Control de Versiones

CVS

(Concurrent Versions System)

Los Sistemas de Control de Versiones (SCV)

- **Un SCV controla los cambios realizados en el código fuente durante el desarrollo de un proyecto:**
  - Quién cambia el código.
  - Cuándo cambia el código.
  - Qué cambia en el código.
- **Versión = cambio con un significado dentro del proyecto**
- **Los programadores poseen un historial completo de cambios:**
  - Es posible volver a un estado anterior de cualquier archivo en caso de que ocurran problemas.
- **Permiten la colaboración de diversos programadores (remotos) en el desarrollo:**
  - Acceso a archivos generados por otros programadores.

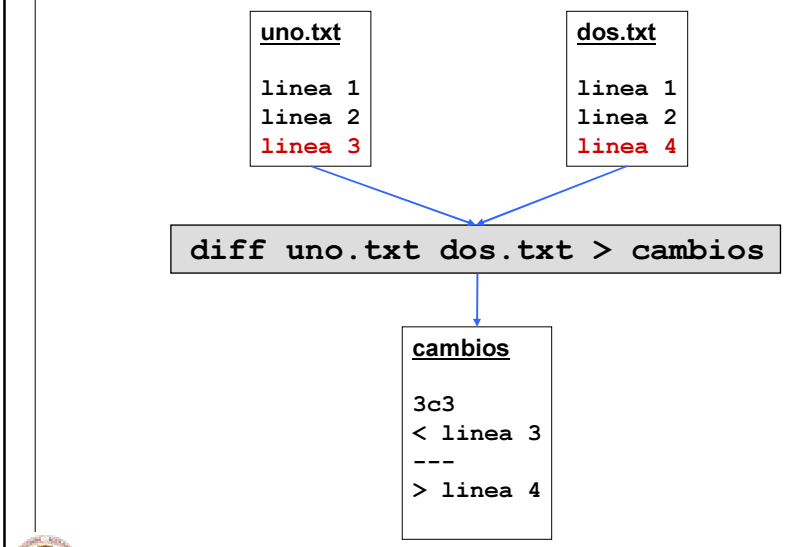
CVS (Concurrent Versions Systems)

- **Es (posiblemente) el sistema de control de versiones más empleado.**
  - ✓ Otros: RCS, SCCS
- **Es nace como un conjunto de scripts para facilitar el uso de RCS (1986). Posteriormente se reescribe en C y se le añade la posibilidad de trabajo distribuido (1990).**
  - ✓ Ciertos formatos se heredan de RCS

CVS (Características)

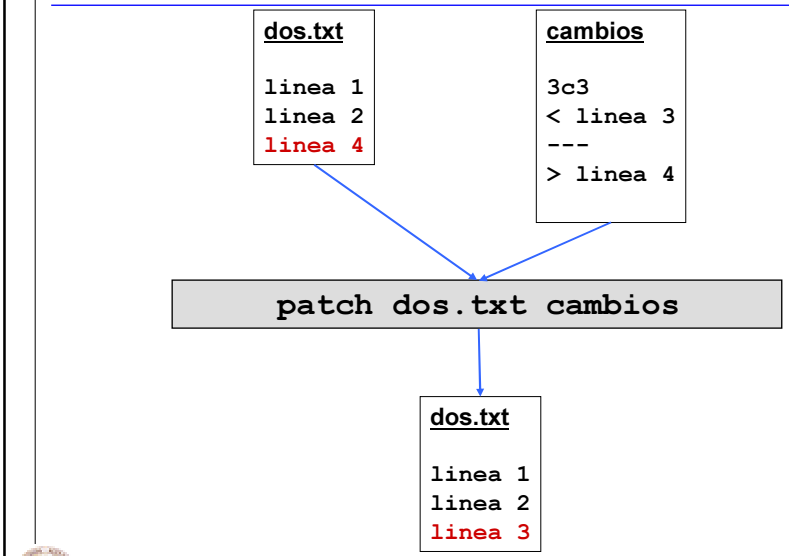
- **Es un sistema descentralizado:**
  - ✓ Cada programador utiliza su propio directorio de trabajo.
  - ✓ Permite la edición concurrente de archivos.
  - ✓ Integra todos los cambios realizados en un archivo (mezcla).
- **Todas las versiones de un archivo se almacenan de manera conjunta en un único archivo de manera incremental:**
  - ✓ Se basa en dos programas: `diff` y `patch`
  - ✓ `diff`: Detecta las diferencias entre dos archivos (texto)
  - ✓ `patch`: Reconstruye un archivo a partir del original y las diferencias.

diff


uno.txt
linea 1
linea 2
linea 3
dos.txt
linea 1
linea 2
linea 4
diff uno.txt dos.txt > cambios
cambios
3c3
< linea 3

> linea 4

patch


dos.txt
linea 1
linea 2
linea 4
cambios
3c3
< linea 3

> linea 4
patch dos.txt cambios
dos.txt
linea 1
linea 2
linea 3

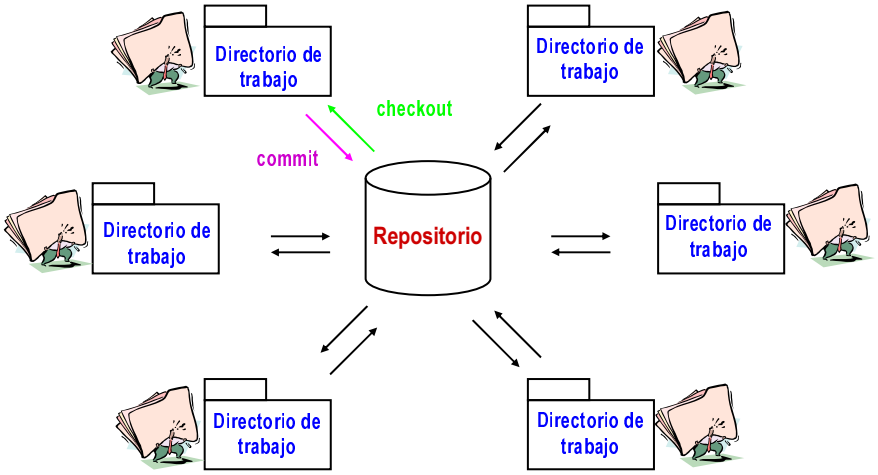
CVS no es

- **CVS no se encarga de la organización de archivos del proyecto.**
- **No es una herramienta de construcción de software (make):**
  - ✓ No controla las dependencias entre archivos.
- **CVS no sustituye a la comunicación entre los programadores:**
  - ✓ CVS trabaja con texto, no con la lógica de la aplicación.
- **CVS no incluye un modelo de gestión y desarrollo del proyecto.**

El repositorio CVS

- **Es el lugar (directorio) donde se almacenan todos los archivos con las diferentes versiones de un proyecto (o de varios).**
- **Es el depósito común de información del proyecto. Se recurre a él para recuperar archivos y almacenar los cambios.**
- **Almacena información de control para CVS.**
- **Almacena las diferencias entre versiones de cada uno de los archivo que constituyen el proyecto.**

Repositorio (2)



Modelo de trabajo con CVS

- **Un programador descarga una copia de trabajo desde el repositorio (`checkout`).**
- **El programador edita libremente su copia.**
  - ✓ Al mismo tiempo, cualquier otro miembro del equipo de desarrollo puede trabajar sobre otra copia de trabajo.
- **El programador finaliza sus cambios y los "entrega" al repositorio de CVS (`commit`), junto con un texto explicativo de las modificaciones realizadas.**
  - ✓ CVS integra los cambios en la copia "master" del proyecto.
- **Otros desarrolladores pueden solicitar a CVS comprobar si la copia "master" ha sufrido cambios recientes (`update`).**

Decisiones políticas sobre el proyecto

- **Cuando actualizar y/o entregar cambios (`update`, `commit`) depende de las preferencias personales y del proyecto.**
  - ✓ P.ej., suele ser habitual entregar una modificación cuando los cambios previstos han finalizado y probado → La copia "master" siempre permita generar una versión ejecutable.
- **No hay limitación temporal para la copia de trabajo. Se puede mantener "abierta" horas, días, ¿semanas?, ¿...?**

Gestión del Repositorio

- **Crear**
- **Importar**
- **Conectar remotamente**

Crear un repositorio

- **Crear un repositorio para uno o más proyectos. Se realiza sólo una vez:**

```
cvs -d /home/xaro/micvs/DEPOSITO init
```

- **Sintaxis general de órdenes cvs:**

```
cvs [-d <repositorio>] [opciones globales] orden
  [opciones] [argumentos]
```

CVSROOT

- **En las órdenes cvs es preciso indicar el repositorio con el que se trabaja. Para evitar escribirlo constantemente se puede asignar la ruta a la variable de entorno `CVSROOT`:**

  ✓ Establece la ubicación del repositorio:

```
export CVSROOT=/home/xaro/micvs/DEPOSITO
```

  ✓ Es conveniente, por simplicidad, usar esta variable:

```
cvs init
```

Crear una entrada (proyecto) nueva

- **Importar el proyecto:**
  - ✓ Ya tenemos el (primer) código del proyecto que queremos controlar con CVS.
  - ✓ El proyecto está organizado en múltiples subdirectorios.
  - ✓ Nos ubicamos en el directorio raíz de nuestro proyecto. P.ej., `~/mi_proyecto`.

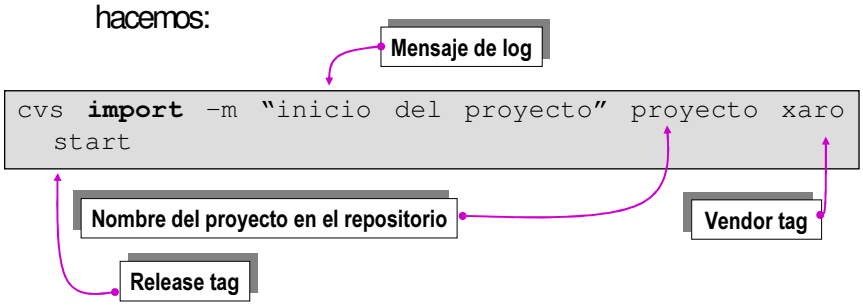
```
xaro@samba:~/mi_proyecto> ls
LEEME.txt  carpeta1  hola.c

xaro@samba:~/mi_proyecto> ls carpeta1
adios.c
```

Contenido inicial de los archivos

```
xaro@samba:~/mi_proyecto> cat LEEME.txt
Esto es sólo un proyecto de prueba.
No tiene ninguna finalidad especial.

xaro@samba:~/mi_proyecto> cat hola.c
#include <stdio.h>
void main()
{
        printf("Hola mundo\n");
}

xaro@samba:~/mi_proyecto> cd carpeta1/

xaro@samba:~/mi_proyecto/carpeta1> cat adios.c
#include <stdio.h>
void main()
{
        printf("Adios mundo\n");
}
```

Importar

- **Importar el proyecto:**
  - ✓ Nos ubicamos en el directorio raíz de nuestro proyecto y hacemos:



  - ✓ Ya hay una entrada para este proyecto en el repositorio.

Importar (2)

```
xaro@samba:~/mi_proyecto> cvs import -m "inicio del
  proyecto" proyecto xaro start
N proyecto/LEEME.txt
N proyecto/hola.c
CVS import: Importing
  /home/xaro/micvs/DEPOSITO/proyecto/carpeta1
N proyecto/carpeta1/adios.c
No conflicts created by this import
```

Conexión remota con el repositorio

- No es preciso trabajar con un repositorio local, CVS permite trabajar con repositorios remotos.
- Existen 4 modos de conexión:
  - ✓ `Pserver` (windows)
  - ✓ `ext` ← El más habitual, implica el uso de un programa de conexión ajeno a CVS. P.ej., `rsh` o `ssh`
  - ✓ `Kserver` (Kerberos System v4)
  - ✓ `Gserver` (Generic Security Services API)

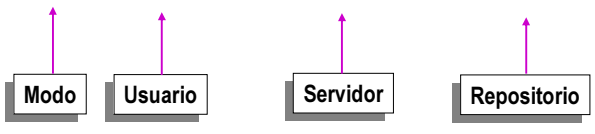
Conexión por `ssh`

- Fijar variable de entorno `CVS_RSH` que indica el programa a emplear en el modo `ext`:

```
export CVS_RSH=ssh
```

- Indicar la localización del repositorio con la sintaxis:

```
export CVSROOT=:ext:xaro@slabii.uv.es:/cvs/COPUT
```



Trabajar con CVS

Crear copia de trabajo

- **Ubicados en el directorio donde queramos trabajar, hacer:**

```
cvs checkout proyecto
```

  - ✓ Asumimos fijada la localización del repositorio

```
CVSROOT = /home/xaro/micvs/DEPOSITO
```

- **Crea una copia de toda la estructura de directorios del proyecto en nuestra ubicación actual.**
- **Cada directorio contiene información para el control de las versiones (directorios CVS).**

Ejemplo de checkout

```
xaro@samba:~/mitrabajo> cvs checkout proyecto
cvs checkout: Updating proyecto
U proyecto/LEEME.txt
U proyecto/hola.c
cvs checkout: Updating proyecto/carpeta1
U proyecto/carpeta1/adios.c

xaro@samba:~/mitrabajo> ls
proyecto

xaro@samba:~/mitrabajo> ls -l proyecto
total 8
drwxr-xr-x    2 xaro users    128 2004-03-23 09:10 CVS
-rw-r--r--    1 xaro users     74 2004-03-23 09:01 LEEME.txt
drwxr-xr-x    3 xaro users     96 2004-03-23 09:10 carpeta1
-rw-r--r--    1 xaro users     62 2004-03-23 09:01 hola.c
```

Contenido de los directorios CVS

```
xaro@samba:~/mitrabajo/proyecto> cd CVS

xaro@samba:~/mitrabajo/proyecto/CVS> ls -l
total 12
-rw-r--r--    1 xaro users         104 2004-03-23 09:10 Entries
-rw-r--r--    1 xaro users           9 2004-03-23 09:10 Repository
-rw-r--r--    1 xaro users          28 2004-03-23 09:10 Root

xaro@samba:~/mitrabajo/proyecto/CVS> cat Entries
/LEEME.txt/1.1.1.1/Tue Mar 23 09:10:58 2004//
/hola.c/1.1.1.1/Tue Mar 23 09:10:58 2004//
D/carpeta1////

xaro@samba:~/mitrabajo/proyecto/CVS> cat Repository
proyecto

xaro@samba:~/mitrabajo/proyecto/CVS> cat Root
/home/xaro/micvs/DEPOSITO
```

Información sobre el estado de cada archivo del directorio

Apunta al proyecto dentro del repositorio

Apunta al repositorio raíz

Guardar modificaciones en el repositorio

- Ejemplo: Hemos modificado el archivo "`hola.c`"

```
#include <stdio.h>
void main()
{
   printf("Hola mundo\n");
}
```



```
#include <stdio.h>
void main()
{
   printf("Hola mundo\n");
   printf("Hola Xaro\n");
}
```

- No todos los cambios realizados se guardan en el repositorio. Sólo los que nosotros deseemos.

commit

```
xaro@samba:~/mitrabajo/proyecto> cvs commit \
-m "Añadido un saludo más"
cvs commit: Examining .
cvs commit: Examining carpeta1
Checking in hola.c;
/home/xaro/micvs/DEPOSITO/proyecto/hola.c,v  <--  hola.c
new revision: 1.2; previous revision: 1.1
done
```

Numeración de revisiones

Actualización de la copia de trabajo

- Antes de realizar `commit` podría haber deseado comparar la versión de "`hola.c`" con la última versión almacena en el repositorio.
- Si otro usuario ha modificado este mismo archivo (u otros) es posible incorporar estos cambios a nuestra copia de trabajo.
- La actualización no descarga de nuevo el proyecto, sólo integra los cambios.

update

Actualizar todos los archivos

```
xaro@samba:~/mitrabajo/proyecto> cvs update
cvs update: Updating .
M hola.c
cvs update: Updating carpeta1
Indica que se ha modificado el archivo 'hola.c'
```

Indica que la copia de trabajo está modificada respecto a lo que hay en el repositorio (nadie ha cambio el repositorio desde el checkout)

Comparar diferencias (`diff`)

Compara la versión del repositorio con el archivo "`hola.c`" de la copia de trabajo.

```
xaro@samba:~/mitrabajo/proyecto> cvs diff hola.c
Index: hola.c
===================================================================
RCS file: /home/xaro/micvs/DEPOSITO/proyecto/hola.c,v
retrieving revision 1.1.1.1
diff -r1.1.1.1 hola.c
5a6
>       printf("Hola Xaro\n");
```

Comparar diferencias, 2 (`diff -c`)

```
xaro@samba:~/mitrabajo/proyecto> cvs diff -c hola.c
Index: hola.c
===================================================================
RCS file: /home/xaro/micvs/DEPOSITO/proyecto/hola.c,v
retrieving revision 1.1.1.1
diff -c -r1.1.1.1 hola.c
*** hola.c      23 Mar 2004 09:10:58 -0000      1.1.1.1
--- hola.c      23 Mar 2004 09:20:07 -0000
***************
*** 3,7 ****
--- 3,8 ----
  void main()
  {
        printf("Hola mundo\n");
+       printf("Hola Xaro\n");
  }
```

Opción –c (contexto) nos da la información de contexto que genera la orden diff

Conflictos

- ¿Qué ocurre si dos usuarios modifican de manera "contradictoria" un archivo?

Xaro

LEEME.txt

```
Esto es sólo un proyecto de prueba.
No tiene ninguna finalidad especial.
Aunque a veces puede generar conflictos.
```

Jesús

LEEME.txt

```
Esto es sólo un proyecto de prueba.
No tiene ninguna finalidad especial.
Fin.
```

Conflictos (2)

Xaro

Xaro actualiza, comprueba que nadie ha cambiado nada y envía el archivo "LEEME.txt" al repositorio

```
xaro@samba:~/mitrabajo/proyecto> cvs update
CVS update
CVS update: Updating .
M LEEME.txt
M hola.c
CVS update: Updating carpeta1

xaro@samba:~/mitrabajo/proyecto> cvs commit -m "Xaro añade una \
linea" LEEME.txt
Checking in LEEME.txt;
/home/xaro/micvs/DEPOSITO/proyecto/LEEME.txt,v  <--  LEEME.txt
new revision: 1.2; previous revision: 1.1
done
```

Conflictos (3)

Jesús actualiza → Detección de conflictos

```
jesus@samba:~/trabajoJesus/proyecto> cvs update
cvs update: Updating .
RCS file: /home/xaro/micvs/DEPOSITO/proyecto/LEEME.txt,v
retrieving revision 1.1.1.1
retrieving revision 1.2
Merging differences between 1.1.1.1 and 1.2 into LEEME.txt
rcsmerge: warning: conflicts during merge
cvs update: conflicts found in LEEME.txt
C LEEME.txt
cvs update: Updating carpeta1
```

Conflictos (4)


```
jesus@samba:~/trabajoJesus/proyecto> cat LEEME.txt
Esto es sólo un proyecto de prueba.
No tiene ninguna finalidad especial.
<<<<<<< LEEME.txt
Fin.
=======
Aunque a veces puede generar conflictos.

>>>>>>> 1.2

jesus@samba:~/trabajoJesus/proyecto> cat CVS/Entries
/hola.c/1.2/Tue Mar 23 09:09:32 2004//
D/carpeta1////
/LEEME.txt/1.2/Result of merge+Tue Mar 23 09:37:27 2004//
```

Conflictos (5)


```
jesus@samba:~/trabajoJesus/proyecto> cvs commit -m "Conflictos"
LEEME.txt
cvs commit: file 'LEEME.txt' had a conflict and has not been
modified
cvs [commit aborted]: correct above errors first!
```

- **Solución: Editar el archivo quitando las marcas y resolviendo el conflicto (toma de decisiones) → `commit`**

Añadir archivos

- **CVS sólo controla los archivos que reconoce como propios.**
- **Para enviar un nuevo archivo al repositorio es preciso marcarlo previamente (`add`) y posteriormente enviarlo al repositorio (commit).**

```
xaro@samba:~/mitrabajo/proyecto> ls
CVS  LEEME.txt  carpeta1  hola.c  nuevo.c

xaro@samba:~/mitrabajo/proyecto> cvs add nuevo.c
cvs add: scheduling file 'nuevo.c' for addition
cvs add: use 'cvs commit' to add this file permanently

xaro@samba:~/mitrabajo/proyecto> commit -m "Añado f" nuevo.c
RCS file: /home/xaro/micvs/DEPOSITO/proyecto/nuevo.c,v
done
Checking in nuevo.c;
/home/xaro/micvs/DEPOSITO/proyecto/nuevo.c,v <-- nuevo.c
Initial revision: 1.1
done
```

Añadir directorios

- En este caso, la orden `add` añade automáticamente el directorio al repositorio sin necesidad de aplicar `commit`.

```
xaro@samba:~/mitrabajo/proyecto> mkdir carpeta2

xaro@samba:~/mitrabajo/proyecto> cvs add carpeta2
Directory /home/xaro/micvs/DEPOSITO/proyecto/carpeta2 added
to the repository
```

Eliminar archivos

- Es preciso eliminar el archivo, marcarlo (`remove`) y posteriormente aplicar `commit`.

```
xaro@samba:~/mitrabajo/proyecto> rm nuevo.c

xaro@samba:~/mitrabajo/proyecto> cvs remove nuevo.c
cvs remove: scheduling 'nuevo.c' for removal
cvs remove: use 'cvs commit' to remove this file permanently

xaro@samba:~/mitrabajo/proyecto> cvs commit \
-m "Eliminado el nuevo archivo"
cvs commit: Examining .
cvs commit: Examining carpeta1
cvs commit: Examining carpeta2
Removing nuevo.c;
/home/xaro/micvs/DEPOSITO/proyecto/nuevo.c,v  <--  nuevo.c
new revision: delete; previous revision: 1.1
done
```

Eliminar directorios

- Eliminar los archivos del directorio (y del repositorio) como se ha visto previamente.
- Cuando se ejecute `update -P` (***prune***, podar) se eliminarán los directorios vacíos de la copia de trabajo.