trabajo del día a día.
- **branches**: Directorio donde se almacenan las distintas ramas (branches) que se crean a partir del trunk. La política para crear branches será "The Branch-When-Needed system". Descrita en el documento de buenas prácticas de subversión http://svn.collab.net/repos/svn/trunk/doc/user/svn-best-practices.html.
- **tags**: Directorio donde se almacena copia del repositorio con un nombre de etiqueta concreto. Esto no es realmente necesario porque los números de revisión en Subversion se aplican a todo el repositorio, por lo que sabiendo el número d revisión se tiene un "snapshot" del repositorio en ese momento. Este directorio existe por conveniencia para dar nombre más legibles de cara al "humano", por ejemplo v1.0.3. En este directorio nunca se debería hacer commit.

Según esto la ruta al trunk de un proyecto sería:

```
svn://nombreCliente/nombreProyecto/trunk
```

Es decir, los directorios trunk, branches, y tags, están dentro de cada proyecto.

Si se está usando como cliente el plugin de Eclipse Subversive, cuando se hace el "Share" del proyecto es importante especifica la opción "**Use single project layout**" para que los directorios se creen correctamente. A continuación se muestra la ventana d opciones mencionada:



6.2. Haciendo commit en el repositorio

En Subversion los commit son transaccionales. Es decir, si hacemos commit de cambios en varios ficheros, en el repositorio se guardarán todos los cambios o ninguno si se produce algún error en mitad de la operación.

Cada vez que se hace un commit aumenta el numero de revisión del repositorio, sí, del repositorio y no de cada fichero individual. Es decir un número de revisión refleja un snapshot de todo el repositorio en un momento determinado (y no de un sólo fichero como ocurre en el CVS).

Teniendo en cuenta la forma de trabajar del Subversion, expresada en los dos párrafos anteriores, los commit se deben intentar hacer por bloque, en lugar de fichero por fichero. De esta manera se consigue que un número de revisión exprese un cambio concreto (el arreglo de un bug, una nueva feature, una refactorización, ...).

Debería ser obligatorio poner un comentario cada vez que se hace commit en el repositorio. Este comentario debería llevar el identificado (del sistema de issue tracking que se esté usando, como por ejemplo el Scarab) del issue que se está cerrando. Est identificador se podría poner como: #issueId. Es decir, el carácter # seguido del identificador que da el sistema de issue trackin que se esté usando (independientemente de si es numérico o alfanumérico).

También debería ser obligatorio, al cerrar el issue en el sistema de issue tracking, hacer referencia al número de revisión (o la lista de números de revisiones) donde se hicieron los cambios. Se podría poner como rNumeroRevision. Es decir la letra r seguid del número de revisión.

7. Conclusiones

Ya hemos hablado otras veces de herramientas de control de versiones, como el CVS; y desde Autentia (http://www.autentia.com) siempre os hemos recomendado su uso.

Nuevamente repetimos esa recomendación. Es fundamental usar un sistema de control de versiones para tener controlados nuestros desarrollos y su correcta evolución. En esta ocasión Subversion nos ofrece una estupenda opción.

En mi trabajo diario en Autentia y en otras empresas me he tenido que enfrentar a la gestión de repositorios CVS, y he tenido que sufrir en propias carnes la carencia que tiene este de no versionar los metadatos. Es por esto que os recomiendo usar Subversion antes que CVS.

Y ya para terminar, os dejo un par de enlaces de obligada lectura, ya que este tutorial no es más que un esbozo de las capacidades de Subversion:

- http://svnbook.red-bean.com/
- http://svn.collab.net/repos/svn/trunk/doc/user/svn-best-practices.html

8. Sobre el autor

Alejandro Pérez García, Ingeniero en Informática (especialidad de Ingeniería del Software)

Socio fundador de Autentia (Formación, Consultoría, Desarrollo de sistemas transaccionales)

mailto:alejandropg@autentia.com

Autentia Real Business Solutions S.L. - "Soporte a Desarrollo"

http://www.autentia.com

Puedes opinar sobre este tutorial aquí

Nuevo servicio de notificaciones

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales, inserta tu dirección de correo en el siguiente formulario.

Subscribirse a Novedades	
e-mail	
	Enviar

Otros Tutoriales Recomendados (También ver todos)

Nombre Corto	**Descripción**
Utilizar CVS NT con JDeveloper 10g	Este tutorial enseña cómo emplear la herramienta CVS NT con Oracle Jdeveloper
Acceso seguro a CVS a través de SSH	Os mostramos como segurizar los accesos a CVS a través de SSH, utilizando herramientas gratuitas
Manejo de Repositorios CVS desde Eclipse	En este tutorial os enseñamos a manejar el repositorio CVS desde la plataforma Eclipse
Instalación de Subversion (SVN) en Windows XP	En este tutorial os mostramos cómo instalar y utilizar la herramienta SVN en vuestro entorno XP
Repositorio CVS en Windows	Os mostramos como montar un servidor para el control de versiones CVS en Windows asi como acceder a él a través de WinCVS