Subversion: Desarrollo colaborativo

Jesús Espino García

Jornadas de Software Libre de la
Universidad Autonoma de Madrid
2007

7 de Marzo de 2007

Introducción

¿Qué es un sistema de control de versiones?

- Sistema para mantener el seguimiento durante el desarrollo.
- Se basa en las revisiones incrementales de los ficheros.
- Permite obtener cualquier versión/revisión en cualquier momento.
- Permite trabajar de forma concurrente a varios desarrolladores.
- Añade información y comentarios al efectuar cambios.
- Es independiente del contenido.
- Puede funcionar en red o localmente.
- Es útil en desarrollos pequeños (de 1 desarrollador) como en grandes (de varios).

¿Qué no es?

- No es un compilador (gcc, g++, javac, mcs, etc).
- No construye aplicaciones (make, ant, jam, etc).
- No elimina el coordinador de proyecto (el jefe).
- No elimina la comunicación entre usuarios (mail, gforge).
- No lleva control de bugs (bugzilla).
- No es un depurador (gdb, ddd).
- No prueba aplicaciones (unit).

Vocabulario

- **Repositorio:** Lugar de almacenamiento de los datos de uno o varios proyectos. Es un directorio en alguna maquina (por ejemplo: /var/lib/svn).
- **Modulo:** Es un directorio especifico del repositorio. Puede identificar una parte del proyecto o ser un proyecto en si.
- **Revisión:** Cada una de las versiones parciales o cambios en los ficheros o repositorio completo. La evolución del sistema de versiones se mide en revisiones. Cada cambio se considera incremental.

Vocabulario

- **Repositorio:** Lugar de almacenamiento de los datos de uno o varios proyectos. Es un directorio en alguna maquina (por ejemplo: /var/lib/svn).
- **Modulo:** Es un directorio especifico del repositorio. Puede identificar una parte del proyecto o ser un proyecto en si.
- **Revisión:** Cada una de las versiones parciales o cambios en los ficheros o repositorio completo. La evolución del sistema de versiones se mide en revisiones. Cada cambio se considera incremental.

Vocabulario

- **Repositorio:** Lugar de almacenamiento de los datos de uno o varios proyectos. Es un directorio en alguna maquina (por ejemplo: /var/lib/svn).
- **Modulo:** Es un directorio especifico del repositorio. Puede identificar una parte del proyecto o ser un proyecto en si.
- **Revisión:** Cada una de las versiones parciales o cambios en los ficheros o repositorio completo. La evolución del sistema de versiones se mide en revisiones. Cada cambio se considera incremental.

Vocabulario

- **Etiqueta:** Información textual que se añade a un conjunto de ficheros (o a un modulo completo) para indicar algún hito importante (por ejemplo: VERSION_0_1).
- **Rama:** Revisiones paralelas de un modulo para efectuar cambios sin tocar la evolución principal. Se puede emplear para pruebas o para mantener cambios en versiones viejas.

Vocabulario

- **Etiqueta:** Información textual que se añade a un conjunto de ficheros (o a un modulo completo) para indicar algún hito importante (por ejemplo: VERSION_0_1).
- **Rama:** Revisiones paralelas de un modulo para efectuar cambios sin tocar la evolución principal. Se puede emplear para pruebas o para mantener cambios en versiones viejas.

Operaciones Básicas

- Introducir datos en el repositorio:

Import Envía la primera copia de un modulo. Sólo se hace una vez.

Commit Manda los cambios locales al repositorio.

- Sacar datos del repositorio:

Check-out Descarga una versión de trabajo a tu maquina. Sólo se hace una vez.

Update Actualiza en la copia local los cambios del repositorio.

Operaciones Básicas

- Introducir datos en el repositorio:

Import Envía la primera copia de un modulo. Sólo se hace una vez.
Commit Manda los cambios locales al repositorio.

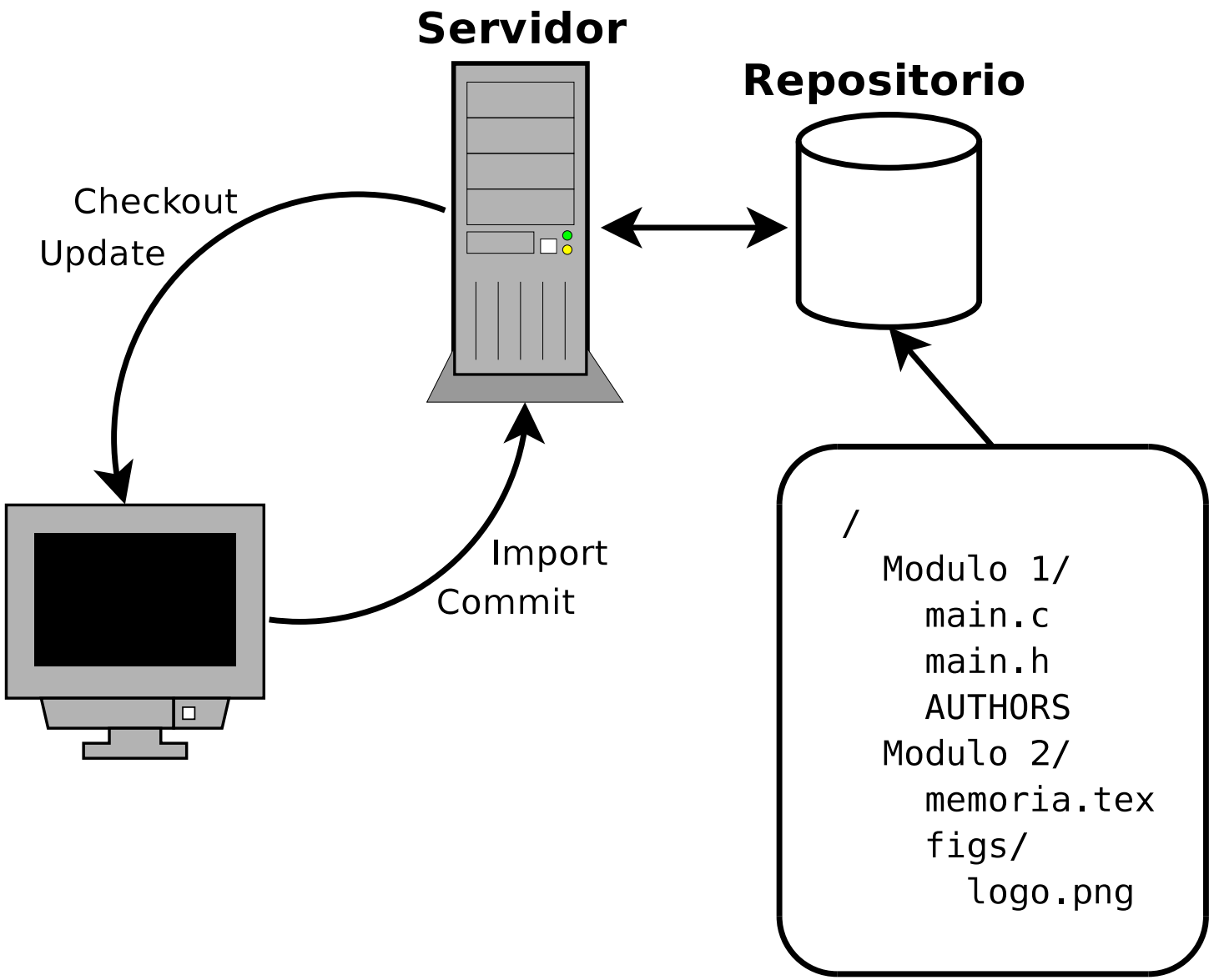
- Sacar datos del repositorio:

Check-out Descarga una versión de trabajo a tu maquina. Sólo se hace una vez.
Update Actualiza en la copia local los cambios del repositorio.

Comentarios

- Cada operación debe comentarse.
- Indica que problema resuelve o que característica añade.
- Sirven para informar al resto de los usuarios y para el seguimiento de las versiones.

Escenario


Servidor
Repositorio
Checkout
Update
Import
Commit
/
Modulo 1/
main.c
main.h
AUTHORS
Modulo 2/
memoria.tex
figs/
logo.png

Subversion

Subversion

- Subversion es un sistema de control de versiones.
- Muy parecido al conocido CVS.
  - Aparece para suplir las limitaciones de este.
- Esta empezando a ser adoptado por muchos desarrolladores libres.
- Mantiene un repositorio centralizado.
- Permite múltiples desarrolladores.
- Es independiente del contenido.
- Admite ficheros binarios sin problema.

Características

- Las revisiones se hacen sobre todo el árbol de directorios.
  - Varios cambios (cambiar/añadir/borrar) pueden ir dentro de una misma revisión.
- Las diferencias almacenadas son binarias. El tratamiento de los ficheros binarios es igual al de los ficheros de texto.
- La copia/movimiento de ficheros no es una operación costosa.

Tipos de servidores

- Los servidores están identificados por una URL.
- El funcionamiento local o por red es idéntico: solo cambia la URL.
- Protocolos soportados:
  - `file:///` Acceso directo al repositorio (en el disco local).
  - `http://` Acceso vía protocolo WebDAV.
  - `https://` Igual que `http://`, pero con cifrado SSL.
  - `svn://` Acceso vía un protocolo personalizado a un servidor svnserve.
  - `svn+ssh://` Lo mismo que `svn://`, pero a través de un túnel SSH.

Estructura de los módulos

Para simplificar la utilización de etiquetas y ramas, los módulos suelen tener tres directorios básicos:

- **trunk:** Directorio para el código fuente.
- **tags:** Directorio para las etiquetas.
- **branches:** Directorio para las ramas.

Estructura de los módulos

Para simplificar la utilización de etiquetas y ramas, los módulos suelen tener tres directorios básicos:

- **trunk:** Directorio para el código fuente.
- **tags:** Directorio para las etiquetas.
- **branches:** Directorio para las ramas.

Crear un repositorio

```
$ svnadmin create /tmp/svn
```

- El fuente no tiene estructura de modulo, puedes usar svn mkdir.
- Es conveniente tener un modulo por repositorio (Pero no necesario).

Introducir datos

Opción `import`:

```
$ svn import file:///tmp/svn
Adding          trunk
Adding          trunk/fichero2.txt
Adding          trunk/fichero3.txt
Adding  (bin)   trunk/fichero4.bin
Adding          trunk/fichero1.txt
Adding          branches
Adding          tags

Committed revision 1.
```

Bajar copia de trabajo

Opción checkout o co:

```
$ svn co file:///tmp/svn/trunk modulo
A  modulo/fichero2.txt
A  modulo/fichero3.txt
A  modulo/fichero4.bin
A  modulo/fichero1.txt
Checked out revision 1.
```

Realizar cambios

Puedes:

- Editar ficheros.
- Añadir ficheros (`add`).
- Copiar ficheros (`copy`).
- Renombrar ficheros (`move`).
- Borrar ficheros (`delete`).

Ejemplo:

```
$ svn add fichero5.txt
A         fichero5.txt
```

Subir cambios

Opción `commit` o `ci`:

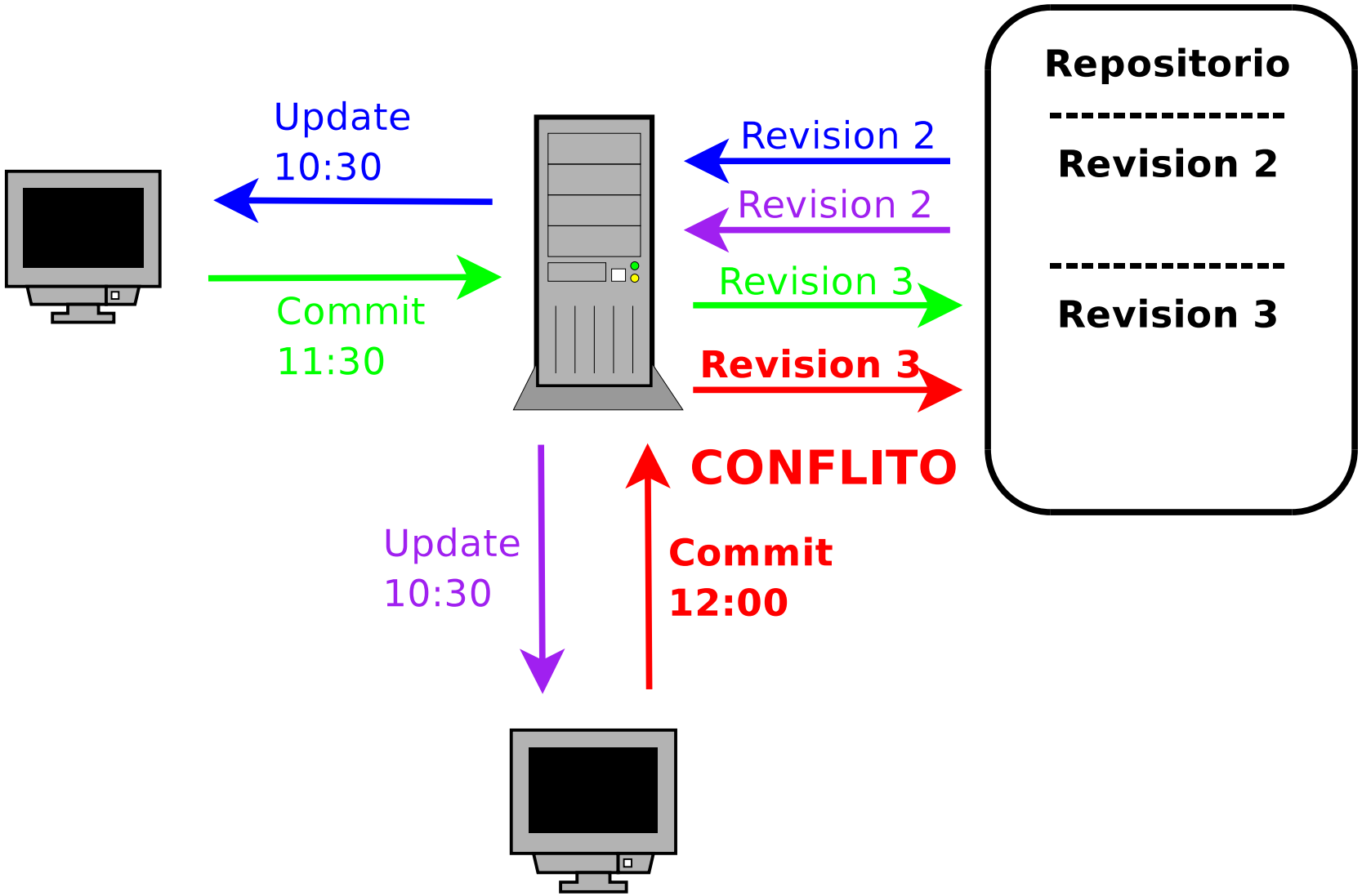
```
$ svn ci
Sending        fichero1.txt
Adding         fichero5.txt
Transmitting file data ..
Committed revision 2.
```

Actualizar copia de trabajo

Opción `update`:

```
$ svn update
U  fichero1.txt
A  fichero5.txt
Updated to revision 2.
```

Conflictos


Repositorio
Revision 2
Revision 3
Update
10:30
Commit
11:30
Revision 2
Revision 2
Revision 3
Revision 3
CONFLITO
Update
10:30
Commit
12:00

Resolución de conflictos

```
$ svn ci
Sending        fichero1.txt
svn: Commit failed (details follow):
svn: Out of date: '/trunk/fichero1.txt' in transaction '6'
$ svn update
C  fichero1.txt
Updated to revision 3.
```

Resolución de conflictos

Primero se edita el fichero:

```
<<<<<<< .mime
Esto es mi cambio local
=======
Este es el cambio en el servidor
>>>>>>> .r3
```

Después se da como resuelto el conflicto:

```
$ svn resolved fichero1.txt
Resolved conflicted state of 'fichero1.txt'
```

Subidos en las ramas

- Para Subversion las ramas y los tags son iguales.
- Funcionan igual que una copia.
- Se copia el directorio trunk a branches/rama o tags/etiqueta.
- Si modificas algo en el nuevo directorio será una rama, en caso contrario una etiqueta.

Subidos en las ramas

```
$ svn copy file:///tmp/svn/trunk file:///tmp/svn/tags/0.1
Committed revision 4.
```

```
$ svnlook tree /tmp/svn | grep /
/
 trunk/
 branches/
 tags/
  0.1/
```

Generar un tarball

```
$ svn export file:///tmp/svn/tags/0.1 version-0.1
A  version-0.1
A  version-0.1/fichero2.txt
A  version-0.1/fichero3.txt
A  version-0.1/fichero5.txt
A  version-0.1/fichero4.bin
A  version-0.1/fichero1.txt
Exported revision 4.
$ tar -zcf version-0.1.tar.gz version-0.1/
```

Keywords

Las keywords son etiquetas que se sustituyen dentro de los ficheros por su valor cuando este es exportado:

- `$LastChangedDate$` o `$Date$`
- `$LastChangedRevision$` o `$Rev$`
- `$LastChangedBy$` o `$Author$`
- `$HeadURL$`
- `$Id$`

Es necesario activarlas:

```
$ svn propset svn:keywords "Date Author" *
```

Operación log

La operación `log` nos permite:

- Ver los comentarios de los cambios sobre un repositorio:

  ```
  $ svn log
  ```

- Ver los comentarios de los cambios que han afectado a un fichero:

  ```
  $ svn log fichero2.txt
  ```

Operación diff

La operación `diff` nos permite:

- Ver los cambios realizados sobre el repositorio desde una revisión:

```
$ svn diff -r 3
```

- Ver los cambios realizados sobre un fichero entre dos revisiones:

```
$ svn diff -r 2:4 fichero1.txt
```

- Ver los cambios realizados entre dos ramas o tags:

```
$ svn diff --old=file:///tmp/svn/tags/0.1 \
    --new=file:///tmp/svn/trunk
```

Otras opciones interesantes

- `status`: Muestra el estado de nuestro repositorio.
- `list`: Lista las entradas del directorio en el repositorio.
- `info`: Muestra la información de nuestro modulo.
- `blame`: Informa sobre quien ha hecho los cambios en un fichero y en que revisión se hicieron.
- `help`: Muestra la ayuda en linea de svn.

Para terminar.

Referencias

- http://svnbook.red-bean.com/
- http://www.sindominio.net/quique/Traducciones/subversion_personal.html

Dudas

. . .

Fin