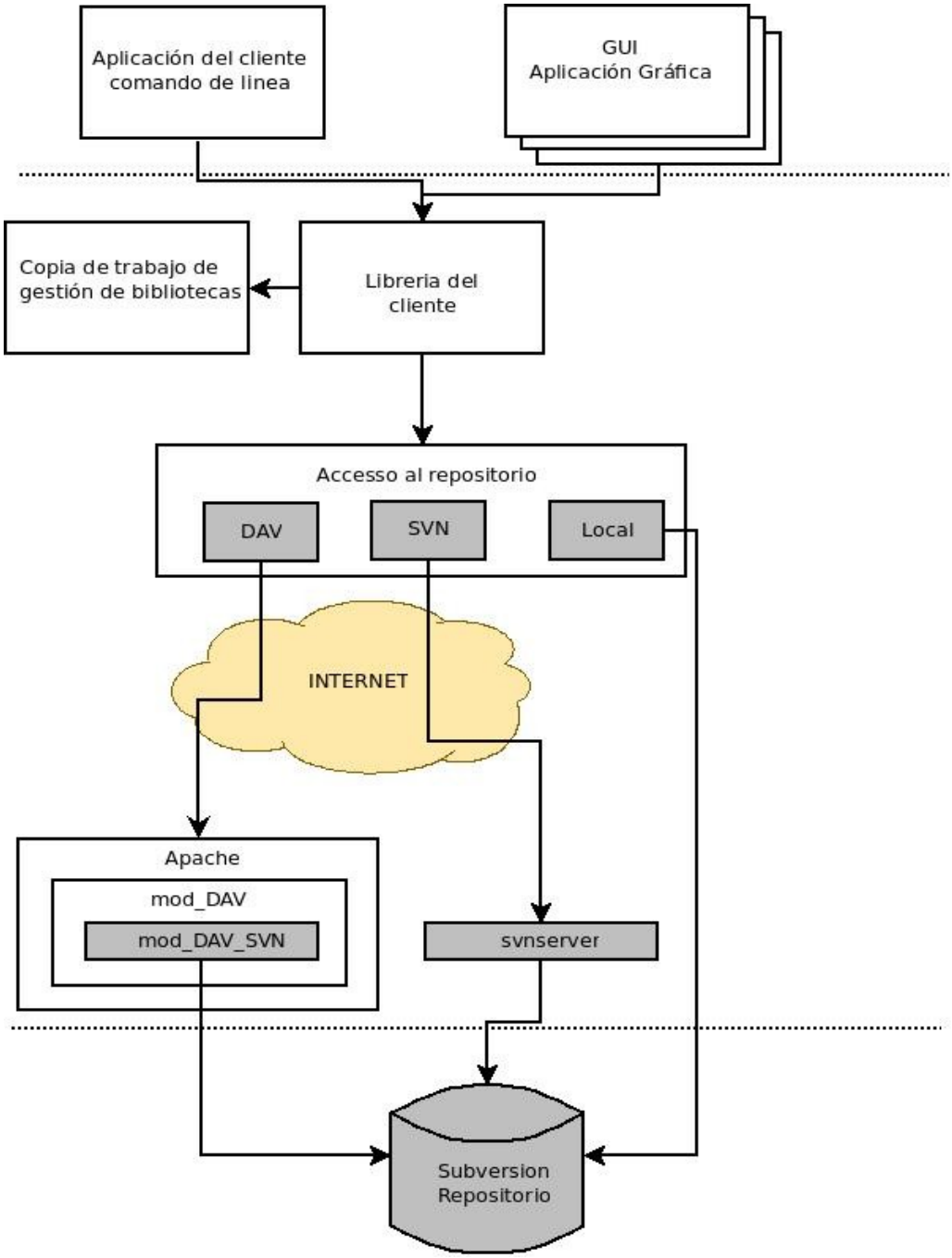
Subversión

¿ Qué es Subversión ?

Es un sistema de control de versiones libre y de código fuente abierto. Maneja tanto ficheros como directorios a través del tiempo. Hay un árbol de ficheros en un repositorio central. El repositorio es como un servidor de ficheros ordinario, excepto porque recuerda todos los cambios hechos a sus ficheros y directorios.



Cliente para Windows

http://tortoisesvn.net/

Paquetes para GNU/Linux

```
# apt-get install subversion libapache2-svn  subversion-tools apache2 openssl apache2-utils
```

Websvn

```
# apt-get install websvn enscript
```

enscript : converts text to Postscript, HTML or RTF with syntax highlighting.

Componentes de Subversión

Dentro de subversión trae varios programas :

- **svn** = El programa cliente de línea de comandos.
- **svnversion** = Programa para informar del estado (en términos de revisiones de los elementos presentes) de una copia de trabajo.
- **svnlook** = Una herramienta para inspeccionar un repositorio de Subversión.
- **svnadmin** = Herramienta para crear, modificar o reparar un repositorio de Subversión.
- **svndumpfilter** = Un programa para filtrar el formato de salida de volcado de repositorios Subversión.
- **mod_dav_sun** = Un módulo para servidor HTTP Apache usado para hacer que su repositorio esté disponible a otros a través de una red.
- **svnserve** = Un servidor independiente, ejecutable como proceso demonio o invocable por SSH; otra manera de hacer que su repositorio esté disponible para otros a través de una red.

Estructura de proyectos

Los proyectos seguirán la estructura habitual de Subversión, 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.
- **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 de revisión se tiene un

"snapshot" del repositorio en ese momento. Este directorio existe por conveniencia para dar nombres 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

Acceso al Repositorio

Esquema	Método de acceso
file:///	Acceso directo al repositorio (en disco local).
http://	Accesso vía protocolo **WebDAV** a un servidor Apache que entiende de Subversión.
https://	Igual que **http://**, pero con cifrado SSL.
svn://	Acceso vía un protocolo personalizado a un servidor **svnserver**.
snv+ssh://	Igual que **svn://**, pero a través de un túnel SSH.

Creación de los directorios

```
# mkdir /svn
# svnadmin create /svn/Proyecto1
# svnadmin create /svn/General
# chmod -R 770 /svn/Proyecto1 /svn/General
# chown -R www-data:www-data /svn/Proyecto1
# chown -R www-data:www-data /svn/General
```

Por un error que no da windows o GNU/Linux tenemos que hacer lo siguiente :

```
# chmod -R g+w /svn/Proyecto1
# chmod -R g+w /svn/General
```

Creamos nuestro directorio de contenidos para ser importado

Si tenemos el siguiente contenido de programas de desarrollo :

```
# ls ~/proyectos

    GPSGPRSComm 6.0L
    GPSGPRSPos 8.27L 2010
    GPSGPRSPrinc 17.0L
```

Para importarlo hacemos lo siguiente :

```
# svn import -m "Importando GPSGPRSComm-6.0L" ~/proyectos/GPSGPRSComm-6.0L file:///svn/Proyectos1/GPSGPRSComm 6.0L/trunk
```

Y así por cada uno importamos los datos.

Ahora podemos borrar el directorio ~/proyectos/GPSGPRSComm-6.0L ya no nos hace falta, ya que importamos por **svn** para poder trabajarlo.

Si queremos listar lo que contiene nuestro sistema versionado ejecutamos :

```
# svn list http://127.0.0.1/svn
```

Para comenzar a manipular los datos del repositorio, necesitamos crear una copia de trabajo de los datos, una especie de entorno de trabajo privado. Para eso le indicamos a **svn** que realice dicha copia de la rama **trunk**.

```
# svn checkout  http://127.0.0.1/svn projectos
```

si tenemos un usuario se lo indicamos poniendo : --username NOMBRE_USUARIO

```
# svn checkout --username NOMBRE_USUARIO  http://127.0.0.1/svn projectos
```

Ahora disponemos de una copia personal de parte del repositorio en un nuevo directorio llamado **projectos**. Podemos editar los ficheros en su copia de trabajo y después depositar esos cambios de nuevo en el repositorio.

- Entrar en la copia de trabajo y editar los contenidos de los ficheros.
- Ejecutar **svn diff** para ver las diferencias introducidas por sus cambios en formato **diff** unificado.
- Ejecutar **svn commit** para depositar la nueva versión de su fichero en el repositorio.
- Ejecutar **svn update** para “sincronizar” su copia de trabajo con el repositorio.

Listar los proyectos en el svn

```
# svn list --username USUARIO https://127.0.0.1/svn
```

Elevar las modificaciones

Una vez que modificamos un archivo tenemos que elevarlo al repositorio y dejamos un comentario de las modificaciones que realizamos.

```
# svn commit -m “comentario” ARCHIVO
```

Actualizar nuestro area de trabajo

Supongamos que alguien modifico un archivo para nosotros obtener las modificaciones que otros hicieron realizamos un **update**.

```
# svn update
```

Ejemplo :

```
U foo.c
U bar.c
Updated to revision 2.
```

- **U foo.c** = El fichero foo.c fue **Updated** (recibidos cambios del servidor).
- **A foo.c** = El fichero o directorio foo.c fue **Added** a su copia de trabajo local.
- **D foo.c** = El fichero o directorio foo.c fue **Deleted** de su copia de trabajo local.
- **R foo.c** = El fichero o directorio foo.c fue **Replaced** en su copia de trabajo local; esto es, foo.c fue borrado, y un nuevo objeto con el mismo nombre fue añadido. Mientras pueden tener el mismo nombre, el repositorio los considera objetos distintos con historiales distintos.
- **G foo.c** = El fichero foo.c recibió nuevos cambios del repositorio, pero su copia local del fichero tenía sus modificaciones. O los cambios no se tocaron, o los cambios eran exactamente iguales que sus modificaciones locales, así que Subversión ha **merGed** satisfactoriamente los cambios del repositorio al fichero sin ningún problema.
- **C foo.c** = El fichero foo.c recibió cambios **Conflicting** del servidor. Los cambios del servidor directamente se superpusieron sobre sus propios cambios en el fichero. Aunque no hay necesidad de aterrarse. Esta superposición necesita ser resuelta por un humano (usted); tratamos esta situación más tarde en este capítulo.

Ver el estado del archivo

Puede obtener el estado de un archivo para saber si esta modificado,etc.

```
# svn status
```

La primera columna dice el estado de un fichero o directorio y/o su contenido.

- **A archivo_o_directorio** = El **archivo_o_directorio** ha sido programado para la adición en el repositorio.
- **C archivo** = El **archivo** está en un estado de conflicto. Esto es, los cambios recibidos del servidor durante una actualización se solapan con cambios locales que usted tiene en su copia de trabajo. Debe resolver este conflicto antes de enviar sus cambios al repositorio.
- **D archivo_o_directorio** = El **archivo_o_directorio** ha sido programado para la supresión del repositorio.
- **M archivo** = El contenido del fichero archivo ha sido modificado.
- **X directorio** = El **directorio** está sin versionar, pero está relacionado con una definición externa de Subversión. Para descubrir más acerca de definiciones externas.

- **? archivo_o_directorio** = El **archivo_o_directorio** no está bajo control de versiones. Puede silenciar la marca de pregunta pasando la opción **--quiet** (**-q**) a **svn status**, o poniendo la característica **svn: ignore** en el directorio padre. Para más información sobre ficheros ignorados.

- **! archivo_o_directorio** = El **archivo_o_directorio** está bajo el control de versiones pero falta o está de alguna manera incompleto. El objeto puede faltar si se ha borrado usando un comando ajeno a Subversión. En el caso de un directorio, puede estar incompleto si ha interrumpido una descarga o una actualización. Un rápido **svn update** responderá el fichero o el directorio desde el repositorio, o **svn revert file** restaurará un archivo que falta.

- **~ archivo_o_directorio** = El **archivo_o_directorio** está en el repositorio como un tipo de objeto, pero actualmente está en su copia de trabajo como otro tipo. Por ejemplo, Subversión puede tener un fichero en el repositorio, pero usted borró el fichero y creó un directorio en su lugar, sin usar los comandos **svn delete** o **svn add**.

- **I archivo_o_directorio** = Subversión está "ignorando" el archivo_o_directorio, probablemente porque usted se lo dijo. Observe que este símbolo solo aparece si le pasa la opción **--no-ignore** a **svn status**.

La segunda columna dice el estado de las propiedades de un fichero o un directorio. Si aparece una **M** en la segunda columna, entonces las propiedades han sido modificadas, si no es **un espacio en blanco** será impreso.

La tercera columna solo mostrará un espacio en blanco o una **L** la cual significa que Subversión ha bloqueado el objeto en el área de trabajo **.svn**. Usted verá una **L** si ejecuta **svn status** en un directorio donde un **svn commit** esté en progreso quizás cuando esté editando el informe de cambios. Si Subversión no se está ejecutando, entonces probablemente Subversión fue interrumpido y el bloque necesita ser eliminado ejecutando **svn cleanup**.

La cuarta columna solo mostrará un espacio en blanco o un **+** el cual significa que el fichero o directorio está programado para ser añadido o modificado con historial adicional adjunto. Esto ocurre típicamente cuando usted **svn move** o **svn copy** un fichero o directorio. Si usted ve **A + ,** esto significa que el objeto está programado para la adición-con-historial. Este puede ser un fichero, o la raíz de un directorio copiado. + significa que el objeto es parte de un subárbol programado para la adición-con-historial,p.e.algún padre fue copiado,M + significa que el objeto es parte de un subárbol programado para la adición-con-historial, y este tiene modificaciones locales. Cuando envíe los cambios, primero el padre será añadido-con-historial (copiado), lo que significa que este fichero existirá automáticamente en la copia. Entonces las modificaciones locales serán enviadas al repositorio.

La quinta columna solo mostrará **un espacio en blanco** o una **S**. Esto significa que el fichero o directorio ha sido movido de la ruta del resto de la copia de trabajo (usando **svn switch**) a una rama.

```
# svn status –verbose
```

o

```
# svn status -v
```

También puedo mostrar los updates :

```
# svn  status --show-updates –verbose
```

Ver diferencias

Puedo ver la diferencia del archivo que estoy modificando con el de la ultima version comiteada.

```
# vi archivo_1.c
# svn diff
        Index: archivo_1.c
        ===================================================
        --- archivo_1.c (revisión: 3)
        +++ archivo_1.c (copia de trabajo)
        @@ -1 +1,3 @@
          #include <stdio.h>
        +
         +#include <mio.h>
```

Las lineas quitadas se muestran con un - (menos) y las agregadas con un + (mas).

Con esta salida podemos armar un parche.

```
# svn diff > archivo_patch.patch
```

Revertir los cambios

Supongamos que estamos editando un archivo y nos damos cuenta que lo que estamos corrigiendo no es lo correcto son **svn revert** podemos volver los cambios que hicimos.

```
# svn revert archivo_1.c
```

Otro ejemplo :

```
# vi Nuevo-archivo.c
# svn starus Nuevo-archivo.c
   ?      Nuevo-archivo.c

# svn add Nuevo-archivo.c
# svn revert Nuevo-archivo.c
```

Tanto los comandos **svn status**, **svn diff** y **svn revert** pueden ser usados sin ningún acceso de red. Esto lo hace sencillo para administrar sus cambios-en-progreso cuando usted está en alguna parte sin una conexión de red, tal como viajando en un avión, montando en un tren o hackeando en una playa.

Subversión hace esto manteniendo almacenes privados de versiones prístinas de cada fichero versionado dentro del área administrativa **.svn**.

Resolver conflictos (fusionando los cambios de otros)

Con **svn status -u** puede predecir conflictos.

Informacion del archivo o información de la ruta

svn info

Ruta: .
URL: https://127.0.0.1/svn/Proyecto_dia%202
Raíz del repositorio: https://svn.mde.com.ar/svn
UUID del repositorio: fe22f13d-7b99-44b2-b20e-70a907bcf685
Revisión: 96
Tipo de nodo: directorio
Agendado: normal
Autor del último cambio: mauro
Revisión del último cambio: 94
Fecha de último cambio: 2011-07-05 15:06:16 -0300 (mar 05 de jul de 2011)

svn info --username USUARIO NOMBRE_ARCHIVO

Ejemplo :

svn info 1wire.c

Ruta: 1wire.c
Nombre: 1wire.c
URL: https://127.0.0.1/svn/GPSGPRSComm-6.0L/trunk/1wire.c
Raíz del repositorio: https://svn.mde.com.ar/svn
UUID del repositorio: fe22f13d-7b99-44b2-b20e-70a907bcf685
Revisión: 30
Tipo de nodo: archivo
Agendado: normal
Autor del último cambio: mrusso
Revisión del último cambio: 30
Fecha de último cambio: 2011-06-30 15:44:19 -0300 (jue 30 de jun de 2011)
Suma de verificación: e37979f924d50f3d2f5689d511942595
Archivo base previo del conflicto: 1wire.c.r20
Archivo de la copia de trabajo previo del conflicto: 1wire.c.mine
Archivo base del conflicto: 1wire.c.r30

Palabras clave de la revisión

El cliente de Subversión entiende un número de palabras claves de la revisión. Estas palabras claves pueden ser usadas en vez del número entero como argumento a la opción –revision, y son resueltos como números específicos de revisión por Subversión.

- **HEAD** = La última revisión del repositorio.
- **BASE** = La revisión “prístina” de un elemento en la copia de trabajo.
- **COMMITTED** = La última revisión en la que un elemento cambió antes (o en) **BASE**.
- **PREV** = La revisión justo anterior de la última revisión en la cual un elemento cambió. (Técnicamente **COMMITTED – 1**).

Mostrar los cambios realizados de un archivo viendo los logs

Nos ubicamos dentro del directorio de trabajo.

```
# svn log --username USUARIO NOMBRE_ARCHIVO
```

Puedo obtener informacion de una revision determinada :

```
# svn log -r 71 --username USUARIO NOMBRE_ARCHIVO
```

Puedo obtener información de un rango de revisión determinada :

```
# svn log -r 71:79 USUARIO NOMBRE_ARCHIVO
```

Puedo obtener información de un rango de revisión determinada, para mas información lo veo poniendo la opción --**version** (**-v**):

```
# svn log -r 71:79 -v USUARIO NOMBRE_ARCHIVO
```

Mostrar las diferencias entre versiones

Nos ubicamos dentro del directorio de trabajo.

```
# svn diff -r 16:6 --username USUARIO NOMBRE_ARCHIVO
```

Para ver el contenido de una version vieja

Con el comando **svn cat** puedo ver una versión determinada :

```
# svn cat -r 79 mi_archivo.txt
```

Listar un revisión determinada

```
# svn list -r 79
```

Agregar un archivo al repositorio

Para agregar un archivo usamos **svn add ARCHIVO**

```
# vi archivo_1.c
        ….

# svn add archivo_1.c
```

Luego para que quede elevado tengo que hacerle un **commit** y poner un comentario.

```
# svn commit -m “Es un ejemplo de agregado”
```

Realizar copia

Para realizar una copia de un archivo del repositorio a otro, esto crea un nuevo objeto.

```
# svn copy archivo_1.c archivo_2.c
# vi archivo_2.c
        …
# svn commit -m “Copia de archivo_1.c a archivo_2.c”
```

Crear directorio

```
# svn mkdir mi-directorio
```

Cambiando la copia local de trabajo

Con el comando **svn switch** transforma una copia local de trabajo existente en una rama diferente.

```
# svn mkdir https://127.0.0.1/svn/Proyecto/branch
# svn switch https://127.0.0.1/svn/Proyecto/branch
```

Esto lo que hace es cambiarme de ruta.

Mover

Esto funciona igual que **svn copy archivo_1.c** ; **svn delete archivo_1.c**

```
# svn move archivo_1.c archivo_2.c
        A       archivo_2.c
        D       archivo_1.c

# vi archivo_2.c
        …
# svn commit -m “Movie de archivo_1.c a archivo_2.c”
```

Borrando un archivo

Puedo borrar un archivo del repositorio.

```
# svn delete archivo_1.c
```

Luego para que quede elevado tengo que hacerle un **commit** y poner un comentario.

```
# svn commit -m “Es un ejemplo de borrado”
```

Copiando cambios específicos

Podemos ver la diferencia entre las versiones y generar una nueva version.

```
# svn diff -r 88:89 Mauro_3.txt

        Index: Mauro_3.txt
        ===================================
        --- Mauro_3.txt (revisión: 88)
        +++ Mauro_3.txt (revisión: 89)
```

```
@@ -1 +1,3 @@
 Esto es una prueba
+
+Prueba 2 Marcos
```

```
# svn merge -r 88:89 https://127.0.0.1/svn/Proyecto2
U    Mauro_3.txt
```

```
# svn status
M    Mauro_3.txt
```

```
# svn commit -m “Mauro_3.txt: pasado a version 89”
```

Con esto genera una nueva versión de Mauro_3.txt siendo la 90.

Realizar un backup

```
# svnadmin dump /svn/Proyecto1 > respaldo-proyecto1
```

```
# file respaldo-proyecto1

    respaldo-proyecto1: Subversion dumpfile (version: 2)
```

Para restaurar :

```
# svnadmin load /svn/Proyecto1 < respaldo-proyecto1
```

Estructura del repositorio

La mayoría de las personas crean un directorio **trunk** que contendrá la “**linea principal**” de desarrollo, un directorio **branches** que contendrá **copias de las ramas**, y un directorio **tags** que contendrá **copias de las etiquetas**.

```
/trunk
/branches
/tags
```

Si un repositorio contiene múltiples proyectos, los administradores normalmente indexan la estructura por proyecto.

```
/paint/trunk
/paint/branches
/paint/tags
```

Configuración de websvn

Podemos reconfigurar en debian mediante dpkg-reconfigure :

```
# dpkg-reconfigure websvn
```

o bien tocando los archivos de configuración :

```
# vi /etc/websvn/svn_deb_conf.inc

<?php
// please edit /etc/websvn/config.php
// or use dpkg-reconfigure websvn

// php is default correctly colourized
$extEnscript[".java"] = "java";
$extEnscript[".pl"] = "perl";
$extEnscript[".py"] = "python";
$extEnscript[".sql"] = "sql";
$extEnscript[".java"] = "java";
$extEnscript[".html"] = "html";
$extEnscript[".xml"] = "html";
$extEnscript[".thtml"] = "html";
$extEnscript[".tpl"] = "html";
$extEnscript[".sh"] = "bash";

// Number of spaces to expand tabs to in diff/listing view across all repositories
$config->expandTabsBy(8);

$config->parentPath("/svn/");
$config->setEnscriptPath("/usr/bin");
$config->setSedPath("/bin");
$config->useEnscript();
```

Configuración de Apache2/SSL

Creamos el certificado :

```
# mkdir /etc/apache2/ssl
# RANDFILE=/dev/random openssl req $@ -new -x509 -days 365 -nodes \
   -out /etc/apache2/ssl/apache.pem -keyout /etc/apache2/ssl/apache.pem
# chmod 600 /etc/apache2/ssl/apache.pem
```

Creamos la asignación de la pagina

```
# vi /etc/apache2/sites-available/svn

 <VirtualHost *:80>
      RewriteEngine on
      RewriteCond %{SERVER_PORT} !^443$
      RewriteRule ^(.*) https://%{SERVER_NAME}/websvn/ [L,R,Nc]
 </VirtualHost>


# vi /etc/apache2/sites-available/svn-ssl

NameVirtualHost *:443
<VirtualHost *:443>
```

```
        DocumentRoot /usr/share/websvn
        ServerAdmin webmaster@localhost

        #   SSL Engine Switch:
        #   Enable/Disable SSL for this virtual host.
        SSLEngine on

        SSLCertificateFile /etc/apache2/ssl/apache.pem

        # Configuration for websvn using php4.

        Alias /websvn /usr/share/websvn

        <Directory /usr/share/websvn>
          DirectoryIndex index.php
          Options FollowSymLinks
          Order allow,deny
          Allow from all
          <IfModule mod_php4.c>
            php_flag magic_quotes_gpc Off
            php_flag track_vars On
          </IfModule>
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/svn-error-ssl.log

        # Possible values include: debug, info, notice, warn, error, crit,
        # alert, emerg.
        LogLevel warn

        CustomLog ${APACHE_LOG_DIR}/svn-access-ssl.log combined

        <Location />
          AuthType Basic
          AuthName "My Subversion Repository"
          AuthUserFile /etc/apache2/dav_svn.passwd
          Require valid-user
        </Location>
    </VirtualHost>
```

```
# vi /etc/apache2/httpd.conf

    Servername 127.0.0.1
```

Creamos el archivo de password dav_svn.passwd

```
# htpasswd -c /etc/apache2/dav_svn.passwd usuario1
```

Activamos el nuevo sitio. Esto lo hacemos realizando un "sym-link" a la configuración de etc/apache2/sites-enabled/. El comando que nos brinda apache para realizar esto, es:

```
# a2ensite svn
```

```
# a2ensite svn-ssl
```

Para activar el módulo ssl y rewrite , hacemos:

```
# a2enmod ssl
# a2enmod rewrite
```

Reiniciamos el servidor de apache :

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