

Subversion (SVN)

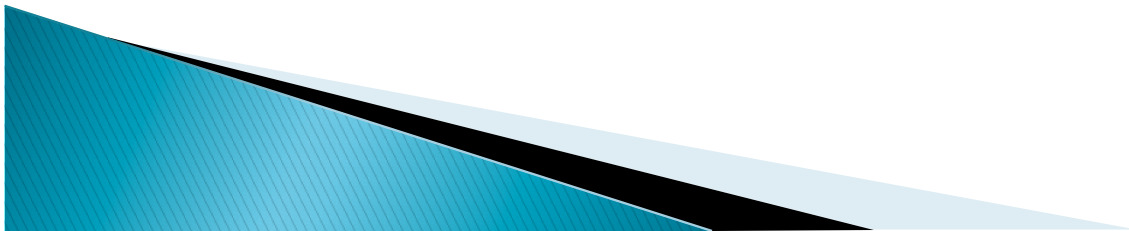
- . Sistema de Control de Versiones
- . Sucesor de CVS

Carlos Armas
Hervey Allen



Contenido

- Qué es control de versiones?
- introducción a SVN
- Principios
- Diferencias con CVS
- Comandos
- Ejemplos
- Configuración y acceso a un repositorio



Qué es control de versiones?

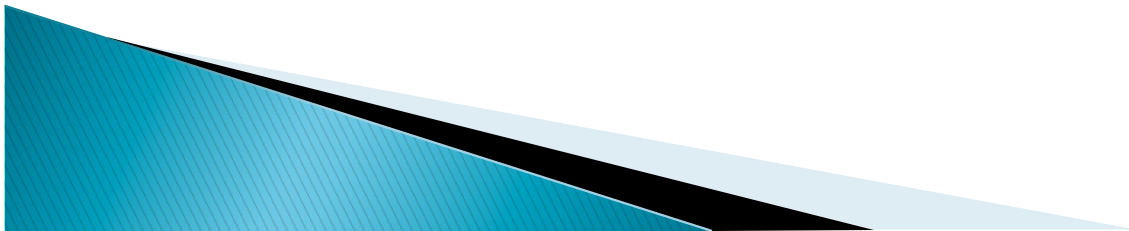
Tres principios básicos:

- Mantener un registro e historia de cambios
- Dar acceso publico a la información
- Mantener diferentes versiones de un mismo conjunto de datos

Qué tipo de datos ?

Código fuente,

- Documentación
- Ficheros de configuración
- En general, cualquier dato



Terminología

- ▶ repositorio

- Copia central de todos los ficheros bajo control, estructurado en árbol de directorios

- ▶ Copia de trabajo

- Copia local de los datos, que puede ser cambiada, en sincronización con el repositorio. Contiene información especial para la interacción con el mismo

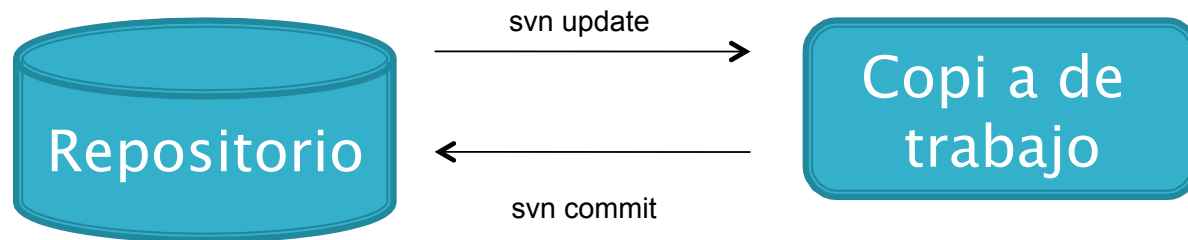
- ▶ Revision

- Un grupo de directorios y ficheros que reflejan el estado del repositorio en un determinado momento



Principios

- ▶ El repositorio es la copia principal
- ▶ Todo el trabajo se hace en la copia de trabajo
- ▶ Los cambios se reflejan (“materializan”) en repositorio (comando *commit*)



Control de cambio, estados

- ▶ Sin cambio y actualizado
 - Copia idéntica al repositorio
 - A *commit* or *update* no hace nada
- ▶ Cambio local y actualizado
 - Copia local cambió, y repositorio no ha recibido cambios de otros contribuyentes
 - *Commit* actualiza repositorio, *update* no hace nada
- ▶ Sin cambio y desactualizado
 - Copia local no ha cambiado, pero repositorio cambió
 - *Update* cambia estado local, *commit* no funciona
- ▶ Cambio local y desactualizado
 - Conflicto! Se necesita un *update*
 - Si SVN no puede resolver automáticamente, se necesita resolución manual



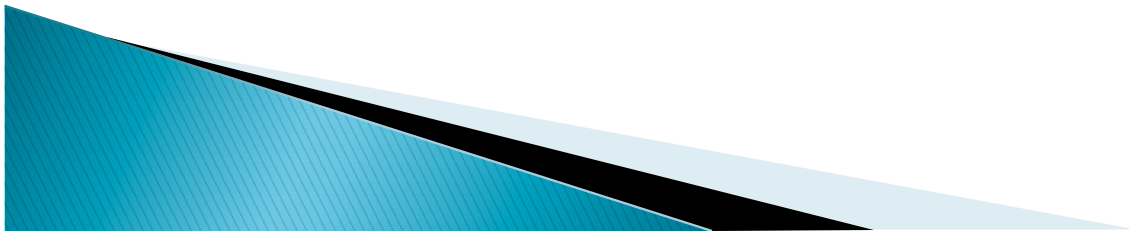
Ejemplo de interacción

▶ Extracción inicial

- `svn checkout <proyecto>`
- `vi <mifichero.conf>` (...cambios ...)
- `svn commit <mifichero.conf>` (*reflejar cambios*)

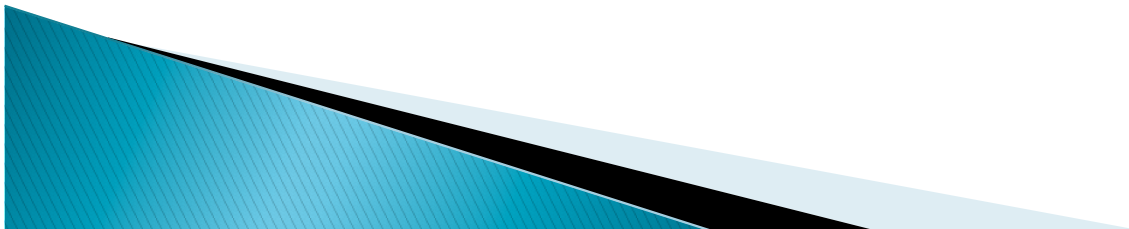
▶ Más cambios:

- `svn update`
- `vi <mifichero.conf>`
- `svn commit <mifichero.conf>`



SVN el repositorio

- ▶ Clientes acceden localmente, o a través de la red
- ▶ SVNROOT environment variable:
- ▶ SVNROOT=
 - /svn/miproyecto # disco local
 - svn://svnserver/svn/miproyecto # via svnserve
 - svn+ssh:// svnserver/svn/miproyecto # via SSH



Crear un repositorio

▶ Instalar

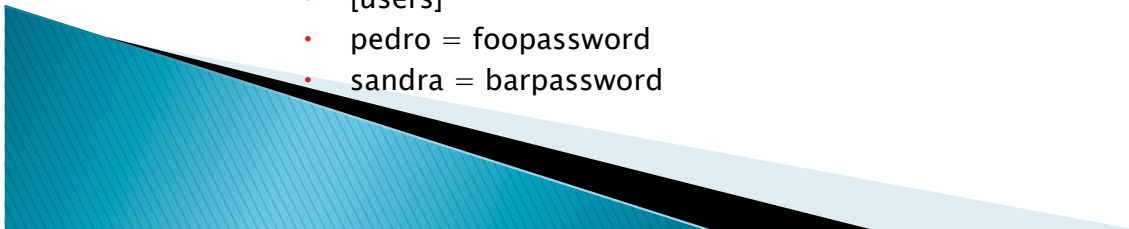
- `#apt-get install subversion`
- `#svncreate <repositorio>`
- Editar `<repositorio>/`

▶ Incluir como “servicio”

- Crear `/etc/init.d/subversion`, que incluya basicamente
 - `svnserve -d -r <repositorio>`
- `#chkconfig --add subversion`
- `#chkconfig --level 2345 subversion on`

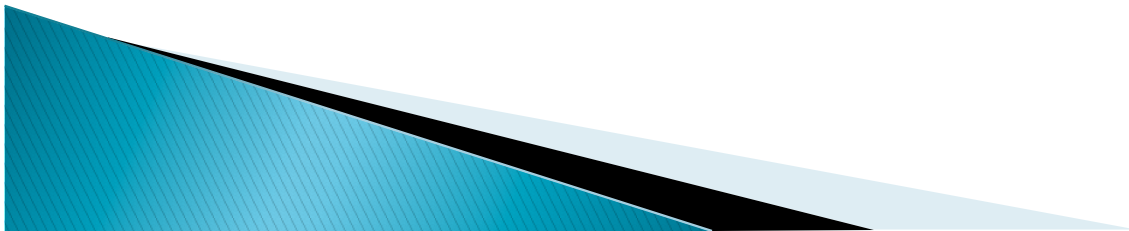
▶ Editar permisos

- Editar `>repositorio>/conf/svnserve.conf`
- Especificar el fichero de passwords:
 - `[general]`
 - `password-db = <userfile>`
 - `realm = example realm`
- Crear usuarios:
 - `[users]`
 - `pedro = foopassword`
 - `sandra = barpassword`



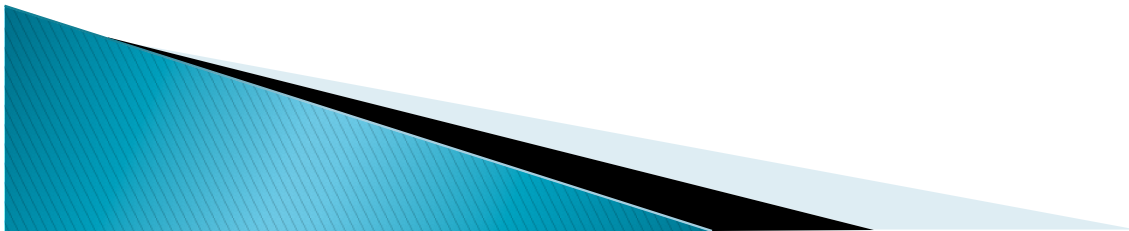
SVN – clientes

- ▶ Existen para varios sistemas operativos
 - svn (UNIX)
 - TortoiseSVN (Windows)
 - ...
- ▶ Acceso local o a través de la red



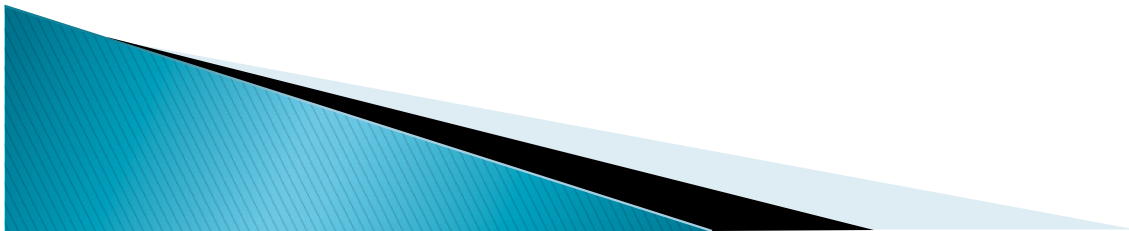
Comandos SVN

- ▶ **import**
 - importa un nuevo proyecto a un repositorio repository
- ▶ **checkout (co)**
 - Copia del repositorio al directorio local
- ▶ **update (up)**
 - Actualiza copia local a partir del repositorio
- ▶ **add**
 - Añade un nuevo fichero o directorio a la copia local
- ▶ **delete**
 - Remueve un fichero de la copia local
- ▶ **commit**
 - Actualiza repositorio a partir de ficheros locales



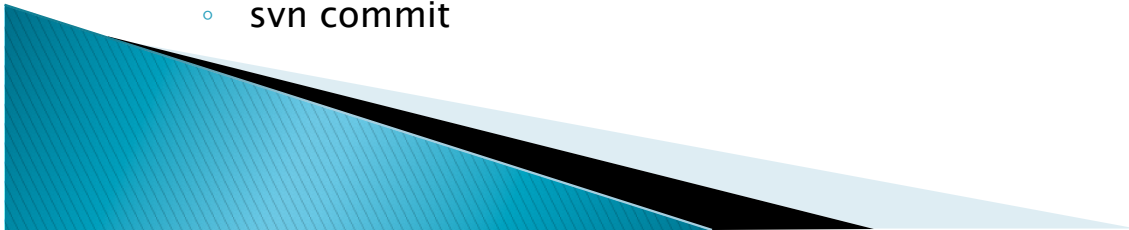
Otros comandos utiles

- ▶ **mkdir**
 - Añade directorio a copia local
- ▶ **status**
 - Estado y version de un fichero
- ▶ **diff**
 - Muestra la diferencia de versiones entre un elemento local y el repositorio
- ▶ **log**
 - Muestra la historia de cambios de uno o mas ficheros
- ▶ Muchos otros: copy, export....



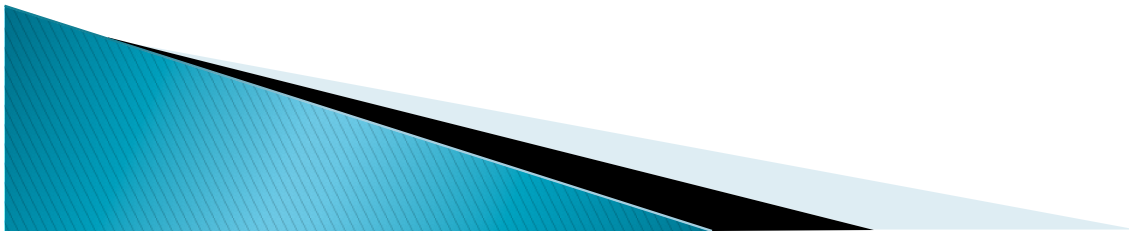
Ciclo de Trabajo

- ▶ Actualiza copia de trabajo
 - svn update
- ▶ Introduce cambios
 - svn add
 - svn delete
 - svn copy
 - svn move
- ▶ Chequea cambios
 - svn status
 - svn diff
 - svn revert
- ▶ Combina con los cambios de otros
 - svn merge
 - svn resolve
- ▶ Completa los cambios
 - svn commit



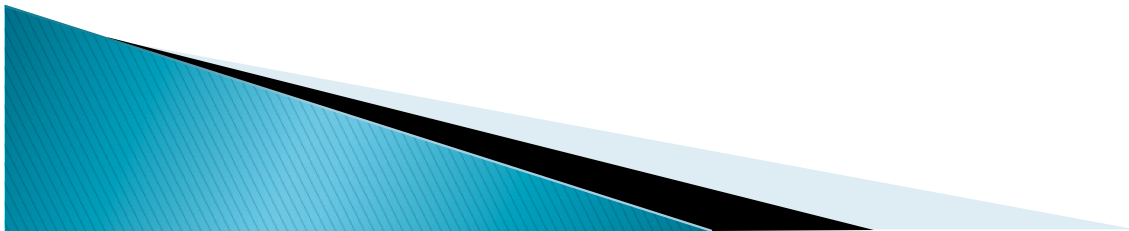
Ventajas, y Diferencias con CVS

- ▶ CVS solamente controla cambios a ficheros
- ▶ SVN crea un sistema de ficheros virtual, que incluye directorios
- ▶ CVS no puede controlar cambios de nombre o copias
- ▶ Como SVN controla directorios, cambios de nombre y copias OK
- ▶ SVN permite Control “atomico” del cambio: o todos los cambios funcionan, o ninguno se acepta
- ▶ CVS no puede proveer semejantes funciones
- ▶ En general, proporciona mayor flexibilidad de acceso, como HTTP via apache, con las consiguientes ventajas



Conclusiones

- ▶ Sofisticado sistema de control de versiones,
- ▶ Muy util para programadores,
- ▶ Para administradores de redes, muchas de las funciones de alto nivel no son necesarias
- ▶ En realidad, tanto CVS como Subversion pueden ser utilizados a nivel de administración de red,
- ▶ Sin embargo no se puede ignorar:
 - La herramienta mas popular es la que mejor soporte recibe,
 - Muchos de nosotros da soporte a equipos de programadores en nuestro trabajo habitual



Referencias

- ▶ “Version Control with Subversion” – O’Reilly
- ▶ Online and free at <http://svnbook.red-bean.com>

