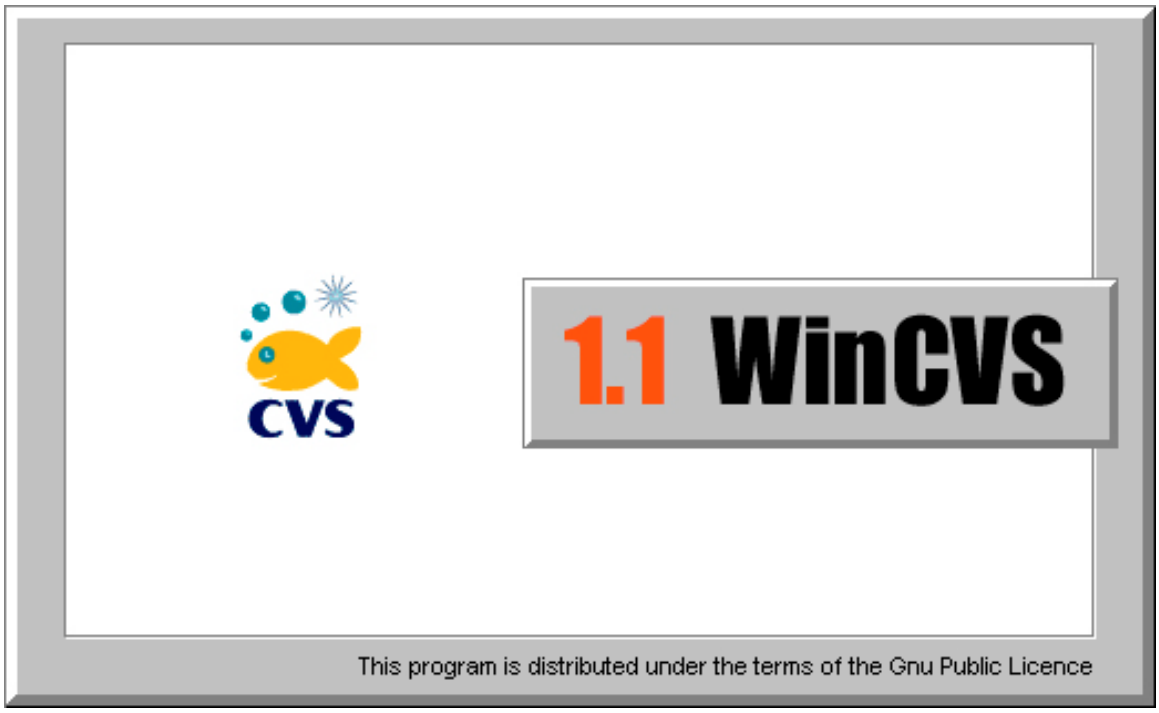
WinCvs Version 1.1

Guía del Usuario



Don Harper

June 1, 1999

Traducido al castellano por Enrique Castilla, Oct-2000.

Seccion 1 – Introducción.

CVS es el acrónimo de Concurrent Versions System. Es un sistema de control de versiones cuyo desarrollo comenzo en 1986, y en el que han participado muchas personas. Es un software de dominio publico. Actualmente quien mantiene CVS es Cyclic Software, en Washington DC, y hay montones de informacion en su sitio web: www.cyclic.com.

El principal "problema" de CVS es que usa una interface a base de comandos en linea. Ya que la mayoria de los desarrolladores prefieren utilizar una bonita interfaz grafica de usuario, varios grupos a lo largo y ancho del mundo han desarrollado interfaces graficas de usuario para el nucleo de CVS. De entre estas interfaces graficas la mejor y disponible para Windows NT/98 es WinCvs, desarrollada y mantenida por un grupo de gente a lo largo del mundo. Se puede encontrar informacion sobre WinCvs en su sitio web en www.wincvs.org.

Notese que las versiones de cvs y WinCvs documentadas aquí difieren un poco de las versiones standard. La diferencia mas visible se puede apreciar en los ejemplos del comando import. Un parche no oficial llamado Main Branch Import ha sido aplicado para permitir la importacion de ficheros directamente al trunk !!! en lugar de una bifurcacion de vendedor. El parche puede encontrarse en http://www.cyclic.com/cvs/dev-trunk-import.txt.

Este documento describe la instalacion de WinCvs y explica algunos de los conocimientos basicos necesarios para empezar a usar CVS. Por favor, dirige todas las preguntas o problemas sobre este manual a Don Harper (usa los grupos de noticias para cuestiones sobre cvs o el mismo WinCvs. [mailto:donharper@earthlink.net].

Nótese también que este documento asume que la máquina es Windows NT, sin embargo, para los usuarios de Windows 98 debería ser tambien útil aunque haya unos pocos detalles diferentes.

Sección 2 – Instrucciones de Instalación

La versión actual de WinCvs es la 1.1, que esta todavia en fase de pruebas beta y esta disposible en el sitio web de WinCvs en www.wincvs.org. Para usar la interface de lineas de comandos embebida en el producto debe obtener Tcl/Tk 8.1.1 en www.scriptics.com.

Para empezar la instalacion de WinCvs, desempaquete en un directorio temporal el fichero zip descargado. Ejecute el programa de instalación (Setup.exe). Instale el software en el directorio Archivos de Programa\GNU\WinCvs 1.1 en un disco local (preferentemente C:).

Para instalar Tck/Tk, ejecute el fichero tcl811.exe descargado de scriptics.com. Instale en el directorio por defecto que sugiera el programa de instalacion.

Si quiere un editor realmente bueno, instale EmEditor (eme200ei.exe). Es un producto shareware, por lo que tendrá que abonar los $25 que cuesta si despues quiere seguir usándolo. Hay mas información sobre EmEditor y su sistema de licencia en la web http://www.nifty.ne.jp/forum/femsoft/index-e.htm. Tambien puede usar cualquier otro editor que prefiera.

Si quiere una utilidad de comparación de ficheros gratuita instale ExamDiff (examdf16.zip). Si quiere mas caracteristicas instale la version shareware ExamDiff Pro (edpro21c.zip), en lugar de ExamDiff. ExamDiff Pro cuesta $25 la licencia. Mas informacion sobre ExamDiff y ExamDiff Pro está disponible en el sitio web http://www.nisnevich.com.

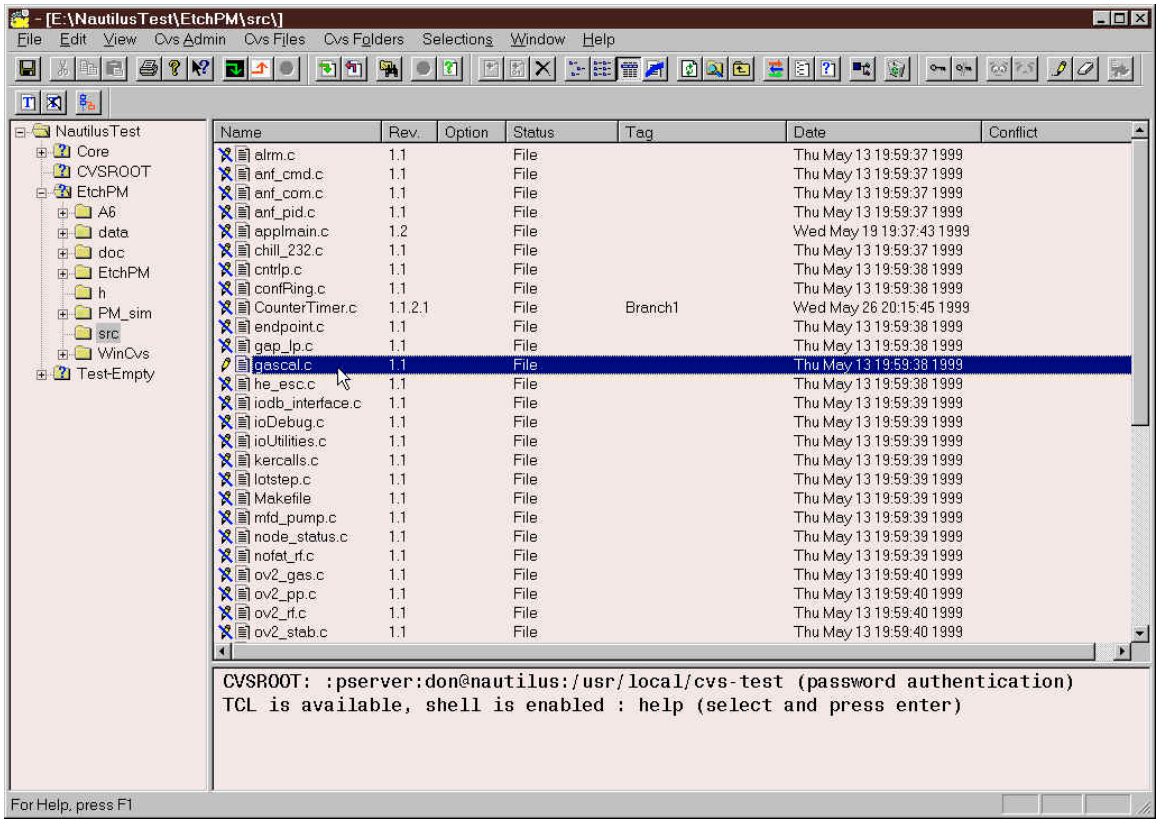
WinCvs se añadirá una entrada en el Menú Inicio, pero si lo desea puede crear ademas un acceso directo a Archivos de Programa/GNU/WinCvs 1.1/wincvs.exe en el escritorio.

Sección 3 – Guía para principiantes de WinCvs

3.1 Estableciendo la Carpeta Raíz del Area de Trabajo (Work Area).

Antes de ejecutar WinCvs cree una carpeta que será la raíz de su area de trabajo en local. Por supuesto puede manejar multiples areas de trabajo con WinCvs, pero se le hará mas fácil el seguir estas instrucciones si crea primero su raíz del area de trabajo.

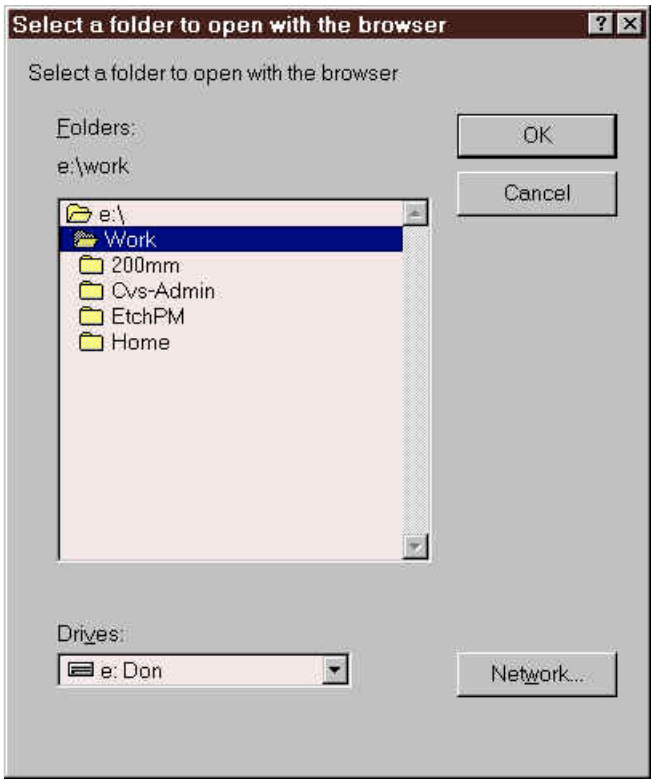
Cuando ejecute WinCvs, lo primero que verá es la pantalla principal del browser. En este ejemplo se ve un area de trabajo seleccionada como actual, por lo cual, lo que muestra el browser difiere de lo que usted verá cuando ejecute WinCvs por vez primera.



Lo primero que debería hacer es cambiar la carpeta raiz del browser a la carpeta que usará como raiz de su area de trabajo. La raiz del browser se puede cambiar tanto desde el menú Cvs Folders->Macros folder->Change Root como desde el icono de los “binoculares” en la barra de herramientas:

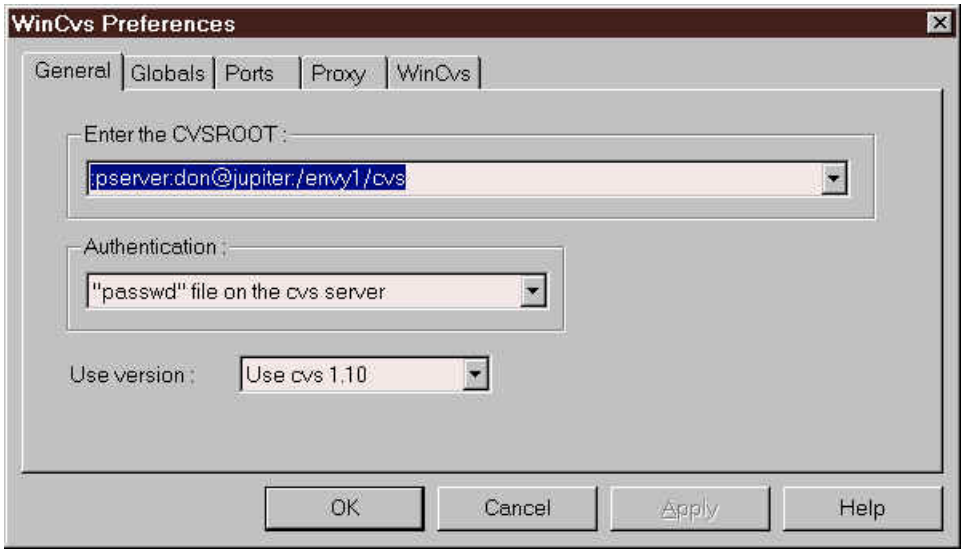
En ambos casos se abrirá un panel donde puede seleccionar la carpeta a abrir con el browser. Localice la carpeta que desee usar como raiz de su area de trabajo y haga doble-click en ella para ver el icono de carpeta abierta. Si solamente selecciona la carpeta con un solo click, seleccionará la carpeta padre en lugar de la deseada. Esto es una característica no deseada (bug) de todos los comandos de WinCvs que usan esta forma de seleccionar carpetas.

En el siguiente ejemplo, la raiz del area de trabajo se establecera a E:\Work. Nótese que la carpeta “Work” esta seleccionada y que muestra el icono de carpeta abierta.



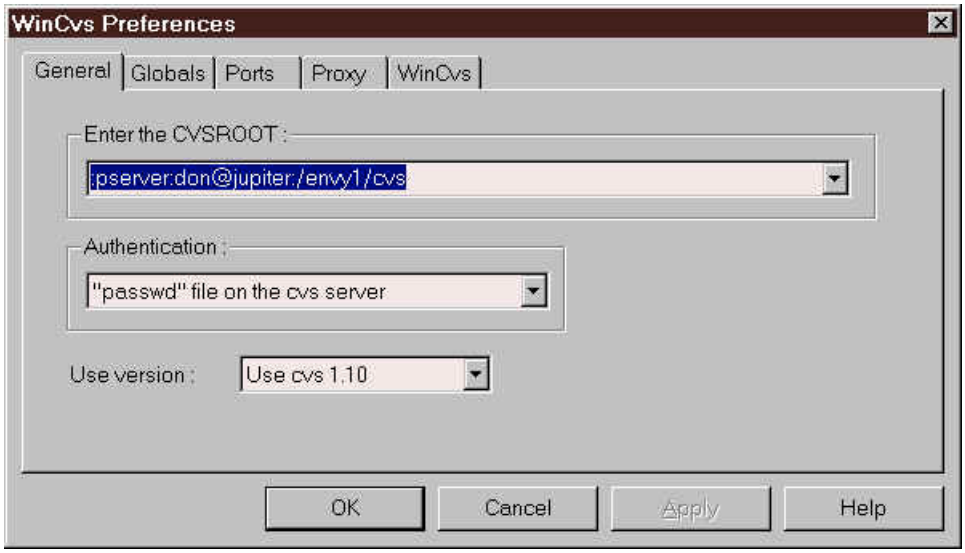
3.2 Estableciendo las preferencias (Preferences) en WinCvs

Abra el panel de preferencias de WinCvs seleccionando Cvs Admin->Preferences en el menú principal. Como se ve, hay cinco paneles de preferencias en WinCvs. El panel General es el mas importante y el que se usa con mas frecuencia.



3.2.1 Panel de preferencias General

Hay tres campos en el panel de preferencias General :



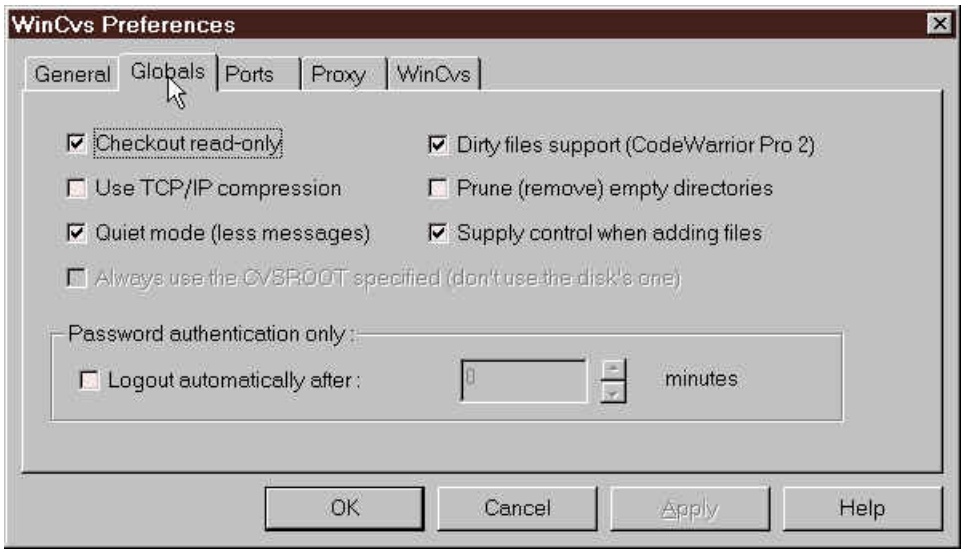
Establezca el CVSROOT a cualquier cosa similar al ejemplo de mas arriba:
:pserver:don@jupiter:/envy1/cvs

En este ejemplo, don es el nombre de usuario en el dominio local que será usado para acceder al repositorio /envy1/cvs en el servidor jupiter. NOTA: debe haber una entrada en C:\WINNT\system32\drivers\etc\Hosts definiendo la dirección IP del servidor (jupiter en este caso):
nnn.nnn.nnn.nnn jupiter

Establezca el campo Authentication a "passwd" file on the cvs server. Establezca el campo Use version a Use cvs 1.10.

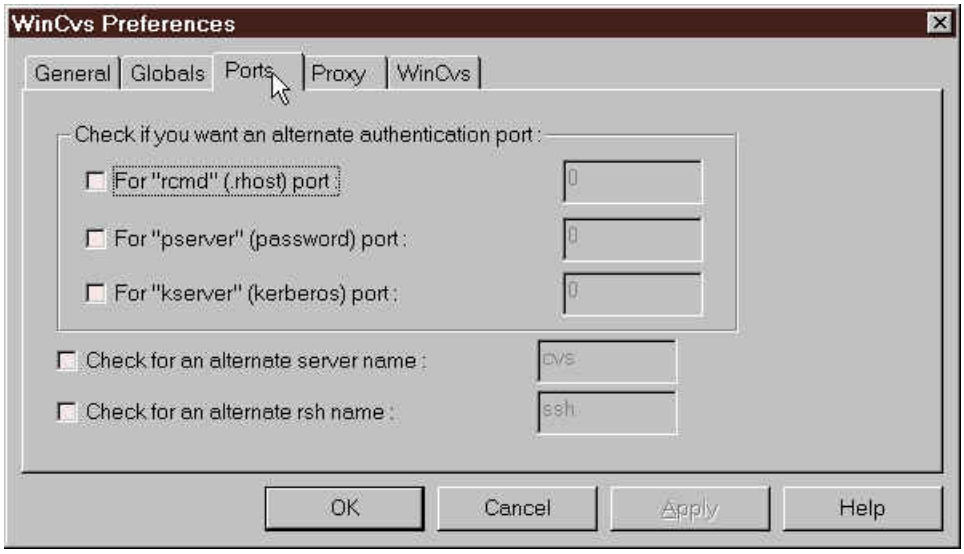
3.2.2 Panel de preferencias Globals

Establezca las opciones del panel de preferencias Globals como se muestra a continuacion, a no ser que este seguro de lo que está haciendo.



3.2.3 Panel de preferencias Ports

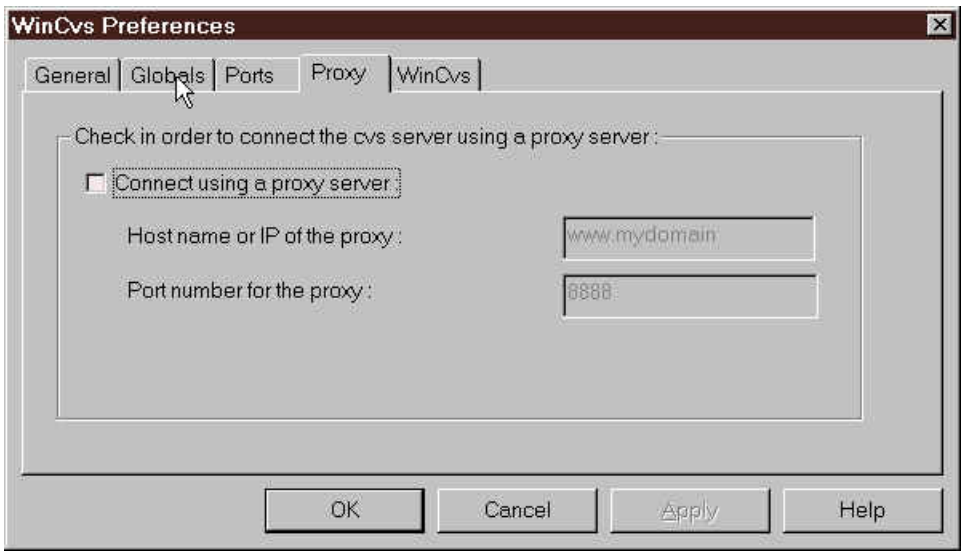
Hay cinco campos en el panel de preferencias Ports:



Normalmente se usan los valores por defecto de estos campos.

3.2.4 Panel de preferencias Proxy

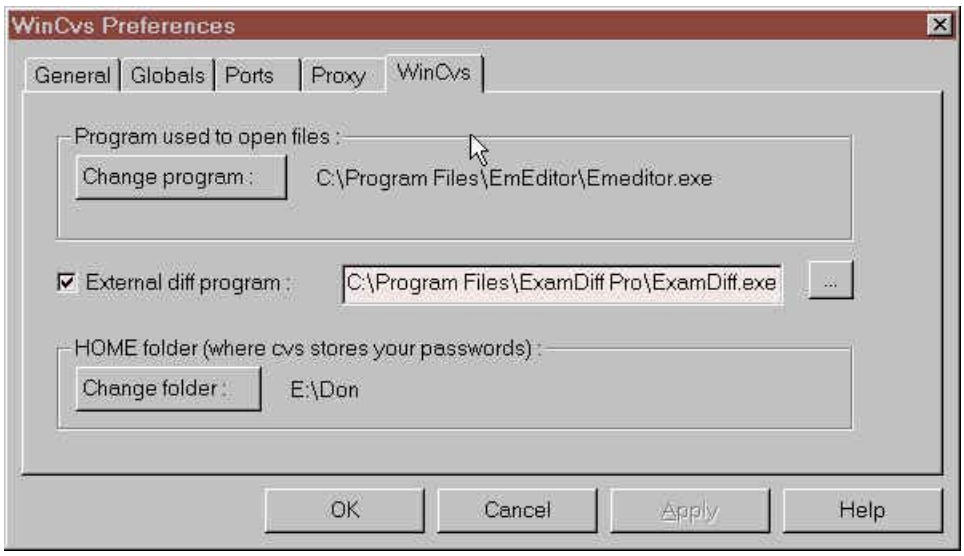
Hay dos campos en el panel de preferencias Proxy :



Este panel es necesario para utilizar cvs a traves de un servidor proxy.

3.2.5 Panel de preferencias WinCvs

Hay tres cosas que establecer en el panel de preferencias WinCvs :



Establezca el program used to open files a su editor favorito. WinCvs ejecutara este programa cuando haga doble-click en cualquir fichero mostrado en el browser que no tenga un tipo de fichero asociado en Windows.

Si quiere usar un programa para ver las diferencias entre ficheros graficamente, marque la casilla y use el botón para localizar su programa favorito que realice esta tarea. Notese que por alguna razon, WinCvs solo ejecuta el programa seleccionado cuando se ejecuta diff desde el modo "graph" (grafico). El modo "graph" se explicara en una seccion posterior.

Establezca HOME folder apuntando a una carpeta donde CVS guardará las informaciones del username y password encriptado. Esta puede ser la misma carpeta con la instalacion de CVS (Archivos de Programa\GNU\WinCvs 1.1) o su carpeta "home" si tiene una en su entorno Windows NT. Si especifica este directorio separadamente del area de trabajo podra añadir, modificar o borrar areas de trabajo solamente con uncomando de login.

3.3 Haciendo log in en el servidor

Antes de que cvs pueda ejecutar ninguna operación sobre ficheros, el usuario debe hacer log in en cvs. Haga log in en cvs seleccionando Cvs Admin->Login… en el menu principal. Se le solicitará una password (la password asociada a su usuario en el dominio local, como se especificó en CVSROOT - Sección 3.2.1):

Introduzca su password y haga click en el botón OK. Si pudo hacer log in sin problemas vera un texto similar al siguiente en la ventana de status:

```
cvs -q login
(Logging in to don@nautilus)

*****CVS exited normally with code 0*****
```

Si no pudo hacer log in y por tanto no pudo entrar en el servidor verá un texto similar al siguiente en la ventana de status:

```
cvs -q login
(Logging in to don@nautilus)
cvs [login aborted]: authorization failed: server nautilus rejected access

*****CVS exited normally with code 1*****
```

Si intenta usar otros comandos de CVS antes de hacer log in en el servidor, los comandos fallarán y vera un similar a este en la ventana status:

```
cvs import: could not open E:\Don/.cvspass: No such file or directory
cvs [import aborted]: use "cvs login" to log in first
```

3.4 Haciendo Check Out de un módulo

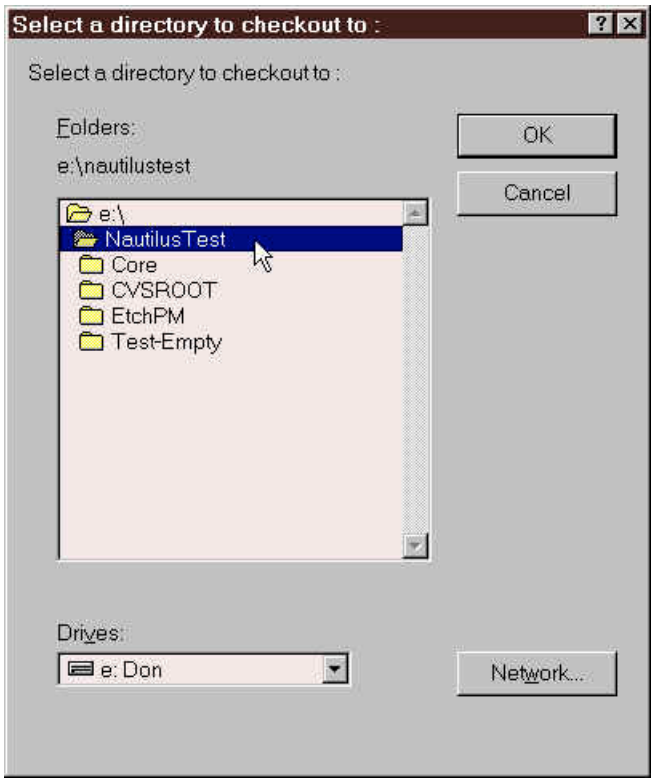
CVS organiza los repositorios en forma de módulos. Un modulo es una jerarquía de carpetas y ficheros empezando en cualquier carpeta en la jerarquía del repositorio. Todo repositorio tiene un modulo llamado CVSROOT que almacena los ficheros administrativos. Es una buena práctica dejar que estos ficheros los mantenga un administrador. Solo el administrador debería poder añadir nuevos módulos al repositorio.

Para hacer checkout de un modulo del repositorio al PC local hay que conocer el nombre del modulo. Si no conoce el nombre del módulo seleccione Cvs Admin->Macros admin->List the modules on the server en el menú principal. Este comando le mostrará la lista de modulos en la ventana de status de WinCvs:

```
*****CVS exited normally with code 0*****

Core          Core
EtchPM        EtchPM
```

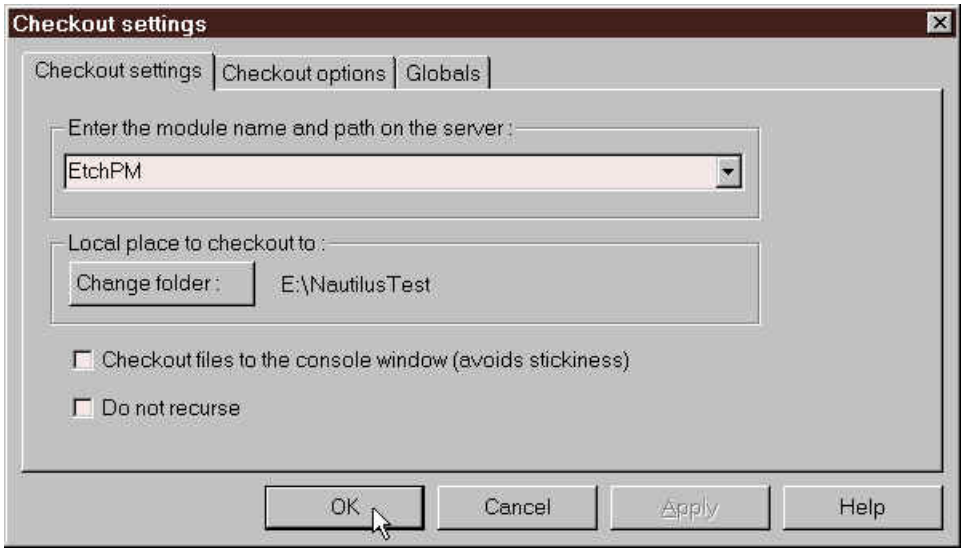
La primera entrada de cada línea es el nombre del módulo y la segunda el nombre del directorio raiz del modulo respecto a la raíz del repositorio.

Una vez que se conoce el nombre del modulo, se puede hacer checkout del mismo para colocarlo en el area local de trabajo seleccionando Cvs Admin->Checkout module… en el menú principal. Se visualizará un panel para permitirle seleccionar la carpeta destino (area de trabajo local). Vaya a la carpeta deseada y haga doble-click en ella para que se seleccione y se muestre el icono de carpeta abierta. Si accidentalmente selecciona la carpeta pero no hace doble click, se hará el checkout del modulo en la carpeta padre. En el ejemplo siguiente se hará checkout del modulo en la carpeta Nautilus Test.

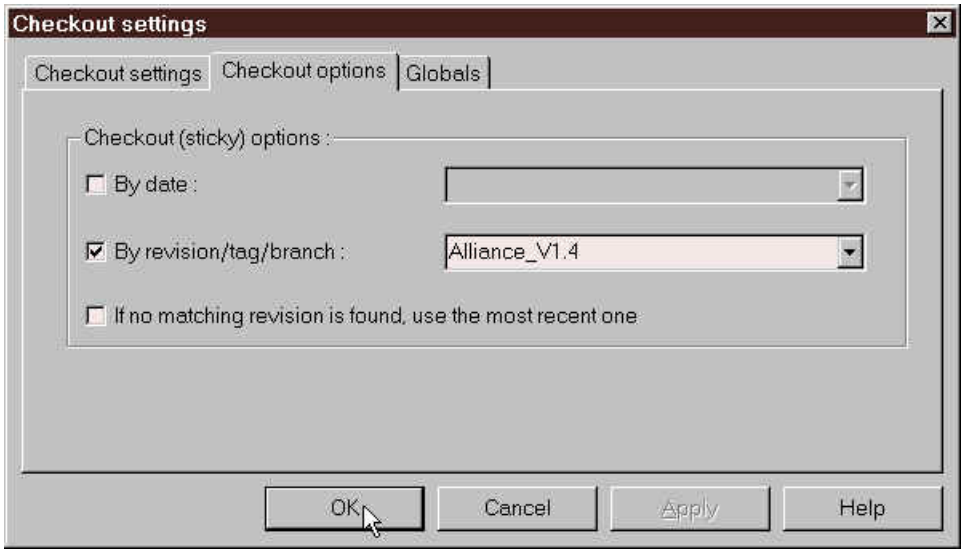


Una vez que haya seleccionado la carpeta destino y haga click en el boton OK, se visualizará el panel Checkout settings. Puede introducir el nombre del módulo en el campo correspondiente (EtchPM en la figura siguiente). Si es necesario, la carpeta destino tambien puede ser modificada desde este panel.

Pulsando en OK en este momento hará checkout de la última versión del módulo para la vifurcación principal (cvs se refiere a esta como el trunk).



Si se desea una versión vifurcada (branched) o etiquetada (tagged) de un módulo, se puede especificar el nombre de la bifurcación (branch) o la etiqueta (tag) en el panel Checkout options :



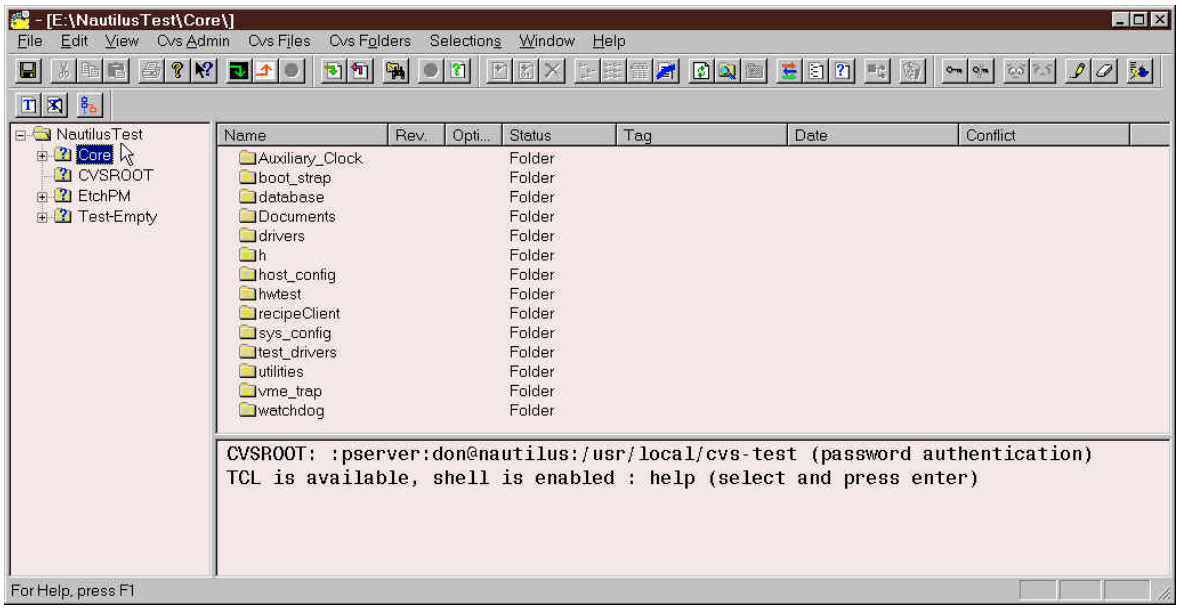
En el ejemplo anterior se hará checkout de la última versión de todos los ficheros del módulo EtchPM de la bifurcación (branch) Alliance_V1.4en lugar de hacerse checkout de la última versión de la bifurcación principal (trunk). Nótese que también es posible seleccionar la la version para que se quiere hacer el checkout especificando una fecha.

Una vez que ha establecido el nombre del modulo y las demas opciones, se hará checkout del módulo cuando se pulse el botón OK. El progreso y el resultado del checkout se pueden ver en la ventana de status:

```
cvs -q checkout EtchPM (in directory E:\NautilusTest)
U EtchPM/Makefile
U EtchPM/EtchPM/Makefile
U EtchPM/EtchPM/data/Makefile
U EtchPM/EtchPM/data/A2/EtchPM.script
U EtchPM/EtchPM/data/A2/Makefile
U EtchPM/EtchPM/data/A2/Alarms/EtchPM.almdef
.
.
.
U EtchPM/src/tasks.c
U EtchPM/src/tempctrl.c
U EtchPM/src/wat_cntrl.c

*****CVS exited normally with code 0*****
```

Desafortunadamente, no porque haya hecho checout de un módulo aparecerá este en el panel izquierdo del browser, en la vista de carpetas. Utilice la tecla F5 o el botón derecho del ratón (Reload View) para actualizar el display, o pulse en el icono Refresh View de la barra de herramientas:

En el ejemplo siguiente se ha hecho checkout de 4 módulos en el area de trabajo Nautilus Test, y la carpeta Core esta seleccionada en la vista de carpetas en el panel izquierdo. Las subcarpetas de Core son visibles en la vista de ficheros en el panel de la derecha.



3.5 Actualizando (update) un área de trabajo

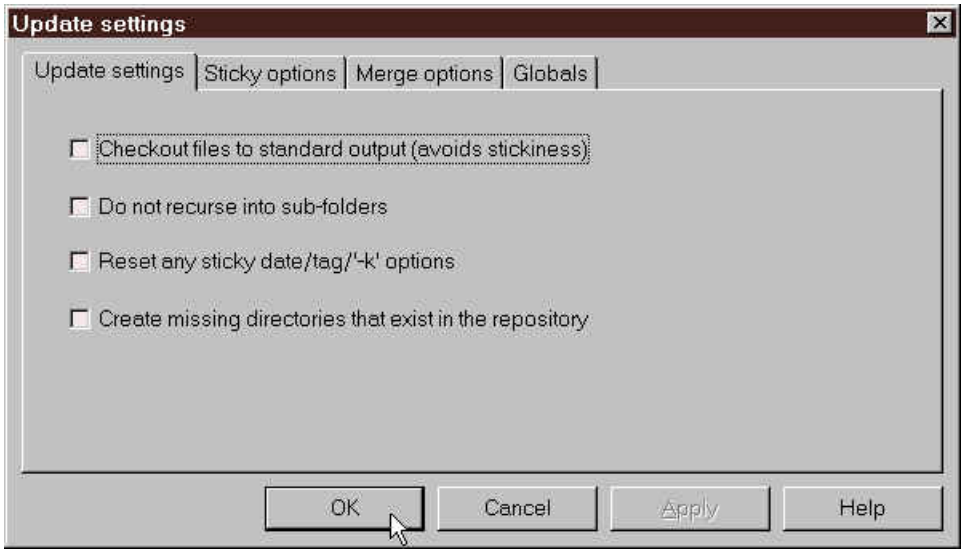
Una vez que se ha creado un area de trabajo, los ficheros pueden pasar a estar desactualizados conforme otros desarrolladores hacen checkin de sus modificaciones desde sus propias areas de trabajo. El comando update de CVS permite que un area de trabajo sea actualizada a los últimos cambios que se hayan realizados en los fuentes (los ultimos checkin,s realizados). Otro uso frecuente del comando update es cambiar entre bifurcaciones (branches) y versiones etiquetadas (tagged version).

El comando update (aligual que otros muchos comandos de WinCvs) se puede invocar sobre una carpeta, un fichero o una selección (el conjunto de carpetas o ficheros que estén seleccionados en ese momento). Por tanto, al comando update se puede acceder desde los menús Cvs Files, Cvs Folders, y Selections. Posiblemente la forma más comoda de actualizar (update) un fichero o carpeta es seleccionarlo con el botón derecho del ratón, lo cual hace que además se visualice el menú Selections, y todo ello con un solo click de ratón. Si quiere usar para esto la barra de herramientas, en ella tambien hay iconos para estas dos operaciones:

Actualizar carpeta (Update Folder):

and Actualizar selección (Update Selection):

Independientemente de cómo se como se invoque el comando update, se visualizará el panel Update settings:



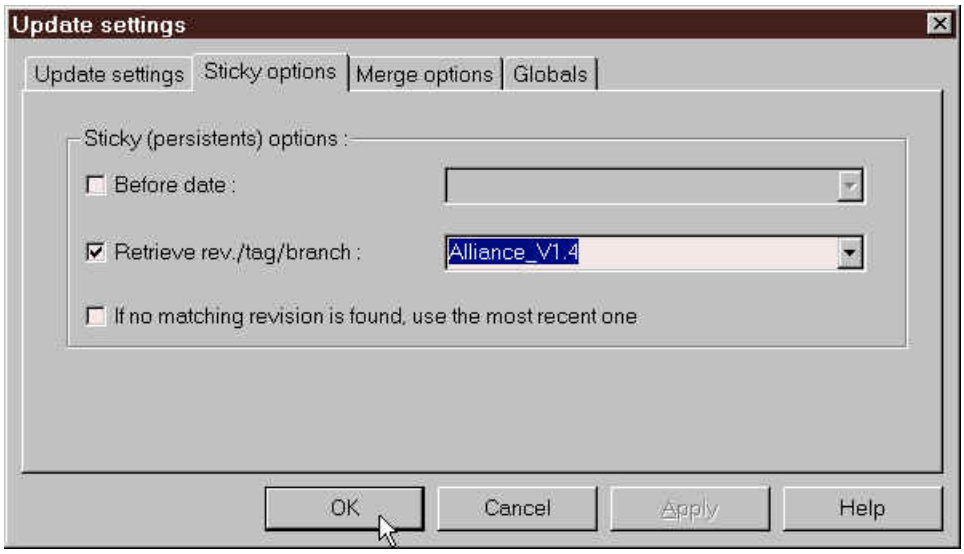
La opción Do not recurse into sub-folders puede ser útil para hacer update solamente sobre la carpeta seleccionada y no sobre todas sus subcarpetas (la recursión es siempre la opción por defecto de cvs).

La opción Reset any sticky date/tag/-k options se usa para hacer update hacia atrás a la última versión de los ficheros en la bifurcación principal. Durante el desarrollo se puede hacer update a una bifurcación

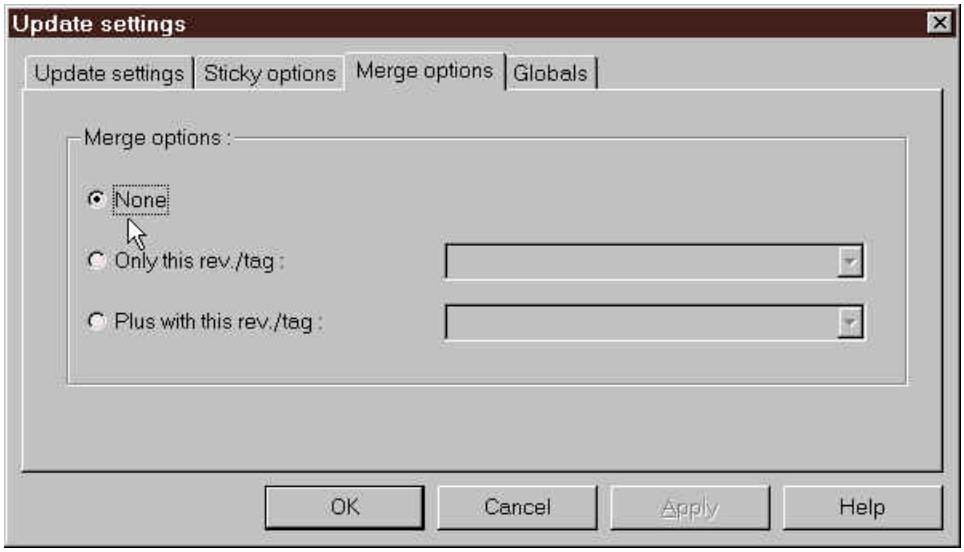
(branch), a una etiqueta (tag) o por fecha, y esto último significa obtener un area de trabajo actualizada a la bifurcación principal. Esto es lo que hace esta opción.

La opción Create missing directories that exist in the repository es útil para obtener los nuevos directorios que han sido añadidos al repositorio desde que se hizo checkout del módulo la vez anterior.

Para hacer update a una bifurcación que no es la principal, hacer update a una versión etiquetada (tagged version) o hacer update por fecha hay que usar el panel Sticky options . En el siguiente ejemplo un area de trabajo que ya existe seria actualizada a la última versión de todos los ficheros de la bifurcación Alliance_V1.4 :



En el panel Merge options hay algunas opciones más sofisticadas relacionadas con las mezclas (merge). Consulte el manual de cvs para más información sobre este panel:



3.6 Editando un fichero

Por defecto, cuando se hace un checkout de un modulo o un update del área de trabajo los ficheros obtenidos son no modificables. Para modificar un fichero hay que usar primero el comando edit de manera que el fichero se marque como modificable. En WinCvs un icono a la izquierda del nombre del fichero en la ventana del browser vista de ficheros indica el estado modificable/no modificable del fichero.

Los ficheros de solo lectura se indican con el icono:

Los ficheros modificables se indican con:

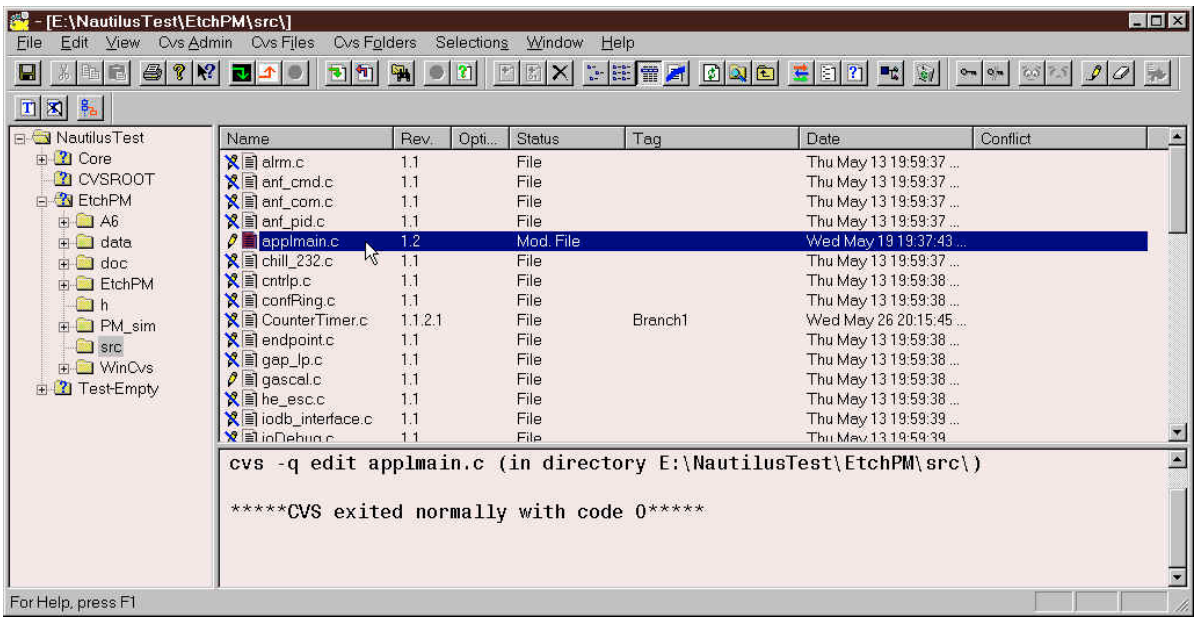
Para invocar el comando edit y hacer un fichero modificable seleccione el fichero y pulse el icono Edit selection de la barra de herramientas:

El comando edit tambien se puede ejecutar desde el menú emergente que se visualiza pulsando el boton derecho del reton, opción Monitors selection->Edit selection o desde la opcion Selections->Monitors selection->Edit selection, pero usar la barra de herramientas es mas cómodo.

Después de que el fichero ha sido marcado como modificable, este puede ser modificado haciendo solo boble-click sobre el.

Para volver a marcar el fichero como no modificable (es decir, de solo lectura), usese el comando unedit desde los menús o la barra de herramientas.

Después de modificar un fichero su icono en la ventana de vista de ficheros del browser se pondrá en rojo y la columna status cambiará a Mod. File como se muestra en el siguiente ejemplo donde fue modificado el fichero applmain.c:



3.7 Visualizar las diferencias antes de hacer commit (modo texto)

A menudo es útil tener la precaución de ejecutar diff antes de hacer commit de un fichero al repositorio. Esta sección describe los pasos necesarios para comparar un fichero a la versión que hay en el area de trabajo con la versión actual del mismo que hay en el repositorio. Las diferencias pueden verse tanto en el modo texto estandar del comando diff en la ventana status como en modo gráfico (para esto último es necesario que haya establecido cual es el programa externo con el que verá las diferencias, como se explicó en la Sección 3.2.5). Esta sección se refiere al formato texto en que se visualizan las diferencias. Vea la siguiente sección (Sección 3.8) para un ejemplo del comando diff en modo gráfico.

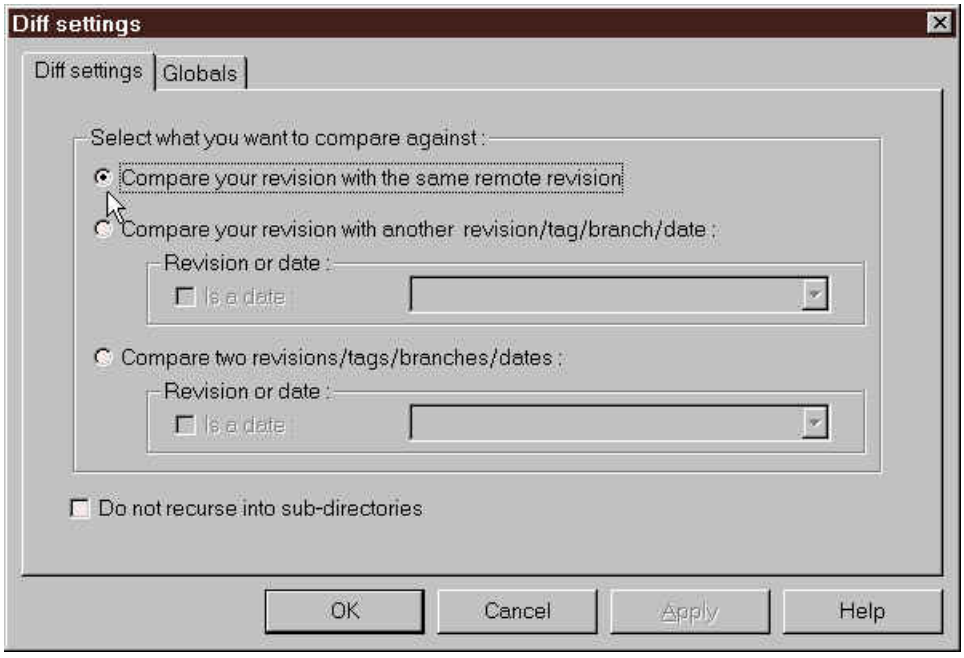
El comando diff se puede ejecutar de cualquiera de estas tres formas:

1) Seleccione el fichero usando el botón izquierdo del ratón y utilice el menú Selections->Diff selection

2) Seleccione el fichero usando el botón derecho del ratón y use la opción Diff selection

3) Seleccione el fichero usando el botón izquierdo del ratón y use la barra de herramientas:

Una vez que se invoca el comando diff, se visualizará el panel Diff settings :



El panel Diff settings tiene varias opciones, pero solo la primera opción tiene sentido en este ejemplo (porque lo que queremos hacer es comparar nuestra versión contra la del repositorio). Asegúrese de que selecciona la primera opción como se muestra arriba y haga click en el botón OK.

La versión en formato texto de las diferencias entre ficheros (si las hay) se visualizará ahora en la ventana de status como se muestra aquí:

```
cvs -q diff applmain.c (in directory E:\NautilusTest\EtchPM\src)
Index: applmain.c
===================================================================
RCS file: /usr/local/cvs-test/EtchPM/src/applmain.c,v
retrieving revision 1.1
diff -r1.1 applmain.c
988,993c988,994
< int             Metal960OPTX_installed = FALSE;
< int             single_plane_transfer = FALSE;
< int             chill_232_present = FALSE;
< int             pump_232_present = FALSE;
< int             lonwork_present = FALSE;
< int             scrubber_232_present = FALSE;
---
> int             Metal960OPTX_installed = TRUE;
> int             single_plane_transfer = TRUE;
> int             chill_232_present = TRUE;
> int             pump_232_present = TRUE;
> int             lonwork_present = TRUE;
> int             scrubber_232_present = TRUE;
> int             advanced_endpoint_installed = TRUE;

*****CVS exited normally with code 1*****
```

El texto de la ventana de status de WinCvs puede ser seleccionado, salvado a un fichero, o impreso simplemente activando la ventana status (haciendo click en cualquier parte de ella) y usando los menús Edit y File. También se puede borrar el texto de esta ventana con Edit->Select All y Edit->Cut (cortarlo).

3.8 Visualizando las diferencias antes de hacer commit (modo gráfico)

Como se mencionó en la Sección 3.7, a menudo es útil tener la precaución de ejecutar diff antes de hacer commit de un fichero al repositorio. Esta sección describe los pasos necesarios para comparar la versión de un fichero en el area de trabajo contra la versión actual del mismo en el repositorio. Las diferencias pueden verse tanto en el modo texto estandar del comando diff en la ventana status, como en modo gráfico (para esto último es necesario que haya establecido cual es el programa externo con el que verá las diferencias, como se explicó en la Sección 3.2.5). Esta sección se refiere al modo gráfico en que se visualizan las diferencias. Vea la anterior sección (Sección 3.7) para un ejemplo del comando diff en modo texto.

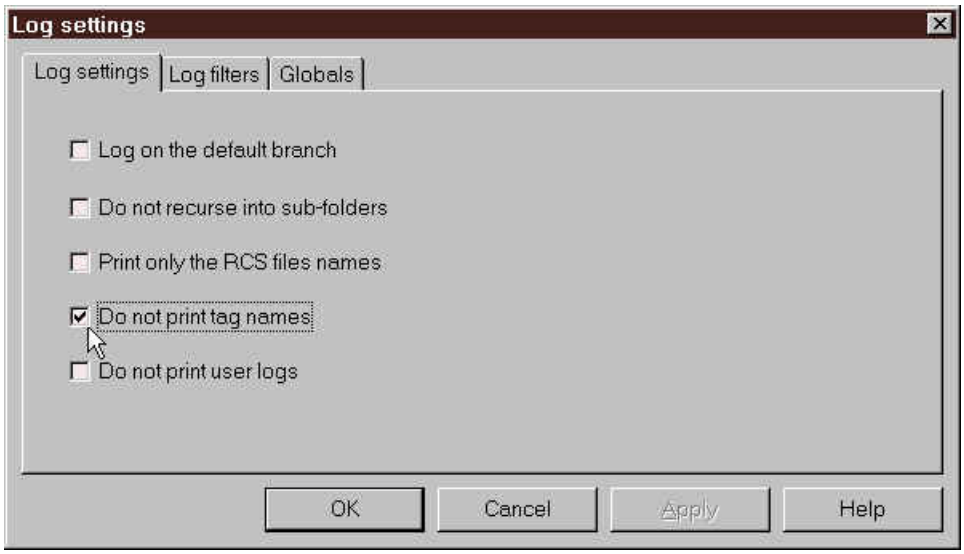
En WinCvs el programa gráfico que muestra las diferencias solo se puede usar desde el modo graph. El modo graph de WinCvs muestra todas las versiones de un fichero en formato gráfico. El gráfico puede ser generado de cualquiera de estas tres formas:

1) Seleccione el fichero usando el botón izquierdo del ratón y el menú Selections->Graph selection

2) Seleccione el fichero usando el botón derecho del ratón y despues Graph selection

3) Seleccione el fichero usando el botón izquierdo del ratón después la barra de herramientas:

Una vez que se ha invocado el modo graph, se visualiza el panel Log settings :



Hay muchas opciones en los paneles Log settings que el usuario puede explorar cuando le sea necesario utilizarlas. En este ejemplo se ha seleccionado la opción Do not print tag names, ya que puede ser una opción usada con mucha frecuencia para evitar el que haya demasiada información en el gráfico que se visualice despues. Los nombres de etiquetas en cvs (tag names) se refieren a los nombres de bifurcaciones (que son útiles de ver), pero tambien se refieren a las versiones etiquetadas (tagged versions) que probablemente no les interesan a los desarrolladores.

Una vez que haya seleccionado las opciones deseadas haga click en el botón OK para visualizar el gráfico.

El gráfico de este ejemplo muestra que para applmain.c hay actualmente dos versiones en el repositorio:

Para visualizar las diferencias entre la versión del fichero en el area de trabajo y la versión del fichero en el repositorio (1.2), haga click primero en la versión mas reciente del fichero en el repositorio, como se

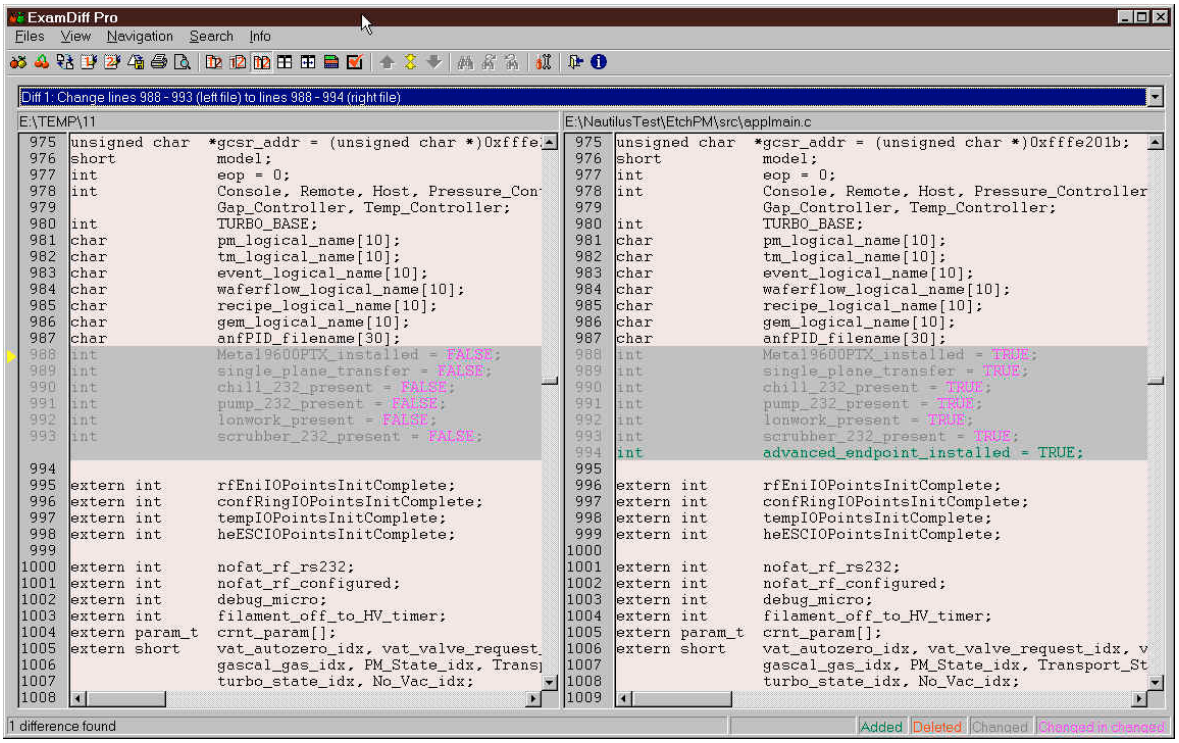
muestra arriba. Notese que en la mitad inferior de la ventana se visualiza información acerca de la versión seleccionada.. La informacion que se muestra es el autor, la fecha-hora de modificación, y el mensaje de log para la versión seleccionada del fichero.

Ahora puede visualizar las diferencias en modo gráfico, tanto desde el menú Graph usando Graph->Diff como desde la barra de herramientas del modo graph:

El programa externo con el que visualizar las diferencias (vea como establecerlo en la sección 3.2.5) se ejecutará como se muestra en el siguiente ejemplo:

Nótese que el programa externo usado en el ejemplo anterior para ver las diferencias (ExamDiff Pro) tiene muchas opciones, incluida la posibilidad de configurar los colores, ignorar los espacios enblanco, etc,... (menú View->Options…). También hay una opción para salvar las diferencias en formato UNIX (menú File->Save Unix Diff File…).

Tenga cuidado al cerrar la ventana del modo graph despues de ver las diferencias, o podría acabar cerrando completamente el programa WinCvs. Una forma mejor de usar WinCvs puede ser usar el modo Tile, que se puede seleccionar con Window->Tile.



3.9 Hacier commit con un fichero o una carpeta

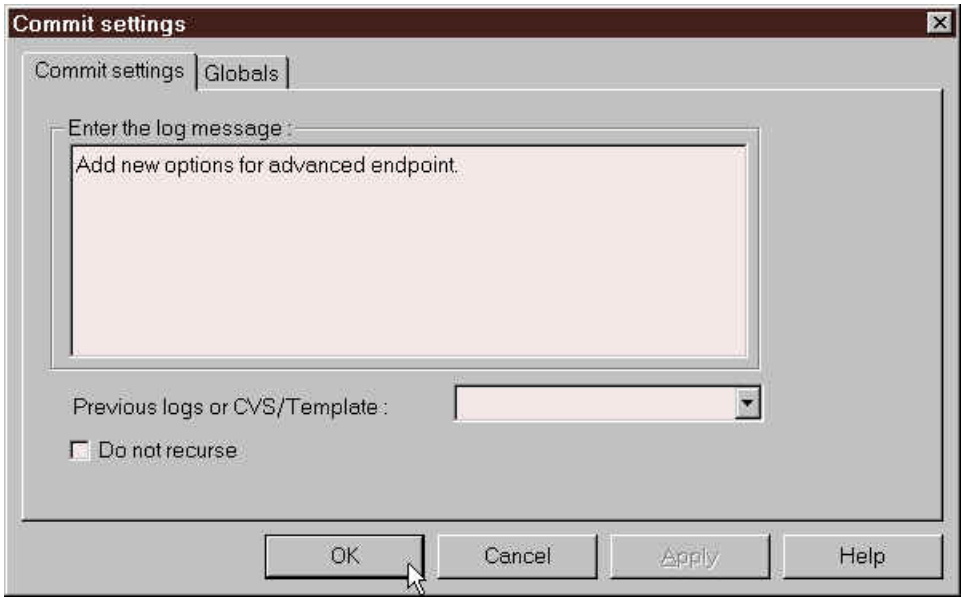
Despues de que se hayan editado uno o mas ficheros, sepuede hacer commit al repositorio con ficheros individuales o con un arbol entero de carpetas usando el comando commit. Se puede invocar el comando commit de varias formas:

1) Seleccione un fichero o carpeta con el botón izquierdo del ratón y seleccione la opción de menú Selections->Commit selection

2) Seleccione un fichero o carpeta con el botón derecho del raton y despues use la opción Commit selection

3) Seleccione un fichero o carpeta con el botón izquierdo del ratón y despues use la barra de herramientas:

Una vez que se ha invocado el comando commit, se visualizará el panel Commit settings :



Teclee un mensaje de log y pulse el botón OK para hacer commit de los cambios. Si ha seleccionado para hacer commit una carpeta y no desea que también se haga commit con sus subcarpetas, marque la opción Do not recurse. Los resultados de la operación de commit se visualizarán en la ventana status como a continuación:

```
cvs -q commit -m "Add new options for advanced endpoint." applmain.c (in
directory E:\NautilusTest\EtchPM\src)
Checking in applmain.c;
/usr/local/cvs-test/EtchPM/src/applmain.c,v  <--  applmain.c
new revision: 1.2; previous revision: 1.1
done

*****CVS exited normally with code 0*****
```

3.10 Añadir ficheros o carpetas al repositorio

Se puede añadir ficheros o carpetas al repositorio tanto con el comando add como con el comando import. Los programadores deberían normalmente usar el comando add para añadir ficheros o unas pocas carpetas. Hay solamente dos situaciones donde se debería usar el comando import:

- Si los ficheros a crear requieren la creación de un nuevo módulo (por ejemplo debido a que no existe previamente un módulo para los mismos). En este caso se debe usar el comando import. Para mantener una determinada convención en la forma de nombrar los módulos, es una buena idea dejar la tarea de creacion de modulos al administrador de cvs. El número de módulos en un repositorio tambien hay que controlarlo para evitar confusiones.cierta homogeneidad en los nombres de los módulos.

- Si se desea añadir una jerarquía de varios niveles a un módulo que ya existe, puede ser tedioso añadir cada nivel manualmente. WinCvs permite añadir todos los ficheros y subcarpetas de una sola carpeta en una sola operación. Realizando esta operación manualmente, habría que repetirla para cada una de las subcarpetas hasta el final de la jerarquía. Además los ficheros binarios habría que añadirlos en una operación aparte, haciendose la tarea considerablemente tediosa. En esta situaciones en mucho mas fácil usar el comando import para importar la jerarquía completa a una subcarpeta de un módulo existente. Tambien es una buena idea utilizar al administrador de cvs o pedirle ayuda cuando haya que importar ficheros de esta forma.

Si el comando add le parece el más apropiado (el caso frecuente), lea la Sección 3.10.1 para más información sobre el uso del comando add . Si necesita usar el comando import lea la sección 3.10.2 o contacte con su administrador de cvs.

3.10.1 Añadir ficheros o carpetas (a un módulo que ya existe) usando add

Antes de que se pueda añadir ficheros o carpetas a un módulo, debe existir un area de trabajo en la cual se haya hecho checkout del módulo en cuestion. Entonces se pueden crear ficheros y carpetas dentro del area de trabajo mediante cualquiera de estos procedimientos:

- crear una carpeta vacia desde el Windows NT Explorer
- copiar los ficheros desde otro lugar.
- copiar una jerarquía existente desde otro lugar (considere el uso de import, Sección 3.10.2)
- crear los ficheros usando un editor de textos, Microsoft Word o cualquier otra aplicación.

Hay tres puntos importantes que hay que mencionar relacionados con el comando add:

1) El comando add NUNCA es recursivo en cvs. Esto significa que añadir una jerarquía a un módulo ya existente es un proceso manual. Hay que añadir manualmente los ficheros y las carpetas para cada nivel de la jerarquía. Si necesita añadir una jerarquía de mas de un nivel al repositorio, considere usar el comando import (Sección 3.10.2).

2) Cvs necesita que se le diga que ficheros tratar como binarios y que ficheros tratar como ficheros de texto. Si intenta añadir un fichero binario como fichero de texto, WinCvs lo detectará y se lo advertira, pero es mas seguro añadir ficheros de texto con las versiones Add de los comandos de WinCvs y los ficheros binarios con las versiones AddBinary

3) El comando add solo añade el fichero al area de trabajo local. Despues hay que usar el comando commit (Sección 3.9) para añadir permanentemente los ficheros al repositorio.

Una vez que todos los ficheros y carpetas han sido colocados en el area de trabajo, estamos preparados para añadirlos al repositorio. Como sucede con otros comandos, WinCvs proporciona varias formas de invocar el comando add:

1) Seleccione los ficheros o carpetas usando el botón izquierdo del ratón y después la opción de menú adecuada:

   Selections->Add selection para ficheros de texto o carpetas
   Selections->Add selection binary para ficheros binarios

2) Seleccione los ficheros o carpetas usando el botón izquierdo del ratón y use el botón derecho del ratón para abrir el menú Selections . Escoja la opción de menú adecuada:

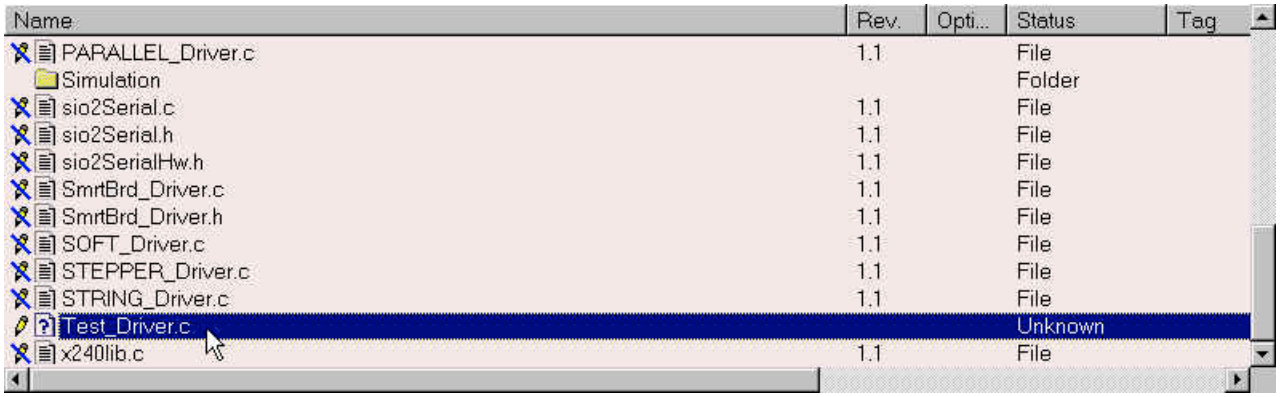
   Add selection para ficheros de texto o carpetas
   Add selection binary para ficheros binarios

3) Seleccione los ficheros o carpetas usando el botón izquierdo del ratón y use el icono adecuado de la barra de herramientas:

   Add selected (para carpetas o ficheros de texto):

   Add selected binary (para ficheros binarios):

Nótese que el fichero o carpeta seleccionado debería verse con un icono "?" y estatus Unknown como se muestra a continuación:

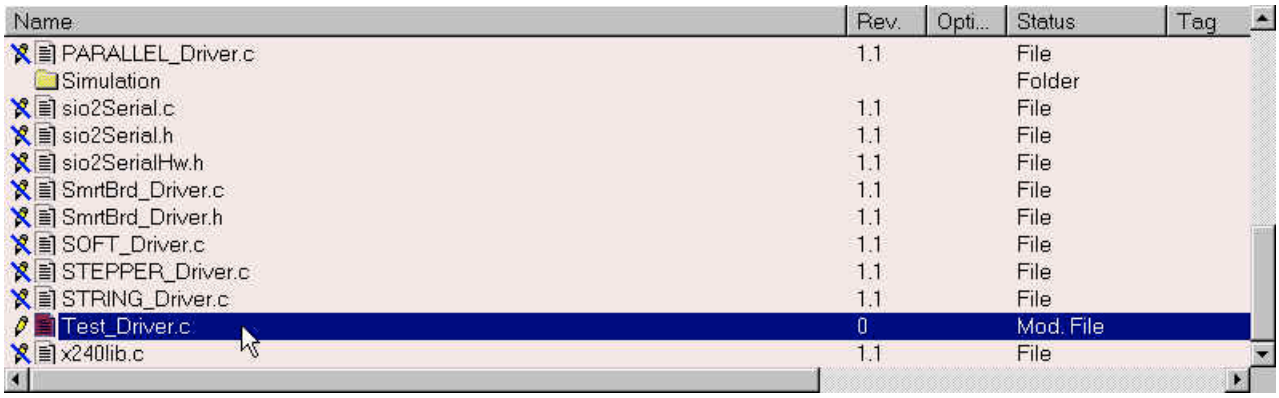


Una vez que el comando add (o add binary) se ha invocado, el fichero o carpeta será añadido al area de trabajo. El estado de la operación add se puede ver en la ventana de status de WinCvs, como se muestra a continuación:

```
cvs -q add Test_Driver.c (in directory E:\NautilusTest\Core\drivers\)
root server: use 'root commit' to add this file permanently

*****CVS exited normally with code 0*****
```

Como se ha indicado, el fichero o carpeta todavia requiere que se haga una operación de commit para añadir realmente el fichero al repositorio.

Despues de que un fichero es añadido al area de trabajo, su status cambia a “Mod File”, y la columna revision (Rev.) debería mostrarse como 0, como se muestra a continuación:



El fichero(s) añadido puede puede ahora ser editado o hacer commit con él contra el repositorio (Sección 3.9).

3.10.2 Añadir ficheros o carpetas usando import

Este comando import de cvs permite añadir una jerarquía existente de ficheros y carpetas al repositorio creando un nuevo módulo o añadiendolos a un módulo ya existente. Como se vio en la Sección 3.10, hay varias razones para que este comando deba ser usado principalmente por un administrador de cvs.

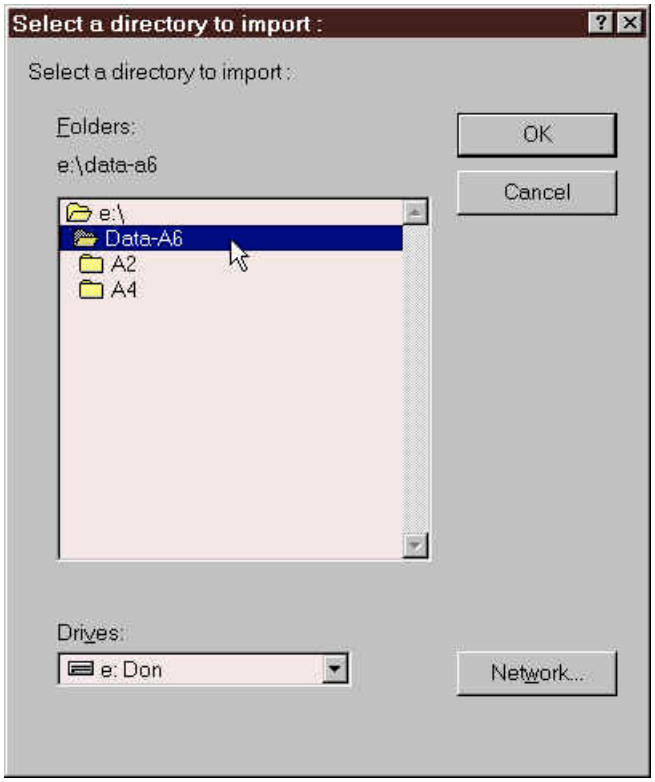
Al contrario que el comando add , la jerarquía que va a ser importada NO debe estar en el area de trabajo local ants de ejecutarse el comando import. Posteriormente, una vez que el comando import se haya completado, se puede hacer checkout de la jerarquía completa al area de trabajo local.

Hay dos ejemplos en esta sección. En el primero (Sección 3.10.2.1), una jerarquía ya existente de ficheros de texto y binarios será importada a un modulo ya existente en el repositorio. En el segundo ejemplo (Sección 3.10.2.2) una jerarquía ya existente de ficheros de texto y binarios será importada como un nuevo módulo.

3.10.2.1 Importar una jerarquía de ficheros a un módulo ya existente

Al contrario que otros comandos, WinCvs solo proporciona una forma de invocar el comando import :

Seleccione Cvs Admin->Import module… en el menú principal. Se visualizará el panel Select:



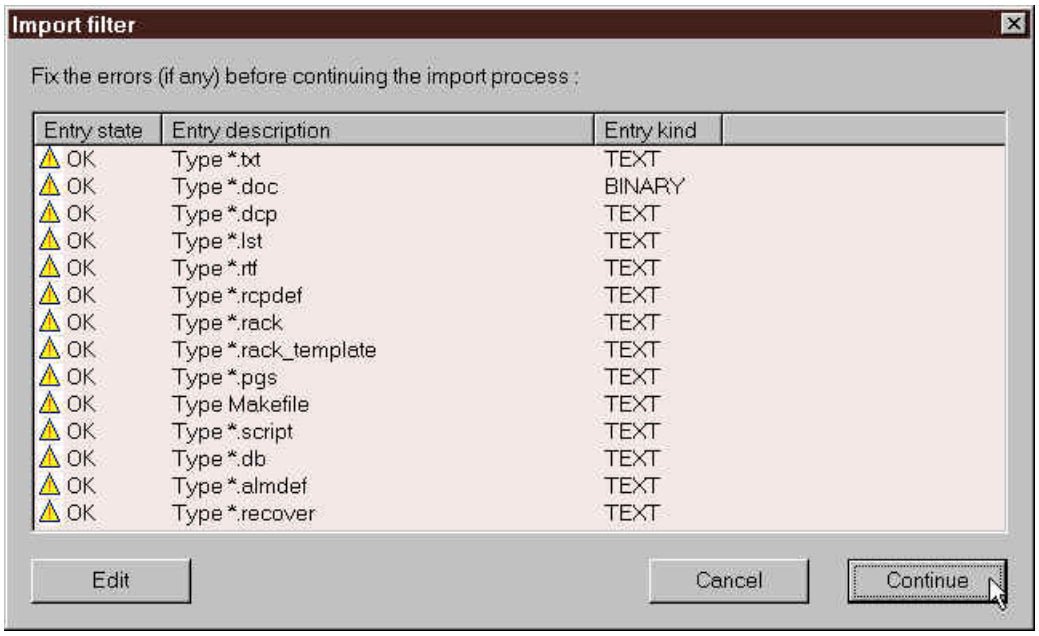
Localice la carpeta que quiere importar y haga boble-click en ella para ver el icono de la misma como una carpeta abierta. Si selecciona la carpeta con un solo click de ratón, se seleccionará la carpeta padre y todas sus subcarpetas en lugar de la deseada. Esto es un bug de todos los comandos de WinCvs que usan esta forma de selección de carpetas. En el ejemplo de arriba la carpeta Data-A6 esta seleccionada y su icono se muestra como una carpeta abierta.

Después de hacer doble-click sobre la carpeta a importar, pulse OK. En ese momento WinCvs examinará la jerarquía para determinar cuantos ficheros se van a importar y determinar el tipo de cada uno de ellos (texto o binario). A este proceso se le llama filtrado (filtering), y puede tomar algún tiempo si la jerarquía de ficheros es grande. Dirante este tiempo la ventana de status de WinCvs visualizará una linea por cada carpeta examinada, como se muestra a continuación:

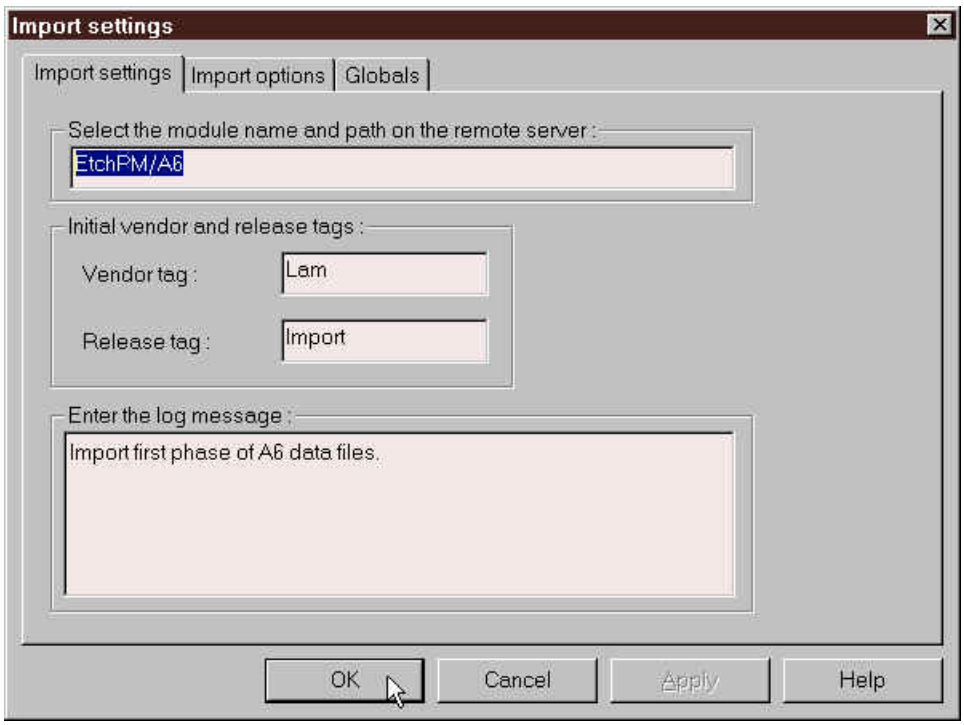
```
Filtering 'E:\Data-A6'...
Filtering 'E:\Data-A6\A2'...
Filtering 'E:\Data-A6\A2\Alarms'...
Filtering 'E:\Data-A6\A2\db'...
Filtering 'E:\Data-A6\A2\pg'...
Filtering 'E:\Data-A6\A2\Rack'...
Filtering 'E:\Data-A6\A2\Recipe'...
Filtering 'E:\Data-A6\A4'...
Filtering 'E:\Data-A6\A4\Alarms'...
Filtering 'E:\Data-A6\A4\db'...
Filtering 'E:\Data-A6\A4\dcp'...
Filtering 'E:\Data-A6\A4\pg'...
Filtering 'E:\Data-A6\A4\Rack'...
Filtering 'E:\Data-A6\A4\Recipe'...
```

Cuando el proceso de filtrado se haya completado, se visualizará el panel Import filter con los resultados del proceso de filtrado. Este panel lista las extensiones de ficheros encontradas en la jerarquía a importar y el tipo que cvs le asignará al añadir el fichero (TEXT o BINARY). En el siguiente ejemplo no hay errores ni warnings. Sin embargo, en algunos casos WinCvs puede encontrar un fichero con una extensión que normalmente se asigna a los ficheros de texto aunque el fichero en cuestión aparezca como binario. Los errores pueden ser ignorados (de cualquir forma, WinCvs hará normalmente lo correcto) o se puede usar el botón Edit para cambiar los valores que WinCvs haya establecido.

Después de que haya revisado el panel Import filter pulse en Continue para empezar el proceso del import:



Despues de pulsar el botón Continue se visualizará el panel Import settings:



En este ejemplo, se está creando una nueva carpeta A6 bajo el modulo EtchPM ya existente. Nótese que la carpeta Data-A6 importada no aparecerá en el repositorio. Todos los ficheros y subcarpetas de Data-A6 serán importados al repositorio bajo la carpeta A6 . Este ejemplo pretende mostrar que una carpeta puede ser importada con un nombre diferente. Hay que introducir cuatro campos antes de iniciar el import :

- Teclee el nombre de módulo y path como se muestra arriba. Notese que cvs REQUIERE que se usen caracteres "/" (forward slash) para separar nombres de directorios. Si se usase un carácter "\" (backslash) se crearía un nuevo módulo en la raiz del repositorio llamado EtchPM\A6, en lugar de un nuevo directorio A6 bajo el dierctorio EtchPM ya existente.

- Introduzca cualquier cosa en los campos Vendor tag y Release tag . Estas etiquetas raramente se usan y pueden ser borradas después.. Por tanto, puede introducir cosas como VTAG y RTAG, y borrarlas despues de que el import se haya completado. Desafortunadamente, CVS las necesita para ejecutar el comando import. Se pueden borrar despues de que se haya hecho el import seleccionando la carpeta de mas arriba en la jerarquía y usando Delete tag, al que se puede acceder desde el menú Selections->Tag selection .

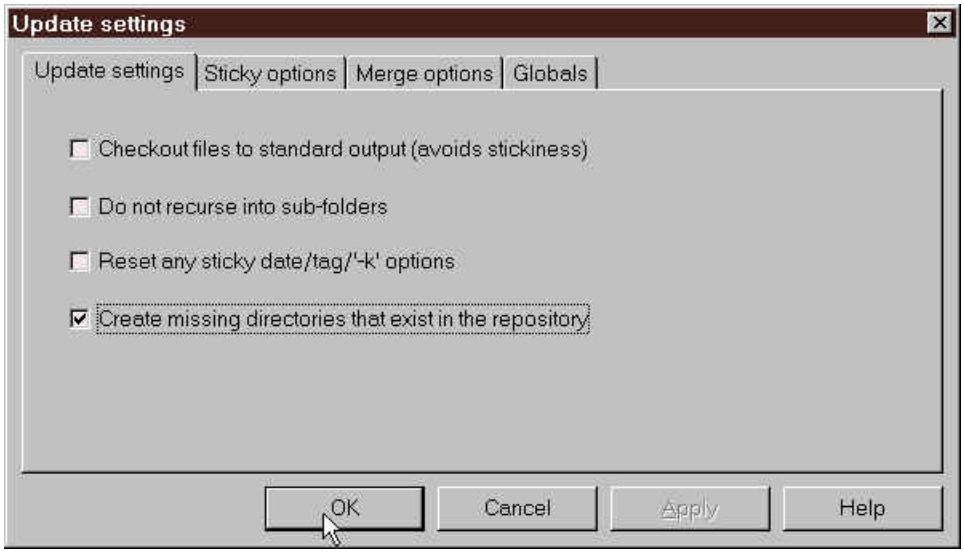
- Introduzca un mensaje de log adecuado (este mensaje será parte de la información de log asociada a los ficheros para la version inicial 1.1), que aparecerá en el fichero si la palabra clave $Log figura en el (solo ficheros de texto).

Despues de haber introducido valores en todos los campos pulse el botón OK para iniciar el import .

Durante el import se ejecuta el comando de cvs del mismo nombre y la salida del mismo se visualiza en la ventana de status de WinCvs, como se muestra en el siguiente ejemplo:

```
cvs -q import -I ! -I CVS -W "*.doc -k 'b'" -m "Import first phase of A6
data files." EtchPM/A6 AVendor Import (in directory E:\Data-A6)
N EtchPM/A6/Endpoint.doc
N EtchPM/A6/EtchPM.doc
N EtchPM/A6/Makefile
N EtchPM/A6/README.txt
N EtchPM/A6/A2/EtchPM.script
N EtchPM/A6/A2/Alarms/EtchPM.almdef
.
.
.
N EtchPM/A6/A4/Rack/SILYLATION.rack
N EtchPM/A6/A4/Recipe/EtchPM.rcpdef
N EtchPM/A6/A4/Recipe/SilyPM.rcpdef

No conflicts created by this import


*****CVS exited normally with code 0*****
```

Ahora que los ficheros han sido importados al repositorio se puede hacer checkout de los mismos al area de trabajo ejecutando el comando update sobre la carpeta padre. En este ejemplo, se ha seleccionado el directorio EtchPM y despues update con el boton derecho del ratón. Se visualiza el panel Update settings:



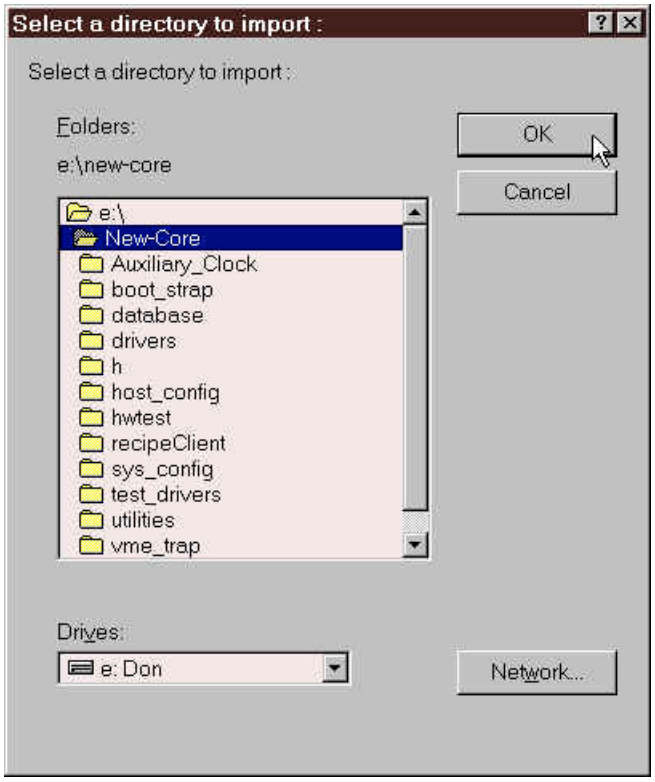
Active la opción Create missing directories that exist in the repository y pulse el botón OK para obtener en el area de trabajo una working copy (copia de trabajo) de los ficheros importados.

Si desea que los ficheros que acaba de importar sean accesiblen en el repositorio como un módulo separado, vea la Sección 4.1. Puede ser útil permitir a los usuarios hacer checkoput de una parte del arbol de un módulo en lugar de obligar a que el checkout se haga siempre del arbol completo.

3.10.2.2 Importar una jerarquía de ficheros a un nuevo módulo

Al contrario que con otros comandos, WinCvs tiene una única forma de invocar el comando import :

Seleccione Cvs Admin->Import module… en el menú WinCvs. Se visualizará el panel Select ...:



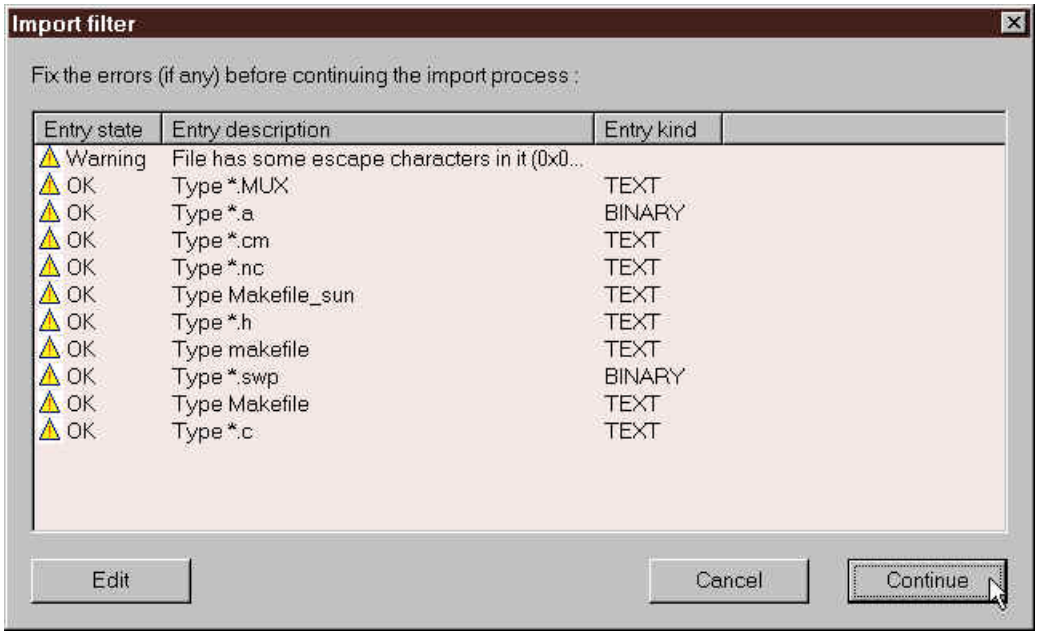
Localice la carpeta que quiere importar y haga doble-click en ella para ver el icono de carpeta abierta. Si selecciona la carpeta con un solo click de ratón, se seleccionará la carpeta padre y todas sus subcarpetas, en lugar de la carpeta deseada. Esto es un bug de todos los comandos de WinCvs que usan esta forma de seleccionar carpetas. En el ejemplo de arriba se ha seleccionado la carpeta New-Core, que muestra su icono de carpeta abierta.

Después de hacer doble-click en la carpeta a importar pulse OK. En ese momento WinCvs examinará la jerarquía para determinar cuantos ficheros se están importando y determinar el tipo de cada uno de ellos (text o binary). Este proceso se llama filtrado (filtering) y puede llevar algún tiempo si la jerarquía es grande. Dirante este tiempo la ventana status de WinCvs visualizará una línea por cada carpeta examinada, como se muestra a continuación:

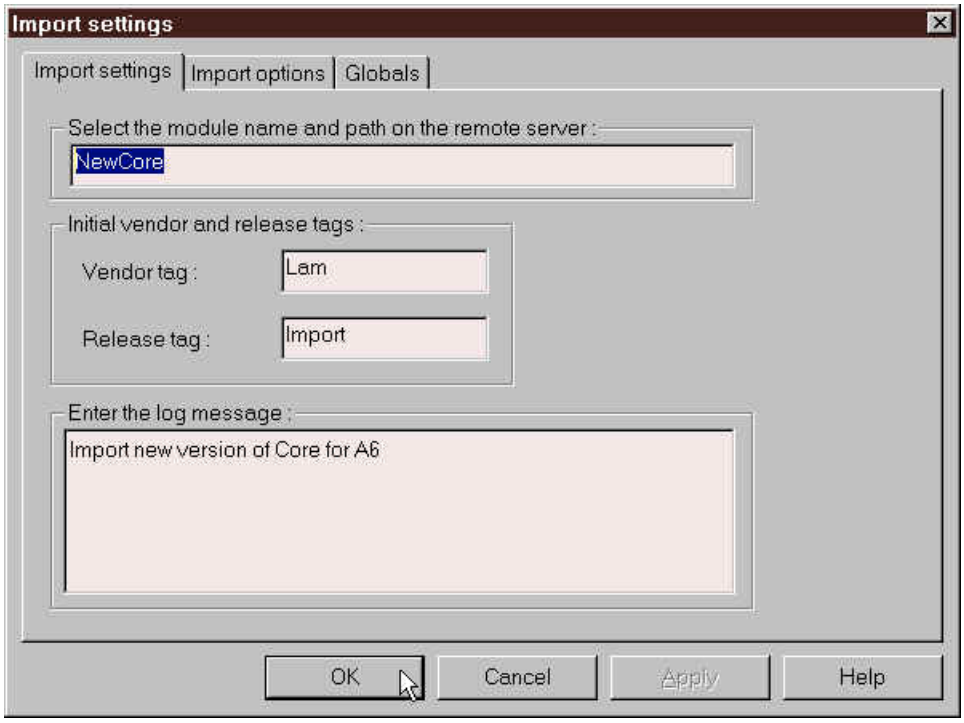
```
Filtering 'E:\New-Core'...
Filtering 'E:\New-Core\boot_strap'...
Filtering 'E:\New-Core\database'...
Filtering 'E:\New-Core\drivers'...
.
.
.
Filtering 'E:\New-Core\h'...
Filtering 'E:\New-Core\utilities'...
Filtering 'E:\New-Core\vme_trap'...
```

Cuando se haya terminado el proceso de filtrado se visualizará el panel Import filter con los resultados del mismo. Este panel lista las extensiones de ficheros encontradas en la jerarquía a importar y el tipo que cvs le asignará al fichero (TEXT o BINARY). En el siguiente ejemplo hay 1 warning y no hay ningun error. El warning indica que en un fichero con una extensión que normalmente indica que el fichero es de texto (TEXT) se han encontrado caracteres binarios. Los errores pueden ser ignorados (normalmente WinCvs hará lo correcto) o se puede pulsar el botón Edit para cambiar lo detectado por WinCvs.

Despues de que haya revisado el panel Import filter, pulse Continue para comenzar el proceso de importación:



Después de pulsar el botón Continue se visualizará el panel Import settings:



En este ejemplo estamos creando una nueva carpeta A6 bajo el modulo EtchPM ya existente. Nótese que el módulo importado Data-A6 no aparecerá en el repositorio. Todos los ficheros y subcarpetas de Data-A6 serán importados al repositorio en la carpeta A6. Este ejemplo muestra que una carpeta puede ser importada con un nombre diferente. Hay que rellenar cuatro campos antes de iniciar el import:

- Introduzca el nombre del path y el módulo como se muestra arriba. Nótese que cvs REQUIERE que se usen caracteres "/" para separar los nombres de directorios. Si utiliza el caraceter backslash (\) crearía un nuevo módulo llamado EtchPM\A6 en la raiz del repositorio en lugar de un nuevo directorio A6 bajo el directorio EtchPM existente.

- Entre cualquier cosa en los campos vendor tag y release tag. Estas etiquetas no se usan normalmente y pueden ser borradas después. Utilice etiquetas como VTAG y RTAG y borrelas después de que el import se haya completado. Desafortunadamente CVS las necesita para el comando import. Pueden ser borradas después de que el import se haya realizado seleccionando la carpeta del nivel superior y usando Delete tag en el menú Selections->Tag selection.

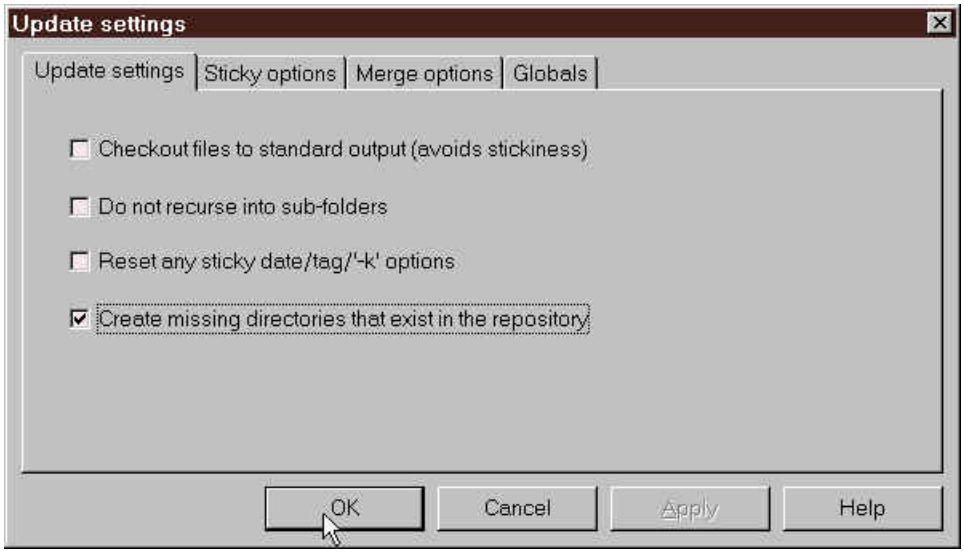
- Introduzca un mensaje de log adecuado (este mensaje será parte de la información de log asociada a los ficheros para la versión inicial 1.1) que aparecerá en el fichero si la palabra clave $Log se especifica en el mismo (solo ficheros de texto).

Después de que se hayan introducido todos los campos, pulse el botón OK para iniciar el import.

Durante la ejecución del import, el comando de cvs que se ejecuta y la salida de este se visualizarán en la ventana de status de WinCvs, como se muestra en este ejemplo:

```
cvs -q import -I ! -I CVS -W "*.a -k 'b'" -W "*.swp -k 'b'" -m "Import new
version of Core for A6" NewCore AVendor Import (in directory E:\New-Core)
N NewCore/Auxiliary_Clock/auxClock.c
N NewCore/Auxiliary_Clock/Makefile
N NewCore/boot_strap/.readConfig.c.swp
N NewCore/boot_strap/makefile
.
.
.
N NewCore/watchdog/Makefile
N NewCore/watchdog/watchdog.c

No conflicts created by this import


*****CVS exited normally with code 0*****
```

Ahora que los ficheros han sido importados al repositorio, se puede hacer checkout de los mismos al area de trabajo ejecutando el comando update sobre el directorio padre. En este ejemplo, seleccione el directorio EtchPM en la ventana del browser de WinCvs y seleccione update con el botón derecho del ratón: Se visualizará el panel Update settings :



Active la opción Create missing directories that exist in the repository y pulse el botón OK para obtener una working copy (copia de trabajo) de los ficheros importados en el area de trabajo.

En este ejemplo de comando import se ha creado un nuevo módulo. Sin embargo, el nuevo módulo no aparecerá en la lista de módulos disponibles hasta que su nombre sea añadido a un fichero administrativo llamado modules. Vea la Sección 4.1 para saber como especificar nuevos módulos en el fichero modules.

3.11 Coordinación entre varios programadores

El modelo que por defecto utiliza cvs para que puedan trabajar varios programadores se llama *unreserved checkouts*. En este modo de funcionamiento varios programadores pueden editar a la vez su copia de trabajo de un mismo fichero. Al primero de estos programadores que haga commit se le avisará de que haga un update si otro programador ya ha hecho commit de sus cambios a un fichero mientras este fichero estaba siendo modificado por aquel. El actualizar automaticamente el fichero con los cambios realizados recientemente por otros programadores lo realiza cvs automáticamente sin que lo tenga que realizar el programador a mano, y solo es necesario hacer la actualización manualmente cuando cvs encuentra modificaciones que estan en conflicto y que no puede resolver el solo.

Cvs también soporta parcialmente el modelo *reserved checkout*. Parcialmente porque cvs no obliga a que un fichero sea bloqueado antes de poder editarlo, pero no permitirá hacer commit de un fichero que está bloqueado por otro programador. Los programadores pueden bloquear y desbloquear ficheros a voluntad.

3.11.1 Como funciona el update y el modelo Unreserved Checkout

Cuando los programadores usan el modelo *unreserved checkout* en cvs, cualquier número de programadores puede estar trabajando en el mismo fichero a la vez. Como se explicó en la Sección 6.3, hay que ejecutar el comando edit para marcar el fichero como modificable antes de que se puedan realizar

cambios en el mismo. En cualquier momento se puede visualizar la lista de programadores que en ese momento están modificando un fichero seleccionando el fichero con el botón derecho del ratón y ejecutando la opción Monitors selection->Editors of selection.

Cuando un programador intenta hacer commit de un fichero pueden suceder una de estas dos cosas:

- Si no hay ninguna versión mas nueva del fichero en el repositorio, se creará una nueva versión de este fichero en el repositorio, como se explica en la Sección 3.9. Esta es la situación normal cuando ningún otro programador está trabajando en el mismo fichero, o bien si lo esta pero se es el primero en hacer commit.

- Si en el repositorio ya existe una nueva versión del fichero, cvs le mostrará un mensaje de advertencia y abortará la operación de commit.

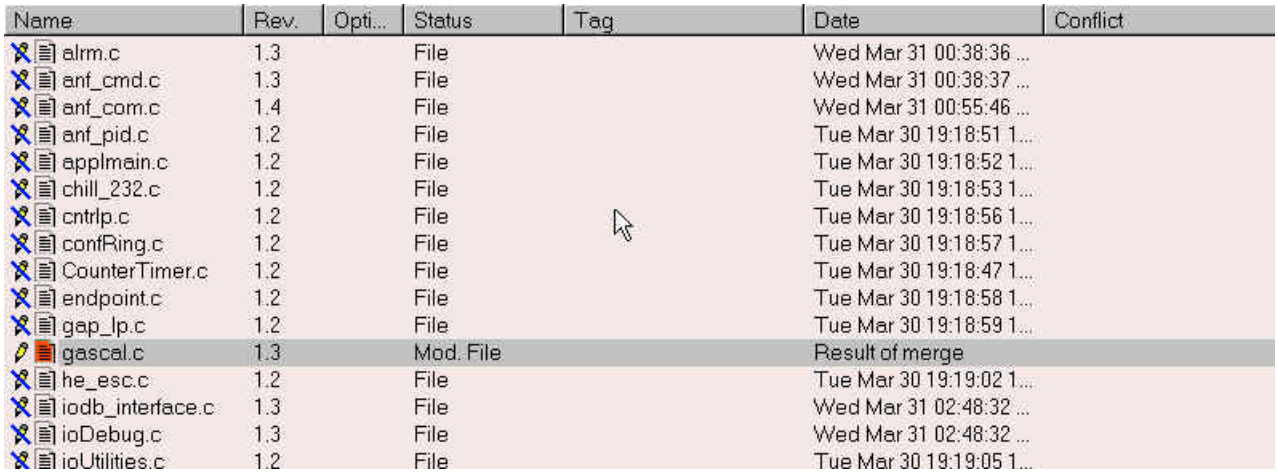
En el caso de que cvs aborte la operación de commit porque encuentre una nueva versión del fichero en el repositorio, WinCvs se lo advertirá en la ventana de status con un mensaje similar al siguiente:

```
cvs -q commit -m "This is the change from user 1" gascal.c (in directory
E:\WorkAreaOne\EtchPM\src)
cvs server: Up-to-date check failed for `gascal.c'
cvs [server aborted]: correct above errors first!

*****CVS exited normally with code 1*****
```

En este caso, se debe usar primero el comando update para actualizar el fichero con los cambios realizados por los otros programadores. Si el comando update se realiza con éxito, se visualizarán mensajes como los siguientes:

```
cvs -q update gascal.c (in directory E:\WorkAreaOne\EtchPM\src)
RCS file: /Store/200mm/EtchPM/src/gascal.c,v
retrieving revision 1.2
retrieving revision 1.3
Merging differences between 1.2 and 1.3 into gascal.c
M gascal.c


*****CVS exited normally with code 0*****
```

En este caso, el status del fichero seguirá siendo Mod. File pero la fecha de modificación cambiará al texto Result of Merge como se muestra a continuación:



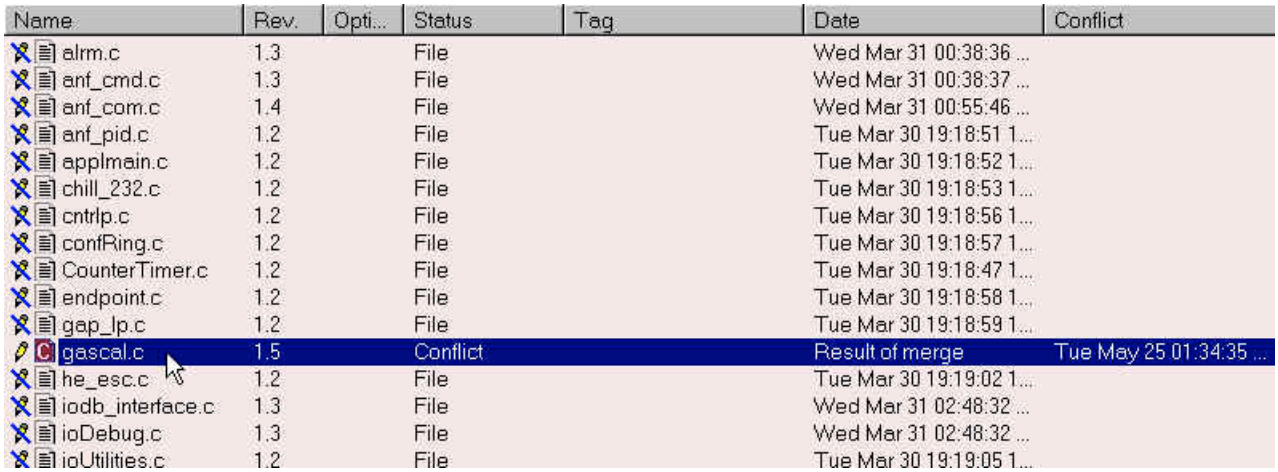
Name	Rev.	Opti...	Status	Tag	Date	Conflict
alrm.c	1.3		File		Wed Mar 31 00:38:36 ...	
anf_cmd.c	1.3		File		Wed Mar 31 00:38:37 ...	
anf_com.c	1.4		File		Wed Mar 31 00:55:46 ...	
anf_pid.c	1.2		File		Tue Mar 30 19:18:51 1...	
applmain.c	1.2		File		Tue Mar 30 19:18:52 1...	
chill_232.c	1.2		File		Tue Mar 30 19:18:53 1...	
cntrlp.c	1.2		File		Tue Mar 30 19:18:56 1...	
confRing.c	1.2		File		Tue Mar 30 19:18:57 1...	
CounterTimer.c	1.2		File		Tue Mar 30 19:18:47 1...	
endpoint.c	1.2		File		Tue Mar 30 19:18:58 1...	
gap_lp.c	1.2		File		Tue Mar 30 19:18:59 1...	
gascal.c	1.3		Mod. File		Result of merge	
he_esc.c	1.2		File		Tue Mar 30 19:19:02 1...	
iodb_interface.c	1.3		File		Wed Mar 31 02:48:32 ...	
ioDebug.c	1.3		File		Wed Mar 31 02:48:32 ...	
ioUtilities.c	1.2		File		Tue Mar 30 19:19:05 1...	

Después de esto, ya se puede ejecutar otra vez comando commit para que el fichero actualizado con los cambios realizados por otros programadores sea copiado en el repositorio.

En el caso de que el comando update falle porque cvs no sea capaz de actualizar el fichero con los cambios que estan en la version del fichero en el repositorio, pero no estan en la copia de trabajo del fichero, se visualizarán mensajes similares a los siguientes:

```
cvs -q update gascal.c (in directory E:\WorkAreaOne\EtchPM\src)
RCS file: /Store/200mm/EtchPM/src/gascal.c,v
retrieving revision 1.4
retrieving revision 1.5
Merging differences between 1.4 and 1.5 into gascal.c
rcsmerge: warning: conflicts during merge
cvs server: conflicts found in gascal.c
C gascal.c

*****CVS exited normally with code 0*****
```

Observe la "C" indicando que se encontraron conflictos. En este caso el status del fichero cambiará a Conflict y la fecha cambiará a Result of Merge como se muestra a continuación:



Name	Rev.	Opti...	Status	Tag	Date	Conflict
alrm.c	1.3		File		Wed Mar 31 00:38:36 ...	
anf_cmd.c	1.3		File		Wed Mar 31 00:38:37 ...	
anf_com.c	1.4		File		Wed Mar 31 00:55:46 ...	
anf_pid.c	1.2		File		Tue Mar 30 19:18:51 1...	
applmain.c	1.2		File		Tue Mar 30 19:18:52 1...	
chill_232.c	1.2		File		Tue Mar 30 19:18:53 1...	
cntrlp.c	1.2		File		Tue Mar 30 19:18:56 1...	
confRing.c	1.2		File		Tue Mar 30 19:18:57 1...	
CounterTimer.c	1.2		File		Tue Mar 30 19:18:47 1...	
endpoint.c	1.2		File		Tue Mar 30 19:18:58 1...	
gap_lp.c	1.2		File		Tue Mar 30 19:18:59 1...	
gascal.c	1.5		Conflict		Result of merge	Tue May 25 01:34:35 ...
he_esc.c	1.2		File		Tue Mar 30 19:19:02 1...	
iodb_interface.c	1.3		File		Wed Mar 31 02:48:32 ...	
ioDebug.c	1.3		File		Wed Mar 31 02:48:32 ...	
ioUtilities.c	1.2		File		Tue Mar 30 19:19:05 1...	

En este caso, el fichero resultante debe ser re-editado para resolver los conflictos manualmente. Los conflictos se pueden localizar buscando en el fichero las cadenas de caracteres **<<<<<<<**, **=======**, y **>>>>>>>**, como se muestra en el siguiente ejemplo:

```
/*************************************************************
*
* new_function_from_user_1
*
*/
<<<<<<< gascal.c
void new_function_1( d, e, f )
=======
void new_function_1( a, b, c )
>>>>>>> 1.5
{
}
```

El programador debe suprimir manualmente las lineas con <<<<<<<, =======, y >>>>>>> y resolver los conflictos para las indicadas. Despues de que los conflictos hayan sido resueltos se debe volver a ejecutar el comando commit.

Si el programador intenta hacer commir sin resolver ninguno de los conflictos, cvs visualizará el siguiente mensaje de error y abortará la operación de commit:

```
cvs -q commit -m "This is the change from user 1\n\n" gascal.c (in
directory E:\WorkAreaOne\EtchPM\src)
cvs server: file `gascal.c' had a conflict and has not been modified
cvs [server aborted]: correct above errors first!

*****CVS exited normally with code 1*****
```

Si el programador resuelve solo algunos de los conflictos pero deja el fichero con otros conflictos sin resolver, cvs visualizará el siguiente mensaje de error cuando se haga el commit:

```
cvs -q commit -m "This is the change from user 1\n\n" gascal.c (in
directory E:\WorkAreaOne\EtchPM\src)
cvs server: warning: file `gascal.c' seems to still contain conflict
indicators
Checking in gascal.c;
/Store/200mm/EtchPM/src/gascal.c,v  <--  gascal.c
new revision: 1.6; previous revision: 1.5
done

*****CVS exited normally with code 0*****
```

Nótese que a pesar de esto cvs hará commit del fichero aunque detecte alguno de los indicadores de conflicto (<<<<<<<, =======, o >>>>>>>). Probablemente esto no sea una característica deseable en cvs, pero es así como funciona. Si accidentalmente se hace commit de un fichero que todavía contiene indicadores de conflictos, el fichero debe ser editado, vuelto a modificar y se debe hacer commit otra vez para suprimir los indicadores de conflicto y resolver los conflictos que queden aún.

3.11.2 Como funciona el sistema de bloqueos y el modelo Reserved Checkout

Como se dijo en la Sección 3.11, cvs soporta el modelo *reserved checkout* . El comando lock es la base de este modelo. Bloqueando un fichero en cvs se evita que otro programador tambien pueda bloquearlo ni hacer commit de una nueva versión del fichero. Esta forma de trabajar es segura en tanto que evita que otro programador pueda hacer commit añadiendo una nueva versión del fichero al repositorio, pero sigue permitiendo que otro programador edite el fichero (su copia en el area de trabajo). Esto puede producir frustración, si un programador olvida bloquear el fichero pero si lo edita. Cuando el programador intente hacer commit del fichero, la operación fallará como se muestra en el siguiente ejemplo:

```
cvs -q commit -m "This is the change from user 2\n\n" gascal.c (in
directory E:\WorkAreaTwo\EtchPM\src)
cvs [server aborted]: Revision 1.7 is already locked by apples

*****CVS exited normally with code 1*****
```

Si todos los programadores cooperan y se ponen de acuerdo en bloquear un fichero antes de editarlo, esta situación no se dará. Como alternativa a esto se puede usar el modelo *unreserved checkout* .

Como sucede con otros comandos, WinCvs tiene varias maneras de invocar el comando lock :

1) Seleccione el fichero(s) usando el botón izquierdo del ratón y el menú Selections:

   Selections->Monitors selection->Lock selection

2) Seleccione un fichero usando el botón derecho del ratón y la opción Monitors selection->Lock selection

3) Seleccione el fichero(s) usando el botón izquierdo del ratón y el icono correspondiente de la barra de herramientas:

Utilize el comando unlock de una de estas formas para desbloquear el fichero:

1) Seleccione el fichero(s) usando el botón izquierdo del ratón y use el menú Selections:

   Selections->Monitors selection->Unlock selection

2) Seleccione un fichero usando el botón derecho del ratón y escoja la opción Monitors selection->Unlock selection

3) Seleccione el fichero(s) usando el botón izquierdo del ratón y utilice el icono correspondiente de la barra de herramientas:

Si la operación de lock tiene éxito, se visualizarán los siguientes mensajes en la vantana status:

```
cvs -q admin -l gap_lp.c (in directory E:\WorkAreaTwo\EtchPM\src)
RCS file: /Store/200mm/EtchPM/src/gap_lp.c,v
1.2 locked
done

*****CVS exited normally with code 0*****
```

Si la operación de lock falla, se visualizará información indicando quien tiene bloqueado actualmente el fichero:

```
cvs -q admin -l gascal.c (in directory E:\WorkAreaTwo\EtchPM\src)
RCS file: /Store/200mm/EtchPM/src/gascal.c,v
cvs [server aborted]: Revision 1.7 is already locked by apples

*****CVS exited normally with code 1*****
```

Desde todos los menús y tambien desde la barra de herramientas se puede ejecutar el comando Log para conocer el estado de un fichero en cuanto al bloqueo. La salida del comando se visualiza en la ventana de status como se muestra en este ejemplo donde el usuario don tiene bloqueada la versión 1.2 del fichero gap_lp.c:

```
Rcs file : '/Store/200mm/EtchPM/src/gap_lp.c,v'
Working file : 'gap_lp.c'
Head revision : 1.2
Branch revision :
Locks : strict
      1.2 : 'don'
```

Hay algunos casos en los que se puede tener bloqueadas multiples versiones de un fichero. Si se intenta usar el comando unlock en este caso, se obtendrá el siguiente mensaje de error:

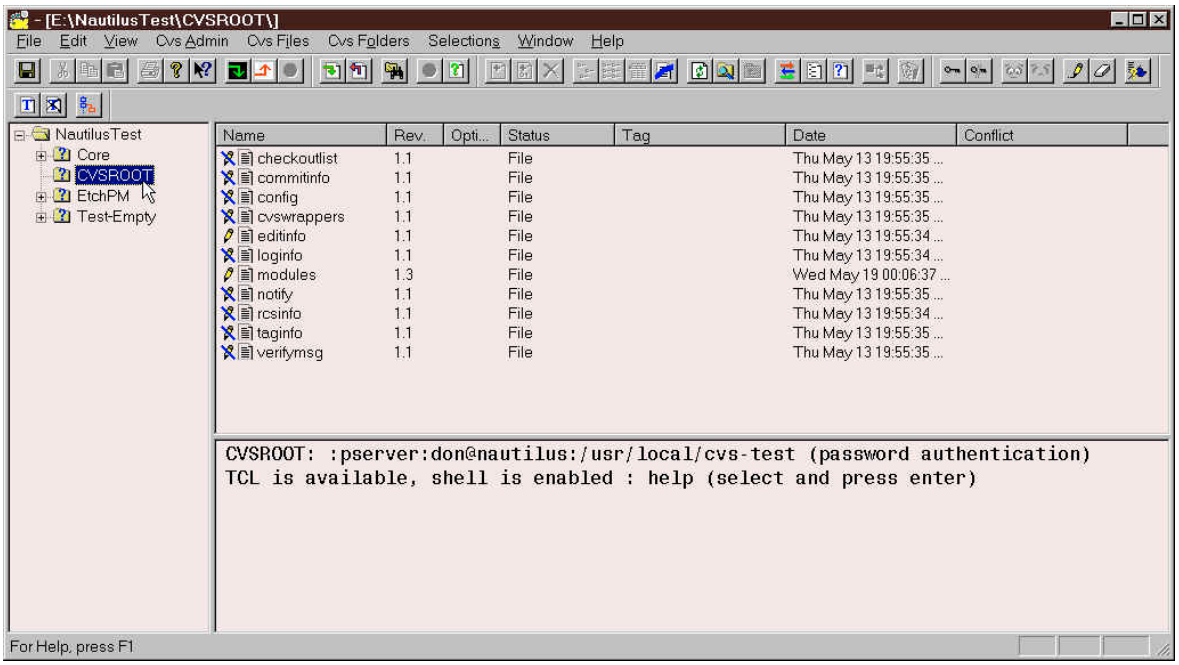
```
cvs -q admin -u gap_lp.c (in directory E:\WorkAreaTwo\EtchPM\src)
RCS file: /Store/200mm/EtchPM/src/gap_lp.c,v
cvs server: /Store/200mm/EtchPM/src/gap_lp.c,v: multiple revisions locked
by don; please specify one
cvs server: cannot modify RCS file for `gap_lp.c'

*****CVS exited normally with code 1*****
```

En este caso, debe desbloquear una determinada versión del fichero usando el comando cvs admin, tecleándolo en la ventana de status de WinCvs. En este ejemplo, la versión 1.1 de gap_lp.c ha sido desbloqueada (el programador ha tecleado aquí el comando cvs –q admin –l1.1 gap_lp.c):

```
cvs -q admin -l1.1 gap_lp.c

*****CVS exited normally with code 0*****

RCS file: /Store/200mm/EtchPM/src/gap_lp.c,v
done
```

Sección 4 – Comandos Administrativos

Esta sección trata sobre comandos que normalmente deberían ser usados solo por un administrador de cvs. Para editar cualquier fichero administrativo, hay que hacer checkout del módulo CVSROOT en un área de trabajo local. Vea las seccion 3.4 para información sobre el comando checkout. A continuación se muestra un area de trabajo con una versión de CVSROOT obtenida mediante checkout:



Observe que hay 11 ficheros administrativos en el módulo CVSROOT. Estos ficheros permiten manejar muchas características avanzadas de cvs. Vea el apéndice C (pag 125) del manual de referencia de cvs para una descripción completa de estos ficheros y sus caracteristicas, incluido el fichero opcional passwd.

En esta sección describiremos solamente el uso del fichero modules .

4.1 Editar el fichero administrativo modules

El fichero modules de cvs lista las jerarquías de carpetas para las que se puede hacer checkout desde el repositorio usando el comando Cvs Admin->Checkout module… del menu principal de WinCvs. Usando el fichero modules a cualquier parte de la jerarquía de un módulo o a varios módulos existentes se les puede asignar un solo nombre.

Si se importa un módulo a cvs y no se pone en el fichero modules, se podrá hacer checkout del mismo, pero no será listado en la salida del comando Cvs Admin->Macros admin->List the modules on the server del menú principal de WinCvs.

Para poder editar el fichero modules, se debe hacer primero checkout del módulo CVSROOT, como se mencionó en la introducción a la Sección 4. Además, el fichero debe ser marcado como modificable, como se dijo en la Sección 3.6.

A continuación se muestra un ejemplo de fichero modules:

```
# Three different line formats are valid:
#   key -a    aliases...
#   key [options] directory
#   key [options] directory files...
#
# Where "options" are composed of:
#   -i prog     Run "prog" on "cvs commit" from top-level of module.
#   -o prog     Run "prog" on "cvs checkout" of module.
#   -e prog     Run "prog" on "cvs export" of module.
#   -t prog     Run "prog" on "cvs rtag" of module.
#   -u prog     Run "prog" on "cvs update" of module.
#   -d dir      Place module in directory "dir" instead of module name.
#   -l      Top-level directory only -- do not recurse.
#
# NOTE:  If you change any of the "Run" options above, you'll have to
# release and re-checkout any working directories of these modules.
#
# And "directory" is a path to a directory relative to $CVSROOT.
#
# The "-a" option specifies an alias.  An alias is interpreted as if
# everything on the right of the "-a" had been typed on the command line.
#
# You can encode a module within a module by using the special '&'
# character to interpose another module into the current module.  This
# can be useful for creating a module that consists of many directories
# spread out over the entire source repository.
#
EtchPM EtchPM
Core   Core
EtchPM-Source EtchPM/src
```

La Sección C.1 (pag 125) del manual de referencia de cvs describe el fichero modules en detalle. Como todos los ficheros después de modificados, hay que hacer commit del fichero modules al repositorio (Section 3.9) antes de que los cambios tengan efecto.

4.2 Acciones a realizar cuando el repositorio queda bloqueado

Internamente cvs realiza un bloqueo sobre el repositorio para evitar accesos simultaneos de varios usuarios . El bloqueo es realmente un directorio de del repositorio llamado `#cvs.lock`. Si una operación de cvs aborta, es posible que este bloqueo sobre el repositorio permanezca, impidiendo que despues de esto se puedan realizar otras operaciones. En este caso el mensaje “waiting for user’s lock” se visualizará en la ventana de status, como se muestra en el siguiente ejemplo donde el usuario R2D2 es el que tiene el bloqueo (el que tiene bloqueado el repositorio):

```
cvs admin -bLockTest2 -u applmain.c

*****CVS exited normally with code 0*****

cvs server: [13:46:22] waiting for R2D2's lock in /Store/200mm/EtchPM/src
```

Normalmente este mensaje significa que el usuario R2D2 esta en el proceso de actualizar el repositorio. Esto normalmente no debería ocupar mucho tiempo, al cabo del cual el bloqueo se quitaría como se muestra a continuación:

```
cvs server: [13:46:52] waiting for R2D2's lock in /Store/200mm/EtchPM/src
cvs server: [13:47:22] obtained lock in /Store/200mm/EtchPM/src
RCS file: /Store/200mm/EtchPM/src/applmain.c,v
1.2.6.2 unlocked
done
```

En los casos en que cvs aborte por alguna razón (p.e. pérdida de la comunicación de red o rearranque de la máquina), puede ser necesario quitar el bloqueo manualmente. El directorio `#cvs.lock` localizado en el repositorio en cuestión y borrado usando el comando rmdir de UNIX (supuesto que el repositorio está en una máquina UNIX).

4.3 Gestión de versiones

El mecanismo básico para mantener varias versiones de un producto usando cvs es la etiqueta. Las etiquetas se describen en la Sección 4.4 (pag 32) del manual Version Management with CVS. Cuando un producto ha sido testeado y está preparada una versión de producción del mismo con un determinado conjunto de versiones de ficheros, entonces los ficheros pueden ser etiquetados con un nombre simbólico. Los ficheros pueden ser modificados después de la versión de producción, pero la versión etiquetada se puede recuperar siempre.

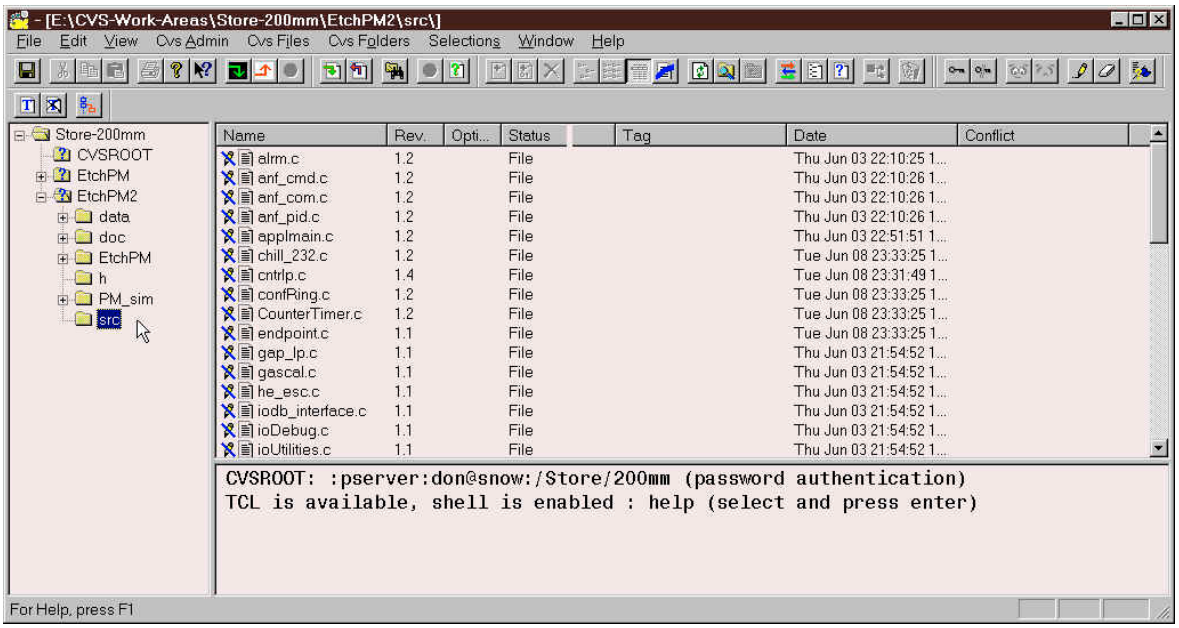
El problema que siempre se plantea con el manejo de versiones es que hacer cuando se detecta un bug despues de que se ha sacado una versión de producción del producto. Probablemente los ficheros del repositorio han sido modificados despues de lanzarse la versión de producción y no son lo bastante estables como para generar una nueva versión de producción que corrija el bug.

La solución a este problema es crear una bifurcación (branch) a partir de la versión que hay en el repositorio etiquetada cuando se generó la versión de producción (que no es la última). Después de esto, los ficheros de la bifurcación pueden ser modificados sin interferir con el hecho de que continúe el desarrollo para la bifurcación principal (técnicamente la bifurcación principal se llama **trunk**). Después de que se haya comprobado la solución de los errores, la etiqueta de la versión de producción original puede ser trasladada a las nuevas versiones de los ficheros, o se puede crear una nueva etiqueta de versión de producción.

Las dos secciones siguientes muestran ejemplos de cómo mover una etiqueta a una nueva versión de producción y como crear una nueva etiqueta para una versión de producción.

4.3.1 Etiquetar una versión de producción

Cuando un conjunto de ficheros han sido testeados y están listos para generar una versión de producción, se deben etiquetar para marcar su estado actual. En el siguiente ejemplo se visualiza un area de trabajo para el producto EtchPM2. Se muestra parcialmente el listado de ficheros de la carpeta src para mostrar el conjunto de números de versiones (1.1, 1.2, and 1.4).



Suponiendo que el area de trabajo EtchPM2 representa un conjunto de ficheros testeados, estos pueden ser etiquetados para generar una versión de producción con el comando tag.

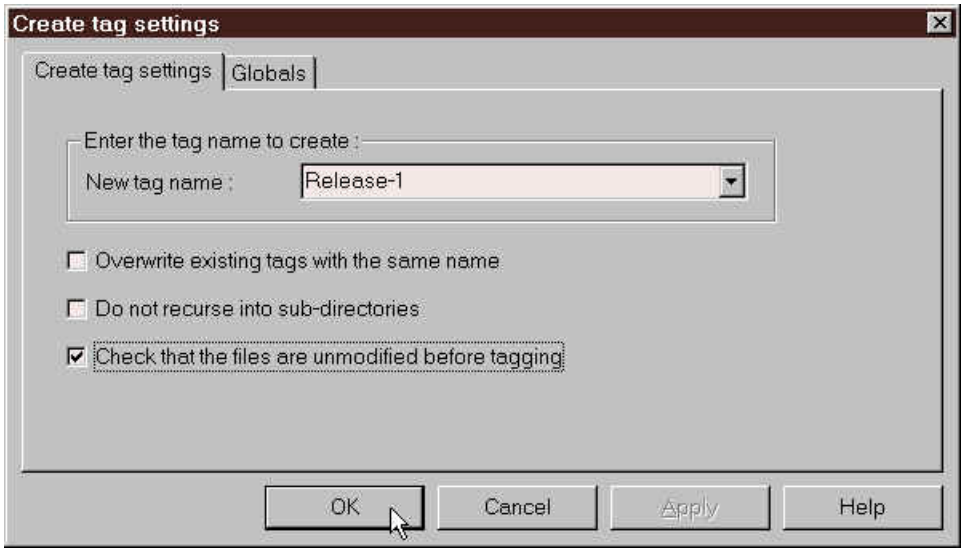
El comando tag se puede invocar de cualquiera de estas formas:

1) Seleccione la carpeta de mayor nivel en la jerarquía (EtchPM2 en este caso) con el botón izquierdo del ratón y ejecute la opción de menú Selections->Tag selection->Create a tag…

2) Seleccione la carpeta con el botón derecho del ratón y seleccione la opción Tag selection->Create a tag… del menú emergente.

3) Seleccione la carpeta usando el botón izquierdo del ratón y después el icono de la barra de herramientas:

Despues de invocar el comando tag se visualizará el panel Create tag settings, como se muestra a continuación:



El panel Create tag settings tiene un campo para introducir el nombre de la etiqueta y tres opciones. El nombre de la etiqueta no debe contener ninguno de los siguientes caracteres: `$ , . : ; @.`

La opción Overwrite existing tags with the same name especifica que cualesquiera etiquetas con el mismo nombre que se encuentren sean trasladas a la versión actual (la que hay en el area de trabajo actual).

La opción Do not recurse into sub-directories puede ser util cuando solo se desea etiquetar una carpeta.

Es una buena práctica marcar la opción Check that the files are unmodified before tagging, pues no tiene mucho sentido tener ficheros en el area de trabajo que hayan sido modificados pero para los cuales no se haya hecho commit.

Despues de introducir la etiqueta y haber seleccionado las opciones deseadas, pulse OK para comenzar el proceso de etiquetado. Durante la operación cvs visualizará los ficheros mientras van siendo etiquetados, como se muestra a continuación:

```
cvs -q tag -c Release-1 (in directory E:\CVS-Work-Areas\Store-
200mm\EtchPM2\)
T Makefile
.
.
*****CVS exited normally with code 0*****
```

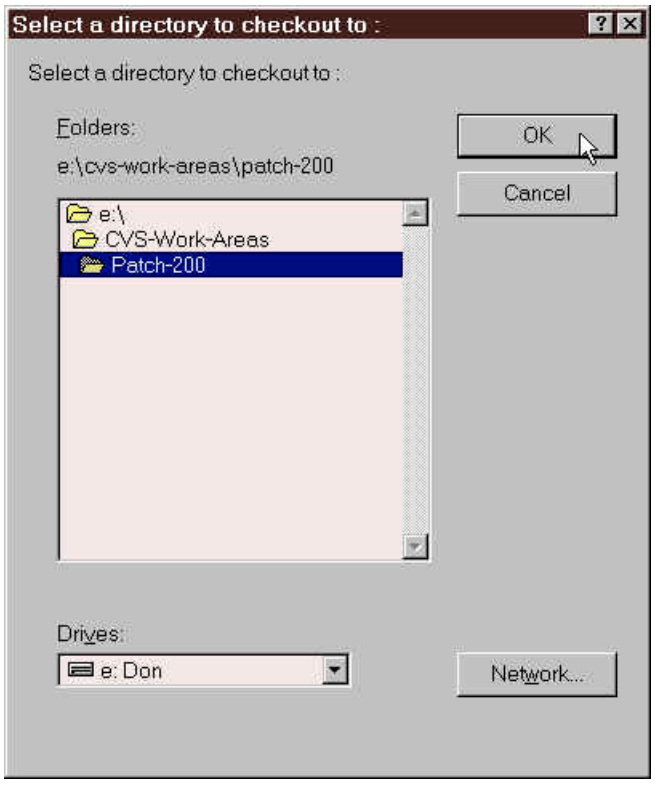
4.3.2 Corregir errores después de generar una versión de producción

Cuando se detecta un error en una versión de producción (probablemente instalada y funcionando en cliente(s)), el procedimiento usual es generar una versión incremental con los mínimos cambios necesarios para corregir el error. El primer paso para esto es crear el area de trabajo en la que se va modificar la versión de producción para generar la versión incremental. Dependiendo del producto, puede necesitarse un solo fichero o todo el arbol del repositorio. El area de trabajo se creará de forma que contenga la versión de todos los ficheros que se etiquetaron con un determinado identificador o etiqueta en el momento de generar la versión de producción, es decir, en el estado en que estaban en ese momento (desde entonces el desarrollo probablemente ha continuado, por ejemplo incorporando nuevas funcionalidades al producto).

4.3.2.1 Crear el area de trabajo

Si ya existe un area de trabajo para el producto o módulo, esta puede ser actualizada a la versión que se quiere modificar con el comando update (Sección 3.5). Sin embargo, esta forma de proceder puede producir resultados no deseados: piensese por ejemplo que sucede si en el area de trabajo ya hay ficheros que han sido modificados pero para los cuales no se ha hecho commit o si la estructura de carpetas ha cambiado desde que se etiquetó la versión que ahora hay que modificar.

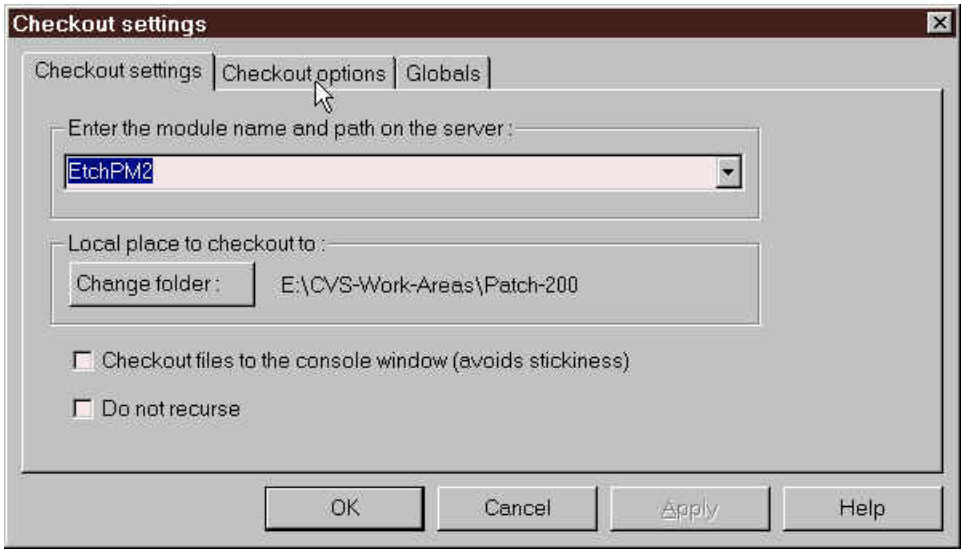
El mejor método para crear un area de trabajo a partir de una versión etiquetada es usar el comando checkout (Sección 3.4) y hacer un checkout de la versión etiquetada que se quiere modificar a una nueva area de trabajo. Cuando se invoque el comando checkout se visualizará el siguiente panel:



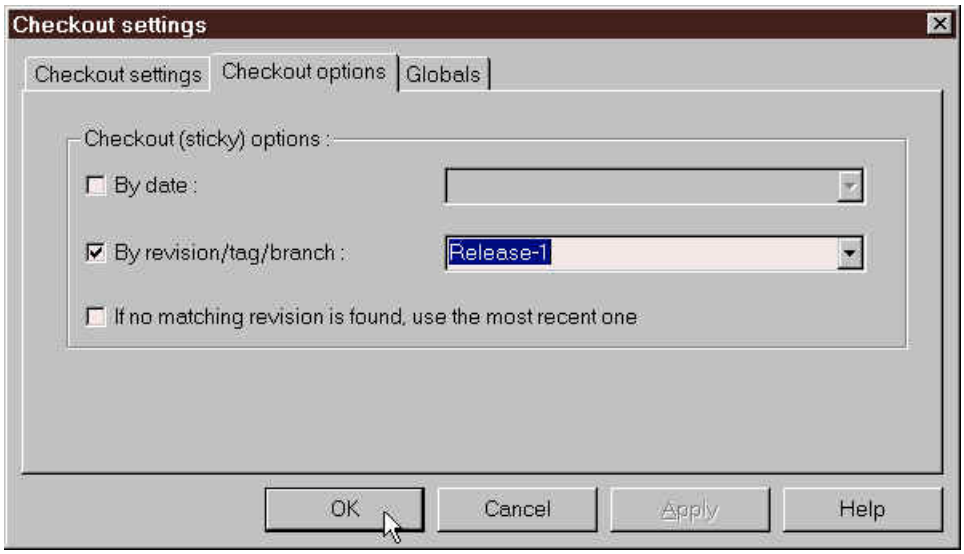
Seleccione la carpeta donde se creará el area de trabajo. Recuerde que debe hacer doble-click en la carpeta y ver el icono de carpeta abierta como se muestra arriba.

Una vez seleccionada la carpeta deseada, haga click en OK para continuar con el proceso de checkout. Se mostrará el panel Checkout settings .

El panel Checkout settings se muestra a continuación. Introduzca el nombre del módulo y vaya al panel Checkout options haciendo click en la solapa como se muestra a continuación:



El panel Checkout options se muestra a continuación. Marque la opción By revision/tag/branch e introduzca el nombre de la etiqueta correspondiente a la versión de producción que desee obtener:

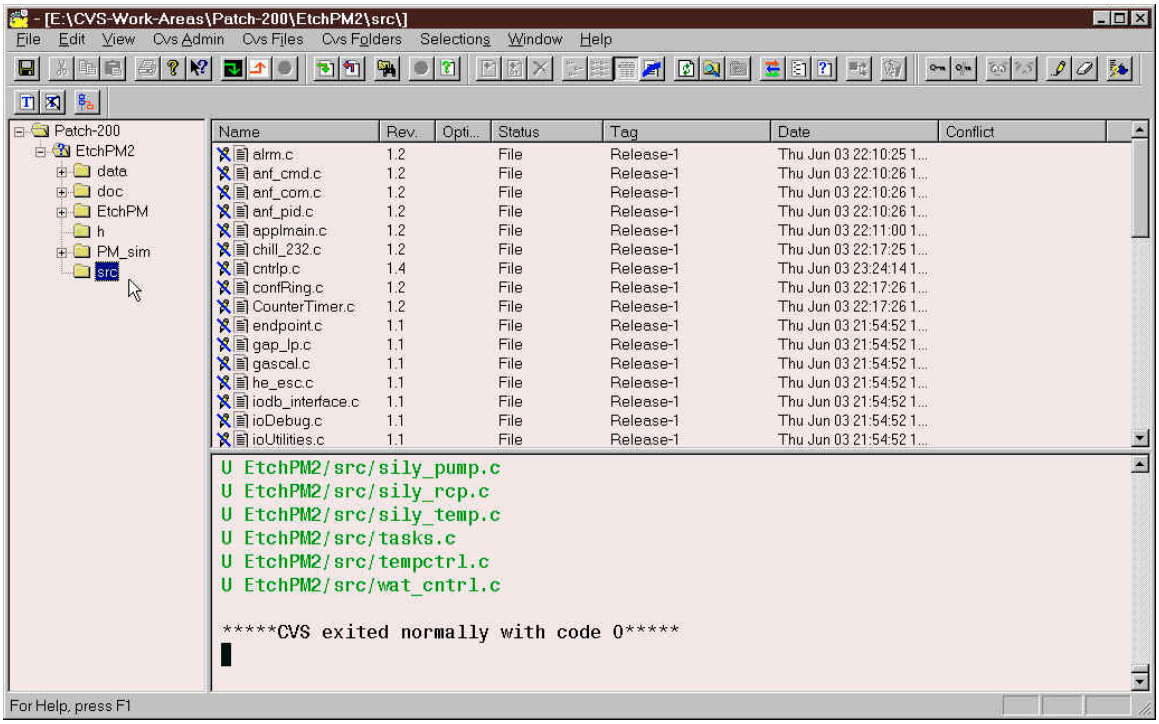


Despues de introducir el nombre de la etiqueta haga click en OK para inciar la operación de checkout.

Durante la operación de checkout, cvs mostrará cada fichero como a continuación:

```
cvs -q checkout -r Release-1 EtchPM2 (in directory E:\CVS-Work-
Areas\Patch-200)
U EtchPM2/Makefile
U EtchPM2/EtchPM/Makefile
.
.
.
U EtchPM2/h/gascal.h

*****CVS exited normally with code 0*****
```

Despues de que se haya realizado el checkout utilice el comando View->Change browser location del menú principal o la barra de herramientas para apuntar al area de trabajo que acaba de crear. A continuación se muestra un ejemplo de area de trabajo creada para la versión etiquetada como Release-1 :



Ahora que se ha creado un area de trabajo con los ficheros a las versiones en que estaban cuando se etiquetó la versión del producto para producción, debe crear una bifurcación para realizar las modificaciones necesarias para corregir el error(es). La sección siguiente explica la forma de crear una bifurcación a partir de la versión que se acaba de poner en el area de trabajo.

4.3.2.2 Crear una bifurcación

Los ficheros del area de trabajo que contiene la versión de producción etiquetada no se pueden etiquetar y hacer con ellos commit. Para hacer modificacionea a estos ficheros se necesita crear una bifurcación. Si ya existe una bifurcación utilice el comando update (Sección 3.5) para poner en el area de trabajo la última versión de la bifurcación. Si no existe una bifurcación para la versión etiquetada que quiere modificar, cree una como se muestra en esta sección.

Como sucede con otros comandos, WinCvs tiene varias formas de invocar el comando branch (fork):

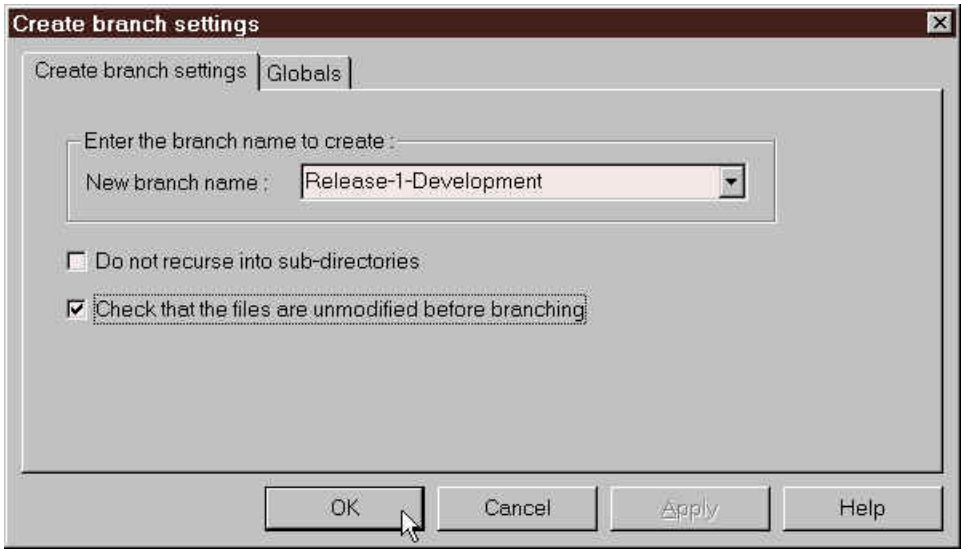
1) Seleccione el(los) fichero(s) o carpeta(s) usando el botón izquierdo del ratón y la siguiente opción de menú: Selections->Tag selection->Create a branch…

2) Seleccione el(los) fichero(s) o carpeta(s) usando el botón izquierdo del ratón y después el botón derecho para abrir el menú Selections. Aquí seleccione la entrada

   Tag selection->Create a branch…

3) Seleccione el(los) fichero(s) o carpeta(s) usando el botón izquierdo del ratón y después el icono de la barra de tareas:

Cuando se invoca el comando branch (fork), el panel Create branch settings se visualizará como se muestra a continuación:



Introduzca el nombre que desee dar a la bifurcación y pulse el botón OK. Es buena práctica dar a la bifurcación un nombre que identifique claramente el propósito del mismo. Como en este caso vamos a desarrollar sobre la versión etiquetada como Release-1 un nombre epropiado puede ser Release-1-Development.

El panel Create branch settings tiene dos opciones que pueden ser útiles en ciertos casos. Es una buena práctica marcar la opción Check that the files are unmodified before branching (comprobar que los ficheros

no han sido modificados antes de crear la bifurcación), ya que no tiene mucho sentido crear una bifurcación a partir de ficheros que han sido modificados pero para los que no se ha hecho commit. También es útil a veces marcar la opción Do not recurse into sub-directories, ya que a veces se crea una bifurcación para una sola carpeta y no se necesita que en la bifurcación estén las subcarpetas de esta (que es la opción por defecto).

La ventana de status mostrará como progresa la creación de la bifurcación, como se ve a continuación:

```
cvs -q tag -b -c Release-1-Development (in directory E:\CVS-Work-
Areas\Patch-200\EtchPM2\src\)
T CounterTimer.c
T Makefile
.
.
.
T PM_WaferFlow.c

*****CVS exited normally with code 0*****
```

La bifurcación se creará a partir de la versión de los ficheros que se crearon en el area de trabajo al hacer checkout. Notese que WinCvs usa el comando tag con la opción –b para crear la bifurcación. También se puede introducir manualmente este comando en la ventana de status de WinCvs.

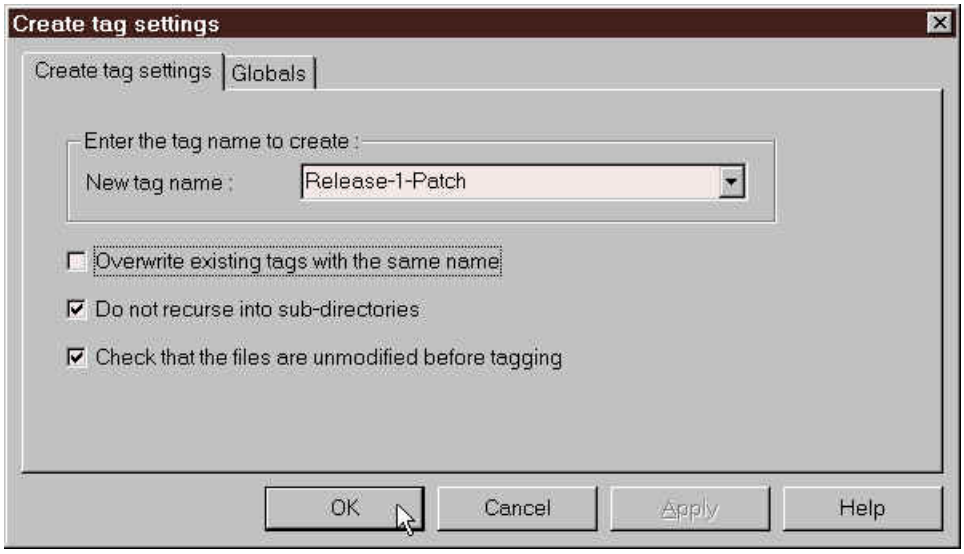
Notese también que no por el hecho de crear una bifurcación se actualiza el area de trabajo a la versión de la bifurcación (la operación de crear la bifurcación se hace contra el repositorio solo). Es necesario usar el comando update (Sección 3.5) para actualizar el area de trabajo a la bifurcación que se acaba de crear.

Una vez que la bifurcación ha sido creada y el area de trabajo actualizada, los ficheros pueden ser modificados y se puede hacer commit de estas modificaciones (contra la bifurcación en el repositorio, la linea principal de desarrollo continua por su camino) como se describió en las secciones 3.6 a 3.9. Una vez que se hayan realizado y probado todas las modificaciones, hay dos posibilidades para etiquetar una nueva versión de producción, que se describen en la siguiente sección.

4.3.2.3 Etiquetar la nueva versión de producción

En el ejemplo anterior se hizo checkout de una versión de producción etiquetada como Release-1, y se modificaron algunos ficheros creandose una bifurcación llamada Release-1-Development. En este punto, la persona responsable de versiones tiene dos opciones: los ficheros modificados pueden ser reetiquetados con la etiqueta originar Release-1 o generar una nueva versión de producción con una nueva etiqueta. En los casos en que Release-1 ha sido suministrada a muchos clientes, la versión de producción original etiquetada como Release-1 no debería se modificada, necesitándose una nueva versión de producción con una nueva etiqueta. Si no se ha llegado a suministrar ninguna copia de Release-1, puede ser una buena opción reetiquetar los ficheros modificados con la etiqueta original Release-1. A esto último se le llama trasladar las etiquetas, pues estas son realmente trasladadas de una versión a otra solo para los ficheros modificados.

En cualquier caso, el comando tag hay que invocarlo como se describió en la Sección 4.3.1. El siguiente panel muestra que opciones hay que marcar para crear una nueva versión etiquetada llamada Release-1-Patch:



Estas opciones se usarían para reetiquetar la nueva versión con la etiqueta original Release-1: