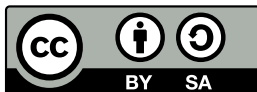


Introducción al Sistema de Control de Versiones Centralizado SVN

Antonio García Domínguez

Universidad de Cádiz



15 de noviembre de 2011

Antes de empezar...

- Estas transparencias están basadas en las de Roberto García Carvajal, usadas en varias ediciones anteriores y disponibles en Wikiinformación (¡gracias!):
`http://osl2.uca.es/wikiinformacion`
- El código fuente de estas transparencias está disponible bajo:
`http://github.com/bluezio/seminario-svn`.

(Sí, en un repositorio Git. ¿Por qué me miráis así?)

Contenidos

- 1 Introducción
- 2 Uso básico
- 3 Uso avanzado

Contenidos

- 1 **Introducción**
 - Antecedentes
 - Conceptos básicos
- 2 Uso básico
- 3 Uso avanzado

Contenidos

- 1 Introducción
 - Antecedentes
 - Conceptos básicos
- 2 Uso básico
- 3 Uso avanzado

¿Por qué usar un SCV?

Copiar ficheros y mandar correos no escala

- ¿Cuál era la última versión?
- ¿Cómo vuelvo a la anterior?
- ¿Cómo reúno mis cambios con los de otro?

Además, señalar a un responsable crea un cuello de botella.

SCV: todo ventajas a cambio de alguna disciplina

- Llevamos un historial de los cambios
- Podemos ir trabajando en varias cosas a la vez
- Podemos colaborar con otros
- Hacemos copia de seguridad de todo el historial

Historia de los SCV

Sin red, un desarrollador

1972 Source Code Control System

1980 Revision Control System

Centralizados

1986 Concurrent Version System

1999 Subversion («CVS done right»)

Distribuidos

2001 Arch, monotone

2002 Darcs

2005 Git, Mercurial (hg), Bazaar (bzzr)

Historia de Apache Subversion (SVN)

Motivación y características principales

- 2000: CVS era el sistema más usado, pero tenía problemas
- “CVS done right”: CollabNet crea Subversion como reemplazo, corrigiendo lo que estaba mal
- Licencia de código abierto: Apache Software License 2.0

Eventos importantes

2001 Puede alojarse a sí mismo

2004 SVN 1.0.0

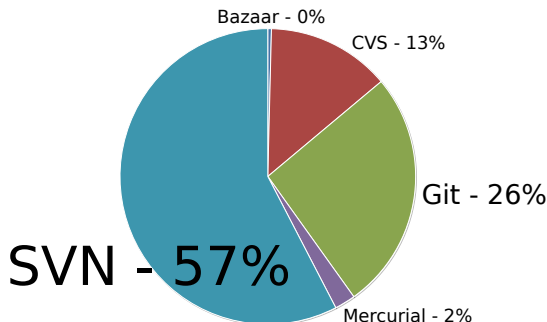
2009 Entra en *Apache Incubator*

2010 Se convierte en un proyecto de primer nivel de Apache

2011 20 octubre: SVN 1.7.1

¿Por qué aprender SVN, si no es lo último?

- SVN permite copias de trabajo parciales y escala mejor que Git ante ficheros binarios grandes: música, imágenes, etc.
- SVN es más fácil de aprender, tiene herramientas más maduras y es más popular.



Porcentajes de proyectos en Ohloh según tipo de repositorio.
<http://www.ohloh.net/repositories/compare>, 2011-11-13.

¿Cuándo no usar SVN?

Copias de seguridad

- No necesitamos el historial: con las últimas n copias nos vale
- Subversion no guarda permisos ni dueños de los ficheros
- Mejor `rsync` o `unison`

Compartir ficheros entre varios usuarios

- Require montar y mantener un servidor Subversion
- Mejor Dropbox o SpiderOak

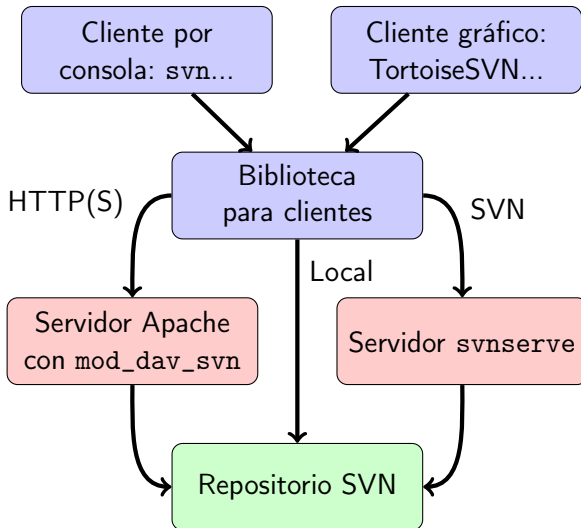
Desarrollo distribuido

- Necesitamos poder trabajar desde cualquier parte
- Necesitamos colaborar sin una entidad central
- Mejor Git... o Mercurial, o Bazaar

Contenidos

- 1 Introducción
 - Antecedentes
 - Conceptos básicos
- 2 Uso básico
- 3 Uso avanzado

Arquitectura de Subversion

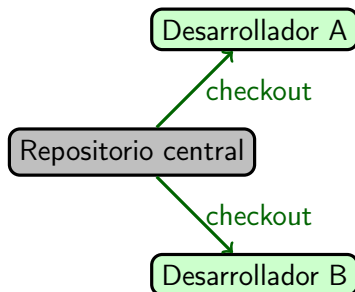


¿Cómo funciona un repositorio?

Repositorio central

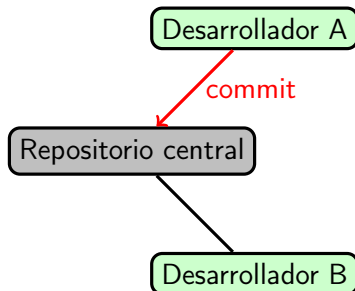
Tenemos nuestro repositorio central con todo dentro.

¿Cómo funciona un repositorio?



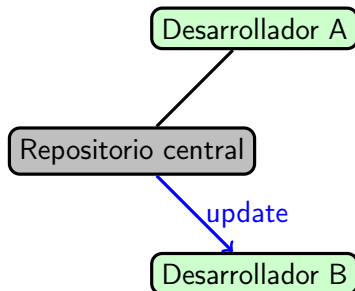
Los desarrolladores crean **copias de trabajo** de la última *revisión* en el servidor.

¿Cómo funciona un repositorio?



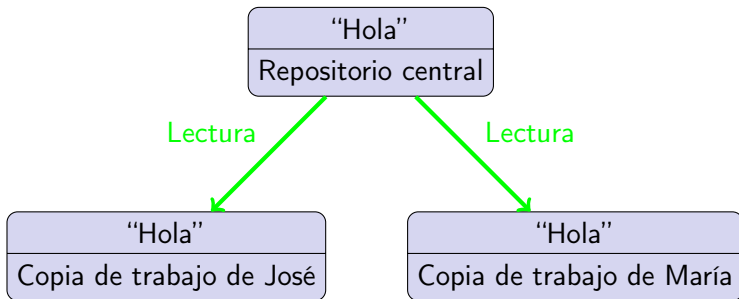
El desarrollador A manda sus cambios al servidor. El servidor los registra como una nueva revisión.

¿Cómo funciona un repositorio?



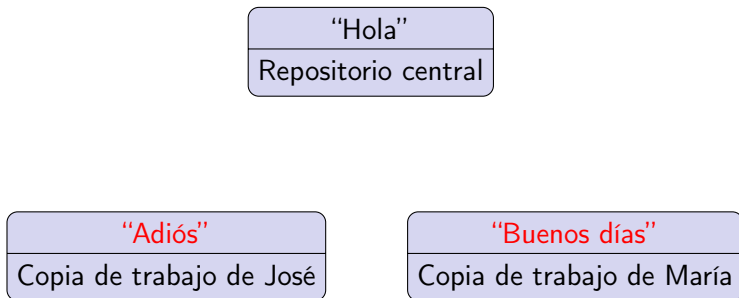
El desarrollador B solicita actualizar su copia de trabajo. El servidor le envía los cambios hechos en la última revisión.

Edición concurrente sobre un repositorio: caso a evitar



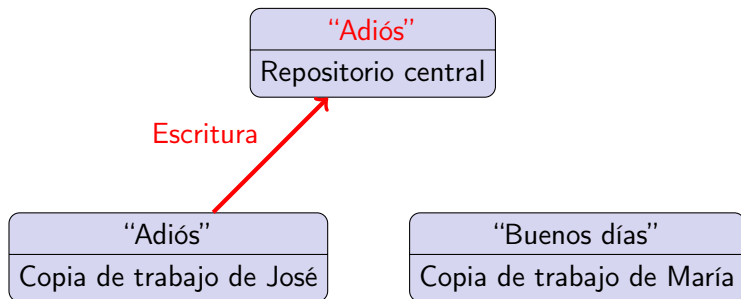
José y María obtienen la última versión del repositorio.

Edición concurrente sobre un repositorio: caso a evitar



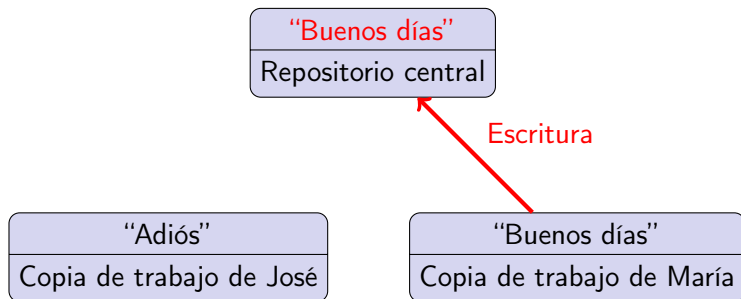
José y María editan sus copias.

Edición concurrente sobre un repositorio: caso a evitar



José se adelanta.

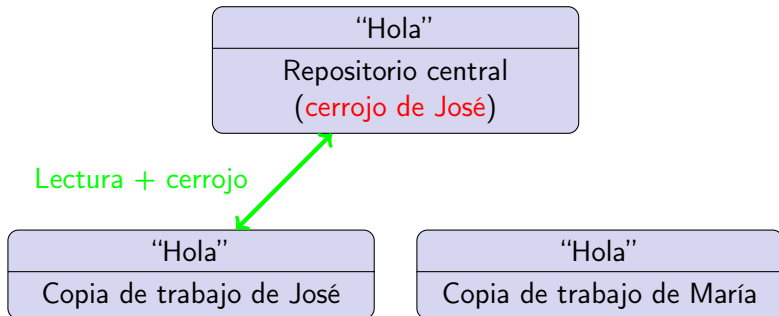
Edición concurrente sobre un repositorio: caso a evitar



María sobrescribe el trabajo de José. ¿Cómo podemos evitarlo?

Solución 1: Bloquear-Modificar-Desbloquear

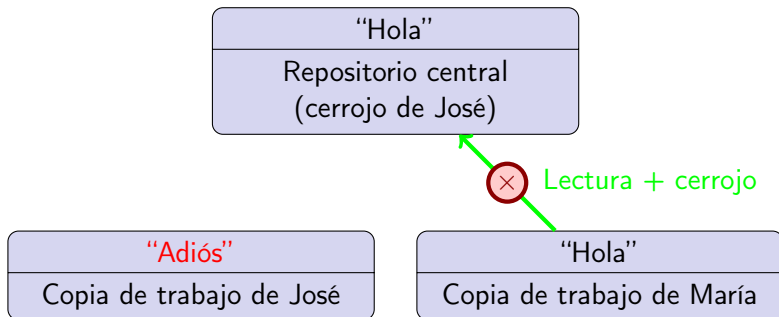
Preferible para archivos binarios, que cambian enteros



José echa el cerrojo sobre el fichero a tocar.

Solución 1: Bloquear-Modificar-Desbloquear

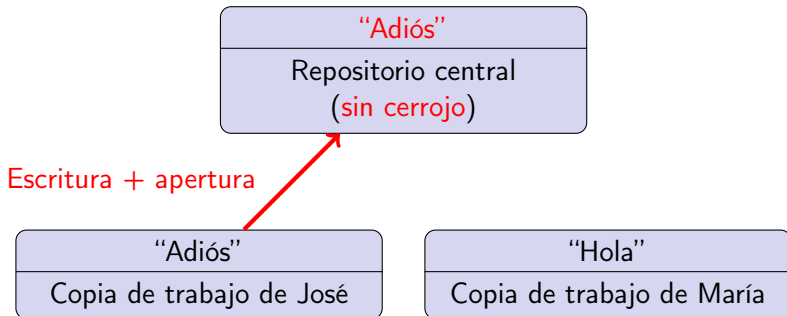
Preferible para archivos binarios, que cambian enteros



María no consigue el cerrojo, José edita.

Solución 1: Bloquear-Modificar-Desbloquear

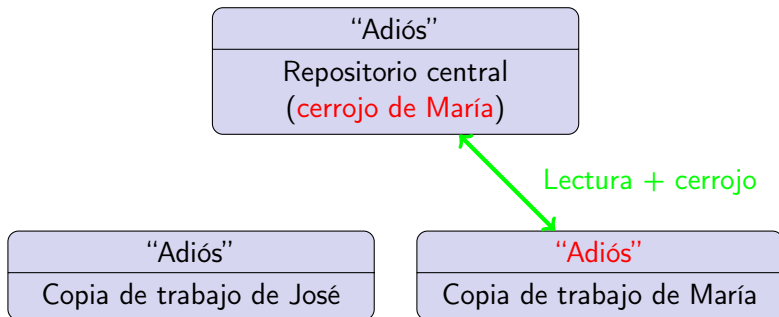
Preferible para archivos binarios, que cambian enteros



José envía sus cambios sobre el fichero y quita el cerrojo.

Solución 1: Bloquear-Modificar-Desbloquear

Preferible para archivos binarios, que cambian enteros



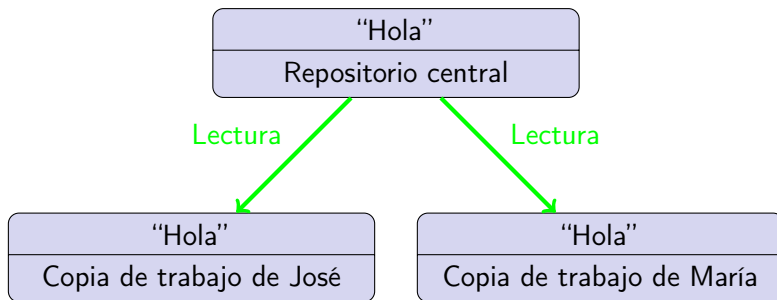
María puede leer y echar el cerrojo.

¿Y si José se va de vacaciones sin quitar el cerrojo?

¿Y si José y María estaban tocando partes distintas del fichero?

Solución 2: Copia-Modificar-Reunir

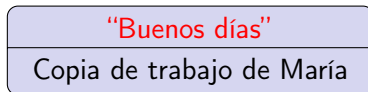
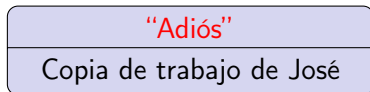
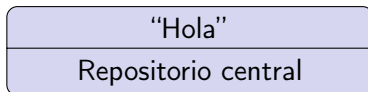
Preferible para archivos de texto, que cambian a trozos



José y María obtienen la última versión del repositorio.

Solución 2: Copia-Modificar-Reunir

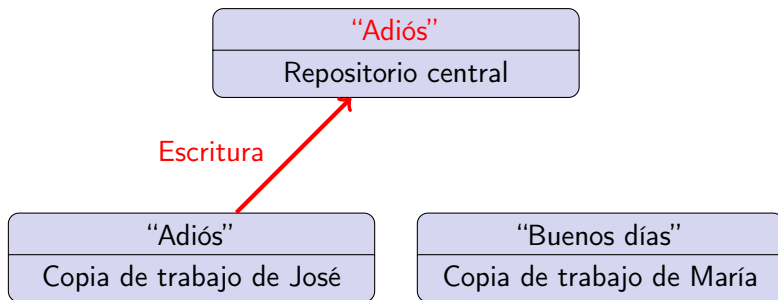
Preferible para archivos de texto, que cambian a trozos



José y María editan sus copias.

Solución 2: Copia-Modificar-Reunir

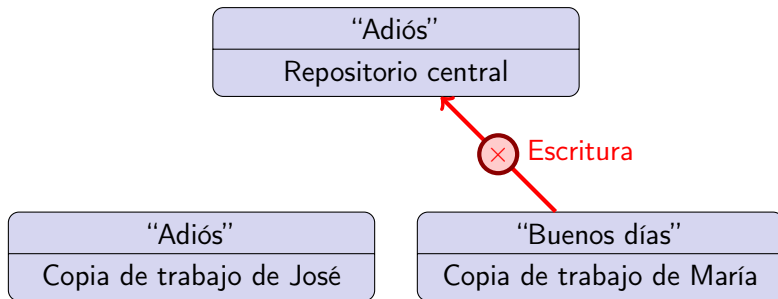
Preferible para archivos de texto, que cambian a trozos



José se adelanta.

Solución 2: Copia-Modificar-Reunir

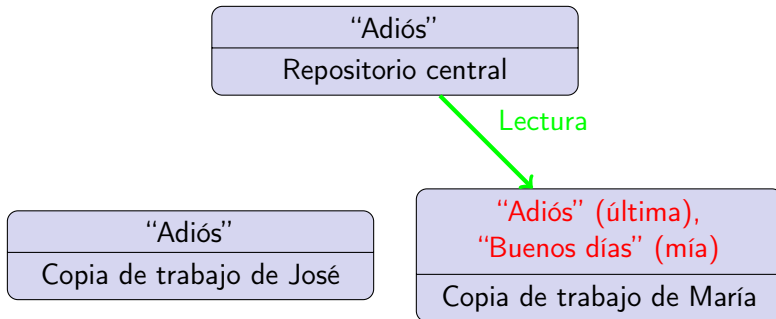
Preferible para archivos de texto, que cambian a trozos



María no puede enviar su versión, que está anticuada.

Solución 2: Copia-Modificar-Reunir

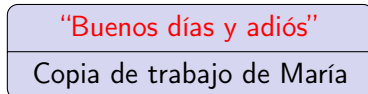
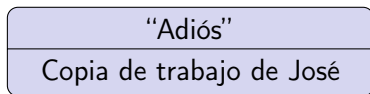
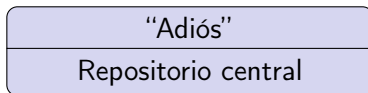
Preferible para archivos de texto, que cambian a trozos



María lee la nueva versión y la compara con la suya.

Solución 2: Copia-Modificar-Reunir

Preferible para archivos de texto, que cambian a trozos



María reúne los cambios.

Solución 2: Copia-Modificar-Reunir

Preferible para archivos de texto, que cambian a trozos



María envía la versión reunida.

Solución 2: Copia-Modificar-Reunir

Preferible para archivos de texto, que cambian a trozos



José obtiene la versión reunida.

No se han necesitado cerrojos, y si se tocan ficheros distintos o partes distintas de un fichero, la reunión es automática.

Contenidos

1 Introducción

2 Uso básico

- Pasos iniciales
- Ciclo normal de trabajo
- Examinar el historial

3 Uso avanzado

Contenidos

1 Introducción

2 **Uso básico**

- **Pasos iniciales**
- Ciclo normal de trabajo
- Examinar el historial

3 Uso avanzado

Cómo consultar la ayuda

- Listado de órdenes disponibles: `svn help`
- Ayuda de una orden:
 - `svn help orden`
 - `svn orden -h`
 - `svn orden -help`

Paso 0: tener un repositorio SVN

¿Qué es un repositorio SVN, realmente?

- Es un directorio con una estructura y contenidos concretos
- Registra todas las *revisiones* de sus contenidos, con fechas y horas, autores y mensajes de resumen
- Normalmente estará en el servidor de nuestra forja
- Se crea mediante `svnadmin create`

```
$ mkdir svnEjemplo
$ svnadmin create svnEjemplo
$ ls svnEjemplo
conf  db  format  hooks  locks  README.txt
```

Paso 1: crear una copia de trabajo del repositorio

Copias de trabajo

- No podemos trabajar directamente sobre un repositorio
- Tenemos que crear una copia de trabajo, en la que veremos los ficheros de la última revisión y podremos modificarlos
- Necesitamos la dirección del repositorio para `svn checkout`:

`http(s)://...` por Apache, sin/con cifrado SSL

`file://...` para rutas locales

`svn://...` para `svnserve`

`svn+ssh://...` para `svnserve` a través de túnel SSH

```
$ svn checkout svnEjemplo ejemplo
```

svn: «svnEjemplo» no parece ser un URL

```
$ svn checkout file:///.../seminario-svn/svnEjemplo/ ejemplo
```

Revisión obtenida: 0

Contenidos

- 1 Introducción
- 2 **Uso básico**
 - Pasos iniciales
 - **Ciclo normal de trabajo**
 - Examinar el historial
- 3 Uso avanzado

Un día típico con Subversion

Proceso general

- 1 Actualizamos nuestra copia de trabajo
- 2 Realizamos nuestros cambios
- 3 Examinamos los cambios
- 4 Deshacemos los cambios que no interesen
- 5 Resolvemos conflictos
- 6 Enviamos nuestros cambios

Actualizar copia de trabajo

Orden: `svn update` o `svn up`

- Actualiza nuestra copia de trabajo con lo último que haya en el repositorio.
- Si tenemos cambios a medio enviar, los intenta reunir automáticamente con lo que había en el repositorio.

```
$ cd ejemplo  
$ svn update  
En la revisión 0.
```


Realizar cambios: añadir ficheros y directorios

Orden: `svn add rutas...`

- Hace que Subversion empiece a controlar los cambios de los ficheros señalados
- Si es un directorio, añade todo lo que está dentro, a menos que se use `svn add -N directorio`
- Código corto de estado: "A"
- Vamos a enviar los cambios con `svn commit`, para enseñar la siguiente orden

```
$ echo "Probando" > f.txt
$ svn add f.txt
A          f.txt
$ svn commit -m "Agregado un fichero muy importante"
Añadiendo      f.txt
Transmitiendo contenido de archivos .
Commit de la revisión 1.
```

Realizar cambios: eliminar ficheros y directorios

Orden: `svn delete rutas...` o `svn rm rutas...`

- Elimina una serie de ficheros o directorios
- SVN ya no monitorizará sus cambios
- Podemos recuperarlos en cualquier momento a partir de versiones anteriores
- Código corto de estado: "D"

```
$ svn rm f.txt
D          f.txt
$ svn commit -m "Al final no era tan importante, no"
Eliminando    f.txt
```

Commit de la revisión 2.

Realizar cambios: copiar ficheros y directorios

Orden: `svn copy origen destino` o `svn cp origen destino`

Es necesario copiar de esta forma para que el destino comparta el historial del origen hasta ahora.

```
$ echo "Otra cosa" > g.txt
$ svn add g.txt
A      g.txt
$ svn ci -m "Creamos el origen"
Añadiendo      g.txt
Transmitiendo contenido de archivos .
Commit de la revisión 3.
$ svn cp g.txt g-copia.txt
A      g-copia.txt
$ svn ci -m "Hemos copiado un fichero"
Añadiendo      g-copia.txt

Commit de la revisión 4.
```

Realizar cambios: mover ficheros y directorios

Orden: `svn move origen destino` o `svn mv origen destino`

Esta orden conserva el historial, aunque cambie la ruta del fichero o el directorio.

```
$ svn mv g-copia.txt h.txt
A      h.txt
D      g-copia.txt
$ svn ci -m "Renombrado g-copia.txt a h.txt"
Eliminando      g-copia.txt
Añadiendo       h.txt
```

Commit de la revisión 5.

Examinar los cambios: cambios dentro de ficheros

Orden: `svn diff [ruta]`

- Compara nuestros ficheros locales con la última revisión obtenida mediante `svn up` (*HEAD*), y nos indica los cambios.
- Se pueden indicar que compare con una revisión concreta usando `-r rev`, o entre dos revisiones con `-r a:b`.

```
$ echo "otra linea" > g.txt
$ svn diff
Index: g.txt
=====
--- g.txt (revisión: 3)
+++ g.txt (copia de trabajo)
@@ -1 +1,2 @@
   Otra cosa
+otra linea
$ svn diff h.txt
```

Examinar los cambios: cambios a nivel de ficheros (I)

Orden: `svn status` o `svn st`

Indica el estado de los ficheros de la copia de trabajo mediante un código de una o dos letras. Ejemplos:

A Añadido

D Borrado

M Modificado

? No está bajo control de versiones

! Desaparecido (borrado sin usar `svn rm`)

~ Tipo equivocado (fichero en vez de dir., o viceversa)

+ Se copiará información de historial

Examinar los cambios: cambios a nivel de ficheros (II)

```
$ touch nuevo-fichero
$ rm h.txt
$ svn mkdir nuevo-directorio
A      nuevo-directorio
$ svn cp g.txt i.txt
A      i.txt
$ svn status
?      nuevo-fichero
M      g.txt
!      h.txt
A +    i.txt
A      nuevo-directorio
```

Deshacer cambios

Orden: `svn revert` (recursivo: `-R`)

- Deshace los cambios realizados sobre la última revisión
- Usuarios de Git: ¡no os confundáis con `git revert`!

```
$ svn revert h.txt
Se revirtió «h.txt»
$ svn status
?      nuevo-fichero
M      g.txt
A +    i.txt
A      nuevo-directorio
$ svn revert -R .
Se revirtió «g.txt»
Se revirtió «i.txt»
Se revirtió «nuevo-directorio»
$ svn status
?      i.txt
?      nuevo-fichero
?      nuevo-directorio
```


Resolver conflictos

```
$ cd ..
$ svn checkout file:///.../seminario-svn/svnEjemplo/ ejemplo-conflicto
A    ejemplo-conflicto/g.txt
A    ejemplo-conflicto/h.txt
Revisión obtenida: 5
$ echo "hacemos un cambio" > ejemplo-conflicto/g.txt
$ echo "hacemos otro cambio" > ejemplo/g.txt
$ cd ejemplo
$ svn ci -m "Hago un cambio a escondidas de ejemplo-conflicto"
Enviando      g.txt
Transmitiendo contenido de archivos .
Commit de la revisión 6.
$ cd ejemplo-conflicto
$ svn up
Se descubrió un conflicto en «g.txt».
Seleccione: (p) posponer, (df) ver dif. completo, (e) editar,
            (mc) mío conflicto, (tc) de ellos conflicto,
            (s) mostrar todas las opciones: C    g.txt
Actualizado a la revisión 6.
Resumen de conflictos:
  Conflictos de texto: 1
```

Enviar cambios

Orden: `svn commit` o `svn ci`

- Se puede proporcionar un mensaje mediante `-m`
- De lo contrario, se abrirá el editor por omisión
- Se necesita una conexión de red para enviar los cambios
- Los cambios son atómicos: si se corta la conexión a la mitad, es como si no hubiera pasado nada

Contenidos

1 Introducción

2 Uso básico

- Pasos iniciales
- Ciclo normal de trabajo
- Examinar el historial

3 Uso avanzado

Examinar el historial: revisiones de un fichero

Orden: `svn log ruta`

Da un listado con las revisiones de una ruta.

```
$ cd ../ejemplo
```

```
$ svn log g.txt
```

```
-----  
r6 | antonio | 2011-11-14 23:19:31 +0100 (lun 14 de nov de 2011) | 1 línea
```

```
Hago un cambio a escondidas de ejemplo-conflicto
```

```
-----  
r3 | antonio | 2011-11-14 23:19:27 +0100 (lun 14 de nov de 2011) | 1 línea
```

```
Creamos el origen  
-----
```

Examinar el historial: revisiones del repositorio

Orden: `svn log -r 4:HEAD`

Da un listado con las revisiones desde la revisión 4 hasta la más reciente desde el último `svn up`.

```
$ cd ../ejemplo
```

```
$ svn log -r 4:HEAD
```

```
-----  
r4 | antonio | 2011-11-14 23:19:28 +0100 (lun 14 de nov de 2011) | 1 línea
```

```
Hemos copiado un fichero
```

```
-----  
r5 | antonio | 2011-11-14 23:19:29 +0100 (lun 14 de nov de 2011) | 1 línea
```

```
Renombrado g-copia.txt a h.txt
```

```
-----  
r6 | antonio | 2011-11-14 23:19:31 +0100 (lun 14 de nov de 2011) | 1 línea
```

```
Hago un cambio a escondidas de ejemplo-conflicto  
-----
```

Examinar el historial: contenido de un fichero

Orden: `svn cat -r rev fichero`

Muestra los contenidos del fichero indicado en una revisión determinada.

```
$ svn cat -r 3 g.txt
Otra cosa
$ svn cat -r 6 g.txt
Otra cosa
hacemos otro cambio
```

Examinar el historial: listado de ficheros

Orden: `svn list -r rev directorio`

Lista los contenidos del directorio indicado en una revisión determinada.

```
$ svn list -r 4 .  
g-copia.txt  
g.txt  
$ svn list -r 2 .  
$ svn list -r 1 .  
f.txt
```

Examinar el historial: autoría de líneas

Orden: `svn blame fichero`

- También `svn praise fichero` o `svn annotate fichero`, según nuestro estado de ánimo :-)
- Anota cada línea de un fichero con la información de la última revisión en que se modificó

```
$ svn blame g.txt
3    antonio Otra cosa
6    antonio hacemos otro cambio
```


Contenidos

- 1 Introducción
- 2 Uso básico
- 3 **Uso avanzado**
 - Ramas y etiquetas
 - Exportación
 - Metadatos

Contenidos

- 1 Introducción
- 2 Uso básico
- 3 **Uso avanzado**
 - **Ramas y etiquetas**
 - Exportación
 - Metadatos

Conceptos de ramas y etiquetas

Ramas: líneas de desarrollo alternativas

- Para realizar cambios significativos sin molestar a los demás, para después traerlos a la rama principal (*trunk*)
- Para realizar correcciones sobre versiones antes publicadas, sin mezclarlas con las novedades de la siguiente versión
- Para mantener una versión estable y otra experimental

Etiquetas

- Nos permiten darle un nombre a una revisión: en vez de “revisión 238”, podemos pedir la “versión 1.0”
- Son una instantánea del estado del repositorio

Ramas y etiquetas en Subversion: estructura básica

Implementación

- Directorios bajo una estructura estándar
- Aprovechan el hecho de que en Subversion, copiar con `svn cp` tiene coste $O(1)$ en tiempo y espacio en el repositorio

```
$ mkdir svnRamas; svnadmin create svnRamas
$ svn checkout file:///.../seminario-svn/svnRamas/ ramas
Revisión obtenida: 0
$ cd ramas
$ svn mkdir trunk branches tags
A      trunk
A      branches
A      tags
$ svn ci -m "Creado estructura de rama principal/ramas/etiquetas"
Añadiendo      branches
Añadiendo      tags
Añadiendo      trunk
```

Commit de la revisión 1.

Ramas y etiquetas en Subversion: creación de etiquetas

Sólo tenemos que hacer una simple copia (mejor con URL):

```
$ svn cp -m "Copiado trunk a tags/1.0" \  
    file:///home/antonio/Documentos/seminario-svn/svnRamas/trunk \  
    file:///home/antonio/Documentos/seminario-svn/svnRamas/tags/1.0
```

Commit de la revisión 2.

Ramas y etiquetas en Subversion: trabajar en rama principal

Cambiamos a trunk mediante svn switch:

```
$ svn switch file:///.../svnRamas/trunk
D    trunk
D    branches
D    tags
Actualizado a la revisión 2.
$ echo "Hola" > f.txt
$ svn add f.txt
A    f.txt
$ svn ci -m "Primer cambio en rama principal"
Añadiendo      f.txt
Transmitiendo contenido de archivos .
Commit de la revisión 3.
```

Ramas en Subversion: uso de una rama aparte (I)

Abrir una rama usa `svn cp` con URL origen y destino:

```
$ svn cp -m "Copiado trunk a branches/inestable" \  
  file:///home/antonio/Documentos/seminario-svn/svnRamas/trunk \  
  file:///home/antonio/Documentos/seminario-svn/svnRamas/branches/inestable
```

Commit de la revisión 4.

Trabajar en una rama sólo requiere cambiar a ella:

```
$ svn switch file:///.../svnRamas/branches/inestable
```

En la revisión 4.

```
$ echo "Nuevo" > nuevo.txt; svn add nuevo.txt
```

```
A      nuevo.txt
```

```
$ svn ci -m "Primer cambio inestable"
```

```
Añadiendo      nuevo.txt
```

```
Transmitiendo contenido de archivos .
```

Commit de la revisión 5.

Ramas en Subversion: uso de una rama aparte (II)

Traer cambios a la rama actual: `svn merge origen`

- Antes había que poner el rango de revisiones de la rama origen a aplicar mediante `-r inicio:fin`
- Desde SVN 1.5, el rango se puede calcular automáticamente si se usa una URL en origen

```
$ svn switch file:///.../svnRamas/trunk
D    nuevo.txt
Actualizado a la revisión 5.
$ svn merge file:///.../svnRamas/branches/inestable
--- Fusionando r4 hasta r5 en «.»:
A    nuevo.txt
$ svn ci -m "Integrado cambios de branches/inestable en trunk"
Enviando      .
Añadiendo     nuevo.txt

Commit de la revisión 6.
```


Contenidos

- 1 Introducción
- 2 Uso básico
- 3 **Uso avanzado**
 - Ramas y etiquetas
 - **Exportación**
 - Metadatos

Exportar la copia de trabajo a un directorio aparte

Orden: `svn export origen destino`

- Copia los ficheros bajo control de versiones a destino
- No copia los `.svn` ni ficheros fuera de control de versiones
- Útil para generar distribuciones limpias de cosas de SVN
- Puede usarse una URL (por ejemplo, la de una etiqueta)

```
$ ls -a .  
.  ..  f.txt  nuevo.txt  .svn  
$ svn export . /tmp/ramas-HEAD  
Exportación completa.  
$ ls -a -C -color /tmp/ramas-HEAD  
.  ..  f.txt  nuevo.txt
```

Contenidos

- 1 Introducción
- 2 Uso básico
- 3 **Uso avanzado**
 - Ramas y etiquetas
 - Exportación
 - **Metadatos**

Propiedades de ficheros y directorios

Concepto general

- SVN puede asociar *propiedades* a ficheros y directorios
- También están bajo control de versiones
- Se pueden cambiar con las órdenes:

`svn pg nombre ruta` Obtiene el valor de una propiedad

`svn ps nombre valor ruta` Cambia el valor de una propiedad

`svn pe nombre ruta` Edita el valor de una propiedad

`svn pl ruta` Lista las propiedades de una ruta

Ejemplos útiles

`svn:ignore` Ignora en status y add una serie de ficheros

`svn:mime-type` Tipo MIME del fichero (para Apache)

`svn:executable` Indica si es ejecutable en sistemas UNIX

Ejemplo con svn:ignore

```
$ touch foo.tmp
$ svn st
?      foo.tmp
$ svn ps svn:ignore foo.tmp .
propiedad «svn:ignore» asignada en «.»
$ svn st
M      .
$ svn ci -m "No me interesa foo.tmp"
Enviando      .
```

Commit de la revisión 7.

Recursos útiles



B. Collins-Sussman, B. W. Fitzpatrick y C. M. Pilato.
Control de versiones con Subversion.

<http://svnbook.red-bean.com>



B. Collins-Sussman.

SVN 1.5 merge tracking in a nutshell.

<http://blog.red-bean.com/sussman/?p=92>



D. Child.

Subversion Cheat Shett.

[http://www.addedbytes.com/cheat-sheets/
subversion-cheat-sheet/](http://www.addedbytes.com/cheat-sheets/subversion-cheat-sheet/)



Tigris.org.

TortoiseSVN: interfaz gráfica para SVN en Windows.

<http://tortoisesvn.tigris.org>

¡Gracias!

antonio.garciadominguez@uca.es