

The Frescale Cup

Sistemas de Control de Versiones

Manual de Práctica (TortoiseSVN)

Author(s):	Calvillo-Cortes, Carlos / Pérez-Navarro, Edgar
Version:	1
Last saved:	2010-04-16 at 18:23
Document file name:	Control de versiones (Manual de Practica) - 1.doc
Release Authority:	
Distribution:	team
Security Classification:	Public Domain
Number of pages:	28

List of Changes

Ver. (X.Y)	Date (YYYY-MM-DD)	Maturity (Draft/ Valid/ Withdrawn)	Author (Name/Departm.)	Description
1	19/ago/2008	Draft	Calvillo-Cortes, Carlos / Pérez-Navarro, Edgar	Creation
2	12/apr/2010	Draft	Calvillo-Cortes, Carlos / Pérez-Navarro, Edgar	Modify for The Freescale Cup

CONTENIDO	
1	INTRODUCCIÓN 3
2	OBJETIVO(S) 3
3	REFERENCIAS 3
4	ABREVIACIONES 3
5	PROCEDIMIENTO(S) 3
5.1	Instalación de TortoiseSVN 3
5.2	Creación de Repositorio (en máquina local) 4
5.3	Creación de Estructura de un Proyecto..... 5
5.4	Agregar archivos al Proyecto..... 8
5.5	Operaciones Commit / Update 12
5.6	Comparar Versiones / Lista de cambios (Change Log)..... 14
5.7	Creación de Rama (Branching / Tagging)..... 15
5.8	Gráfica de revisiones 21
5.9	Integrar modificaciones (Merge)..... 23
5.10	Simular Actividades de Equipo () 28
6	ENTREGABLES ERROR! BOOKMARK NOT DEFINED.

1 Introducción

Los sistemas de control de versiones son piezas fundamentales para el desarrollo de cualquier proyecto de software embebido (y proyectos en general). En el presente manual se practicará con el software TortoiseSVN que es un cliente del sistema Subversión (SVN) para el control de cambios. Subversion y TortoiseSVN son ambos de licencia libre y su uso esta ampliamente difundido.

2 Objetivo(s)

El objetivo de esta práctica es familiarizarse con el uso de una herramienta para el control de versiones (TortoiseSVN). Conocer las operaciones básicas de un sistema de control de versiones y mediante una forma práctica y del mundo real conocer las ventajas de usar un sistema de control de versiones para un proyecto.

Se busca que en su proyecto usen este sistema de control de versiones para que puedan trabajar varias personas a la vez y tener una sola línea de desarrollo.

3 Referencias

No.	Document Name	Date/ Revision	Link (if applicable)
/R1/	Sitio Oficial de TortoiseSVN	-	http://tortoisesvn.tigris.org/
/R2/	TortoiseSVN Manual	v 1.5.0	http://tortoisesvn.net/support

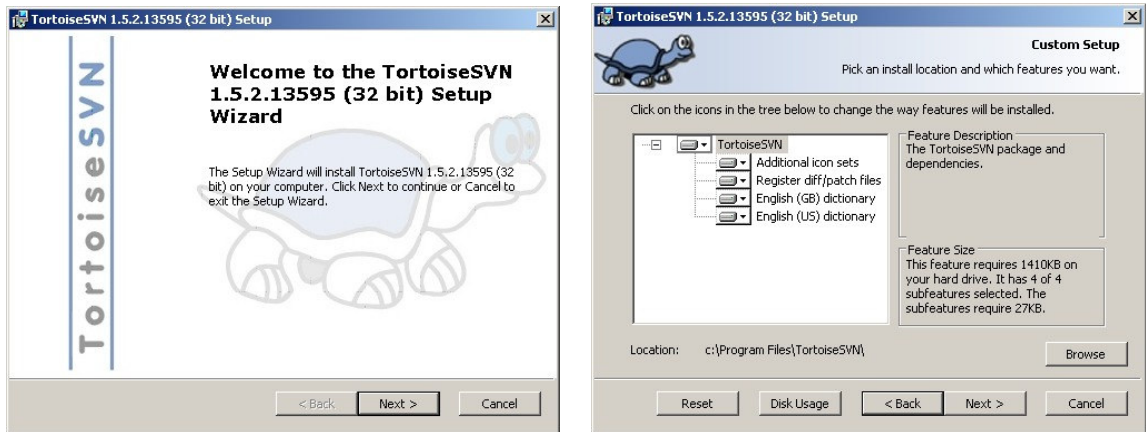
4 Abreviaciones

SVN Subversion

5 Procedimiento(s)

5.1 Instalación de TortoiseSVN

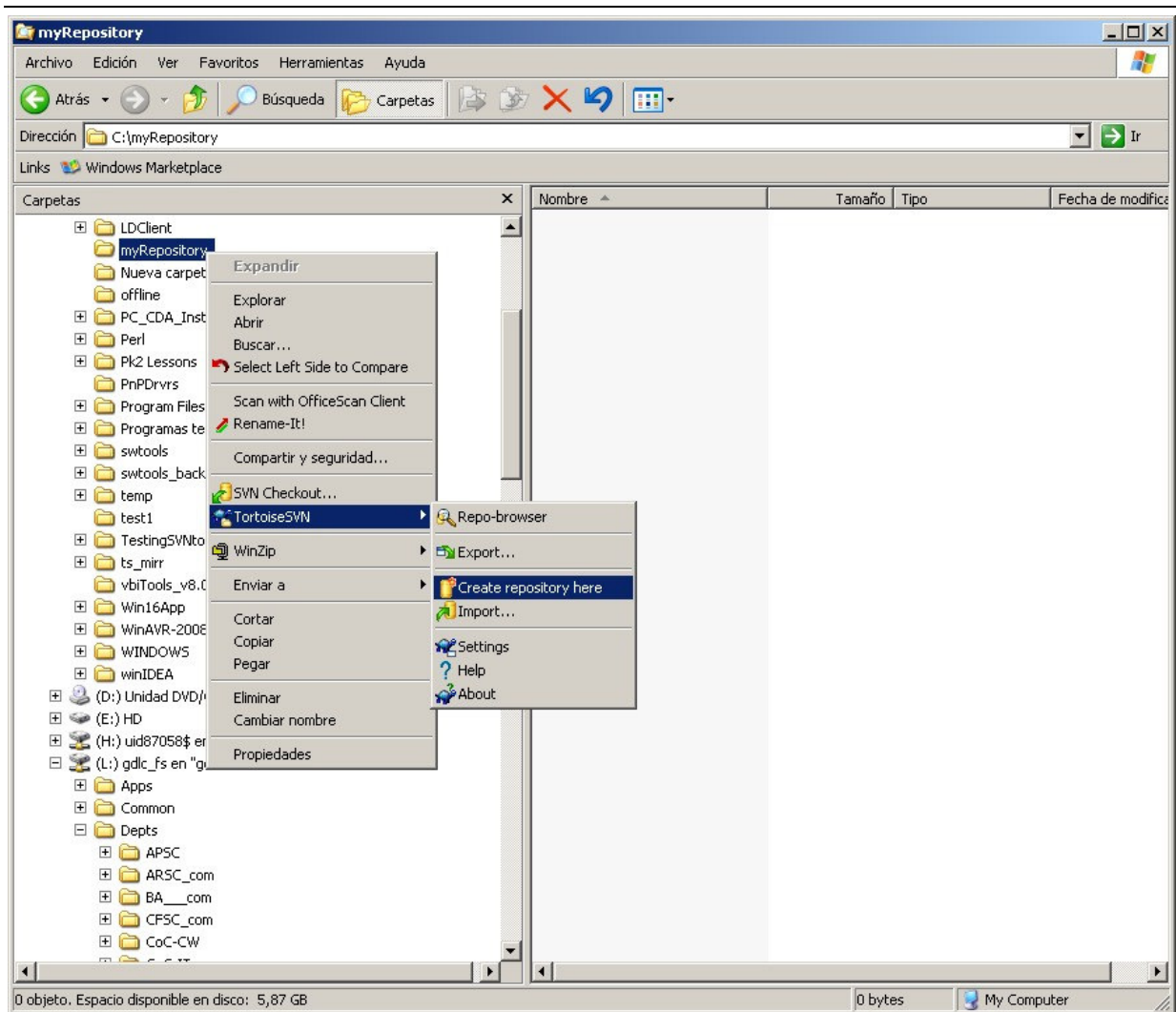
Descargar el paquete de instalación más reciente de la página <http://tortoisesvn.net/downloads>. Ejecutar el instalador y seguir las instrucciones en pantalla. Esto se tiene que hacer para todas las computadoras en las que piensen usar para el proyecto y en las que quieran hacer cambios (de cualquier documento o archivo del proyecto). Por ejemplo, si el equipo se conforma de 5 personas y cada una tiene computadora entonces instale en cada una de ellas el programa TortoiseSVN así todos los integrantes podrán hacer cambios y tener acceso a los documentos. Los pasos 5.1



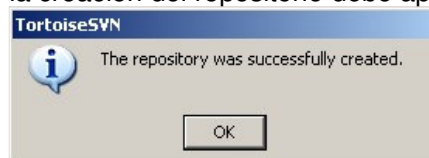
5.2 Creación de Repositorio (en memoria USB)

Crear una carpeta llamada myRepository en una memoria USB (ej. G:\myRepository). En el explorador de Windows posicionar el puntero del ratón sobre la carpeta recién creada, hacer click con el botón secundario y en el menú contextual dar click en TortoiseSVN -> Create Repository Here.

Nota: Aunque la imagen muestra que se creó en C, para fines prácticos créenla en una USB para que así puedan tener todo el desarrollo de una manera portable y pueda ser fácilmente usada/accedida por diferentes computadoras. En la "vida real" el repositorio se crearía en un servidor conectado a la red y podría ser accedida desde cualquier parte del mundo mediante Internet (si así se configurara).



Un mensaje notificándonos de la creación del repositorio debe aparecer.



Con esto se ha creado el repositorio que es la base de datos en donde se guardará la información necesaria para llevar el control de versiones de los archivos/proyectos que deseemos.

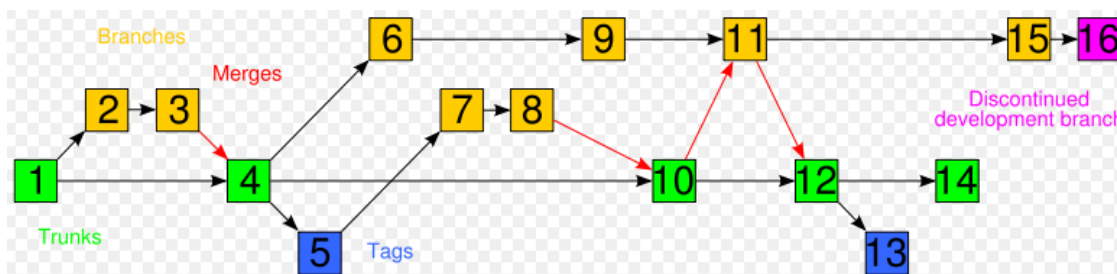
5.3 Creación de Estructura de un Proyecto

Una vez creado el repositorio lo siguiente es crear la estructura (carpetas) del proyecto con el que vamos a trabajar. Cabe notar que un repositorio puede contener uno o varios proyectos, la recomendación es la siguiente:

- Un solo proyecto por repositorio: Crear un repositorio por proyecto siempre que estos sean independientes entre sí es una buena práctica.
- Varios proyectos en un repositorio: Se recomienda crear varios proyectos en un mismo repositorio cuando estos guardan una relación entre sí, es decir puedan considerarse como 'subproyectos'. Un ejemplo común en el ámbito de sistemas embebidos sería crear

dentro de un mismo repositorio dos proyectos o mejor dicho subproyectos uno que correspondería al 'bootloader' y otro a la 'aplicación', ambos están relacionados y usualmente se entregan al cliente juntos para poder cargárselos como un todo al microcontrolador destino.

De la teoría recordemos que un proyecto puede tener la línea principal de desarrollo (trunk), una línea de ramificaciones (branches) y otra más de versiones estables etiquetadas (Tags). Esto lo podemos visualizar en la siguiente figura:



Para hacer esto posible se crean carpetas para cada línea de trunk, branches y tags dentro del repositorio, existe una recomendación para crear dicha estructura de carpetas, detalles sobre las diferentes posibilidades pueden verse en el documento /R2/ en la sección "4.1.5. Repository Layout".

Para esta práctica crearemos la siguiente estructura de carpetas dentro del repositorio.

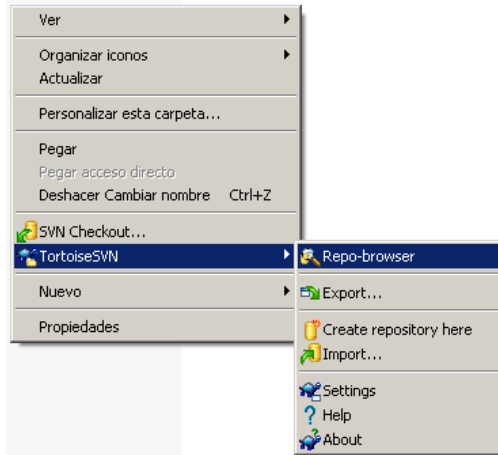
```
myRepository/trunk
myRepository/branches
myRepository/tags
```

Dentro de la línea principal de desarrollo (carpeta trunk) crearemos la carpeta para nuestro nuevo proyecto; le llamaremos "dummyProject".

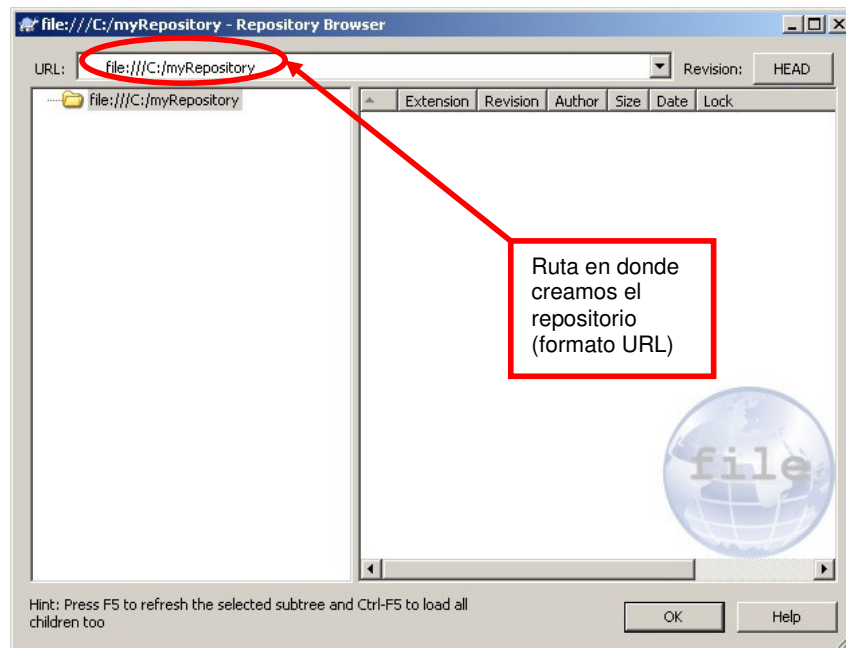
```
myRepository/trunk/dummyProject
```

Toda esta creación de carpetas la haremos utilizando el **"Repository Browser" del TortoiseSVN y NO el explorador de Windows**. Los pasos serían los siguientes.

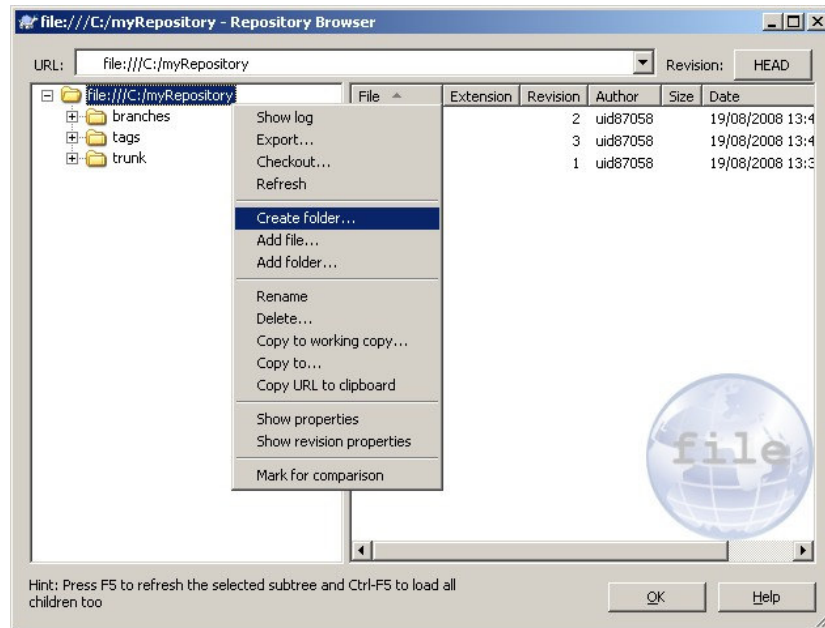
1. Abrir el "Repository Browser". En el explorador de Windows hacer Click con el botón secundario del ratón sobre cualquier carpeta, en el menú contextual hacer click en TortoiseSVN ->Repo-browser.



El repositorio se abrirá, si no aparece la ruta URL correcta verificar que lo sea, para nuestro ejemplo esta ruta sería "`file:///G:/myRepository`". **Nota:** Si esta computadora detecta la memoria USB con otra letra entonces en lugar de G: replázala por la letra que se le haya asignado a la memoria USB.



2. Con la ventana de "Repository-Browser" haciendo click con el botón secundario sobre nuestro repositorio y usando el comando "Create Folder..." crear cada una de las carpetas mencionadas (trunk, branches, tags y dummyProject dentro de la carpeta trunk).



La estructura de directorios debe quedar como sigue:

```
myRepository -----|---> branches
                    |---> tags
                    |---> trunk ---> dummyProject
```

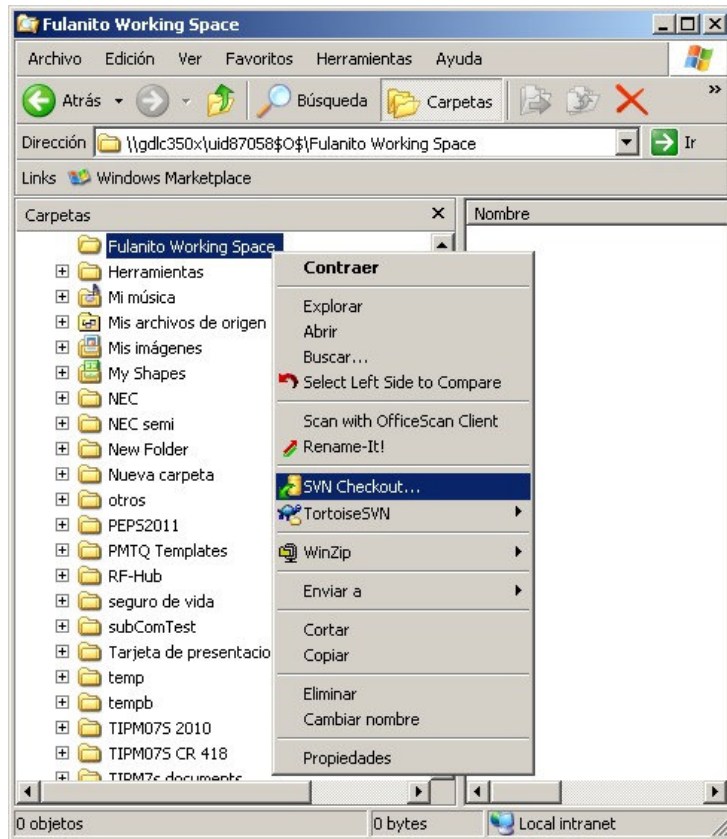
Hasta aqui hemos creado la estructura principal de nuestro proyecto llamado dummyProject en el repositorio myRepository.

5.4 Agregar archivos al Proyecto

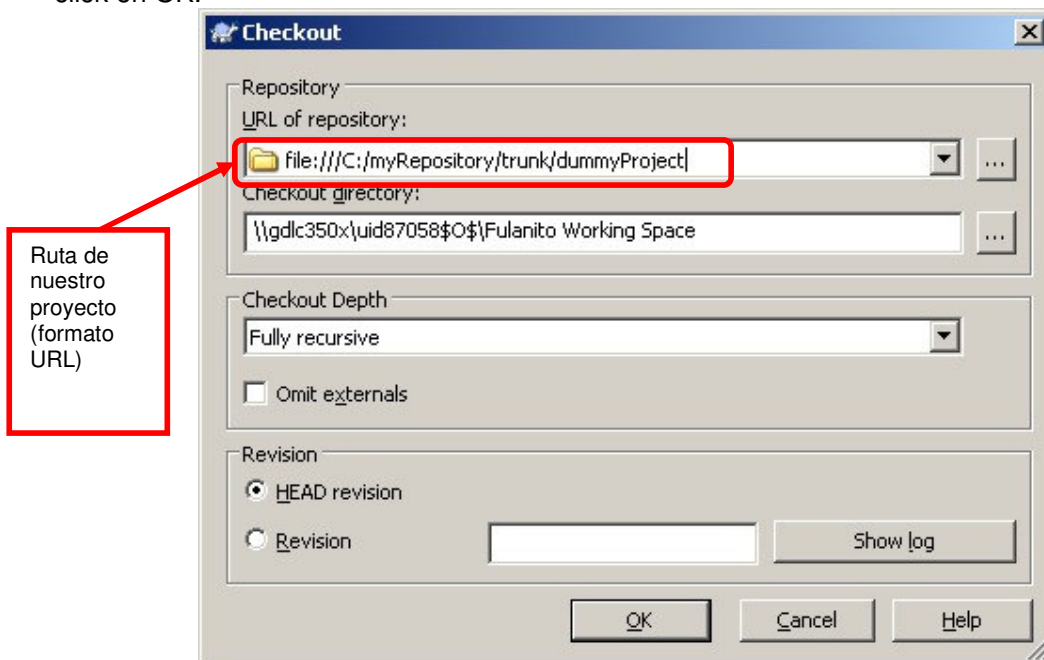
Una vez que tenemos lista la estructura de carpetas para nuestro proyecto comenzaremos a agregar archivos al mismo (archivos de código fuente, documentos, nuevas subcarpetas, etc..).

Para ello crearemos primero un "work area" con el comando "checkout", dentro de este "work area" colocaremos los archivos/carpetas que deseemos ingresar al repositorio de nuestro proyecto y por ultimo haremos un "commit" para actualizar el repositorio.

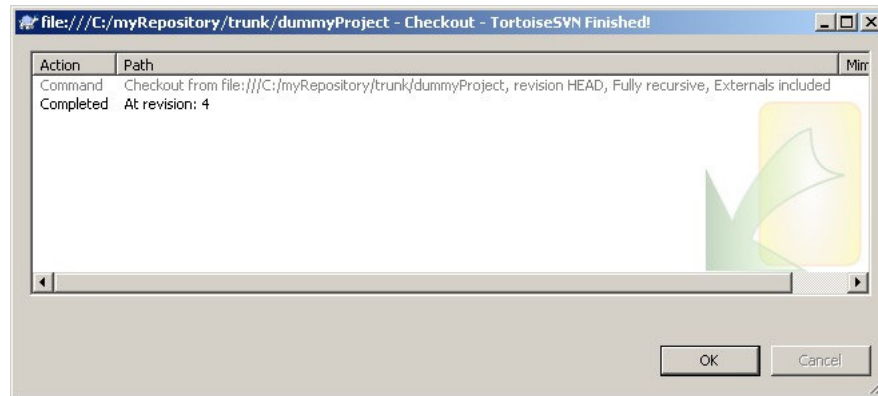
1. Con el explorador de Windows crear en 'Mis documentos' una carpeta llamada "Fulanito Work Spaces" (substituir Fulanito por un nombre propio), después haciendo click con el botón secundario sobre esta carpeta, en el menú contextual hacer click en "checkout".



2. En la ventana de "Checkout" en el campo "URL of repository" escribimos el URL de nuestro proyecto. Para este ejemplo el URL sería <file:///G:/myRepository/trunk/dummyProject>. Hacer click en OK.



El campo "checkout directory" se deberá poner automáticamente.
Una figura como la siguiente nos deberá aparecer para indicarnos que el "checkout" se ha realizado correctamente.



Un pequeño círculo verde aparecerá en el icono de la carpeta y se creará una carpeta oculta `.svn` que no debemos modificar bajo ninguna circunstancia.

3. Con el explorador de Windows creamos los siguientes archivos/carpetas en nuestra "work area" recién creada.

Carpetas:

- Fulanito Working Space\docs
- Fulanito Working Space\source
- Fulanito Working Space\testing

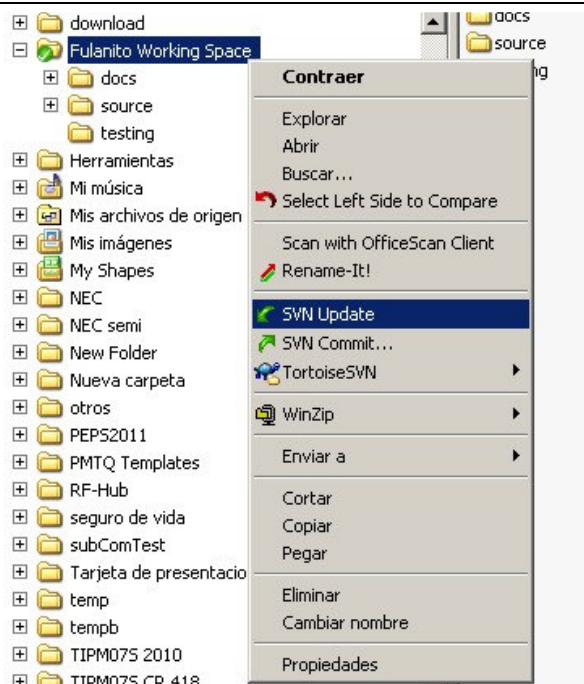
Archivos (sin contenido):

- Fulanito Working Space\docs\readme.txt
- Fulanito Working Space\source\main.c
- Fulanito Working Space\testing\testPlan.doc

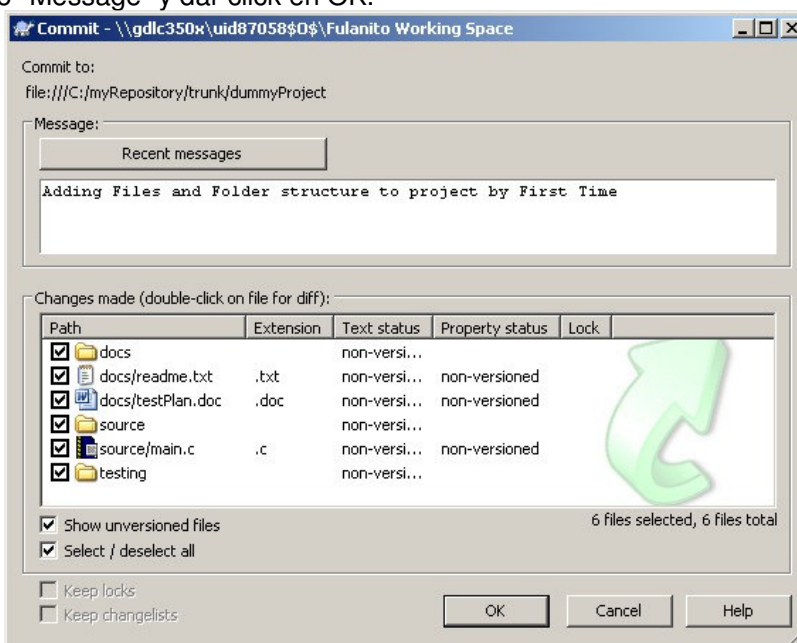


4. Hacer click con el botón secundario del ratón sobre la carpeta "Fulanito Working Space" y en el menú contextual seleccionar el comando "SVN Commit..".

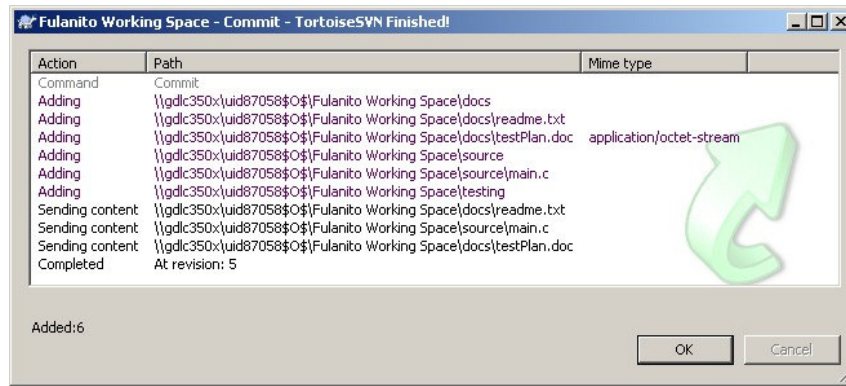
The Freescale Cup Manual de Práctica (TortoiseSVN)



En la ventana que aparecerá seleccionar todos los archivos, escribir un comentario adecuado en el recuadro "Message" y dar click en OK.



Si no hay ningún problema aparecerá una ventana como la siguiente:



Con esta operación "commit" hemos actualizado el repositorio de nuestro proyecto agregando a éste los archivos/carpetas que creamos en el "work area", a grandes rasgos la operación "commit" actualiza el repositorio con los cambios que hayamos hecho en nuestra "work area".

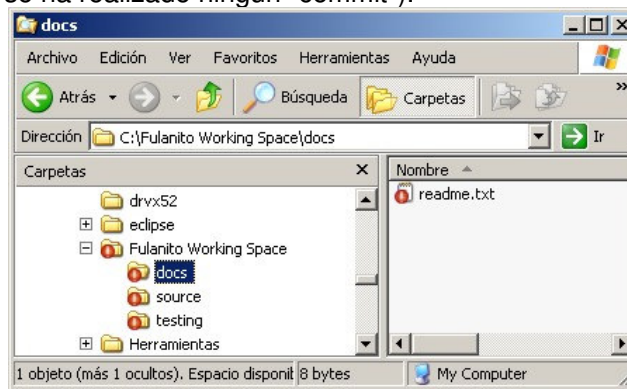
Ahora los archivos readme.txt, main.c, testPlan.doc y todo aquello que pongamos dentro del "work area" estará bajo control de versiones, nótese los iconos sobre las carpetas/archivos dentro de nuestra "work area".

5.5 Operaciones Commit / Update

La operación "commit" actualiza los cambios hechos en el "work area" al repositorio para hacerlos visibles por otros usuarios y para "publicarlos".

Para ejemplificar el uso de la operación "commit" haremos lo siguiente:

1. Agregar una línea de texto en el archivo readme.txt, escribir la función void main(void) en el archivo main.c, agregar una tabla al archivo testPlan.doc.
Al hacer todos estos cambios, los archivos tendrán ahora el icono con una cruz roja, esto nos indica que se han hecho cambios recientemente y que dichos cambios no están actualizados en el repositorio (no se ha realizado ningún "commit").

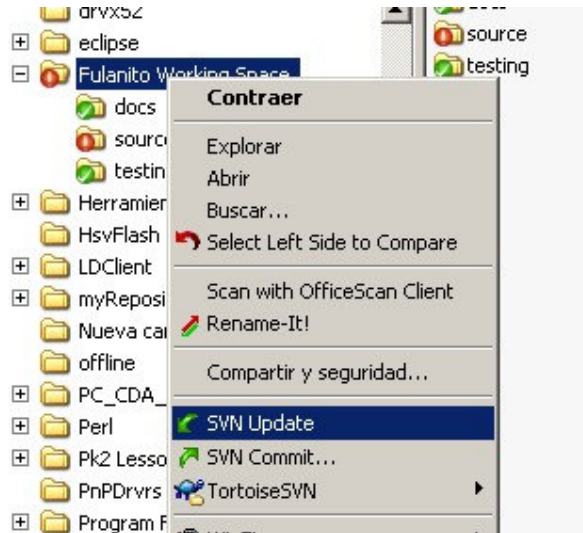


2. Hacer "commit" sobre la carpeta "Fulanito Working Spaces" para actualizar el repositorio con estos cambios. Los iconos cambiarán nuevamente a círculos verdes con palomitas indicando que el repositorio está actualizado con los cambios hechos en el "work area".
La operación "commit" se puede hacer sobre carpetas completas o sobre archivos individuales. Cada vez que se realice un "commit" el número de revisión del repositorio se incrementará.
3. Hacer cambios a cada uno de los archivos y hacer "commits" individuales para cada archivo.

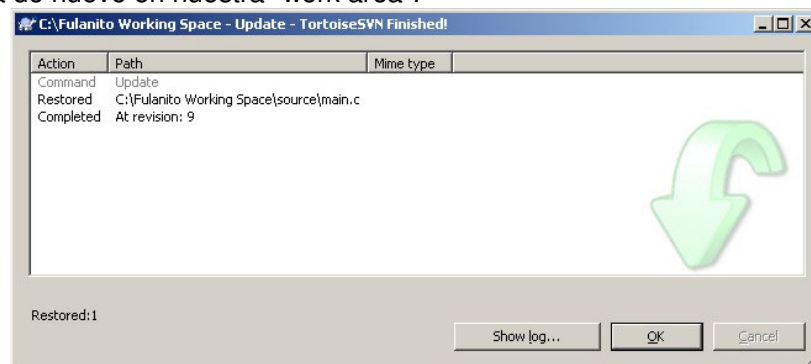
La operación "Update" nos permite actualizar nuestra "work area" desde el repositorio, podría verse como la operación inversa a "commit". Algunos ejemplos del uso del comando "update" son los siguientes:

Recuperar archivos borrados del work area.

- 1.- Desde el explorador de Windows eliminar el archivo main.c de la carpeta "source" en el "work area". Borrarlo también de la papelera de reciclaje.
- 2.- Para recuperar el archivo que acabamos de borrar sobre la carpeta del "work area" (Fulanito work space) hacemos un "update".



Esta operación actualizará nuestra "work area" con el contenido del repositorio, y como la última revisión del repositorio (desde el último "commit") contenía el archivo que acabamos de eliminar, éste aparecerá de nuevo en nuestra "work area".



Obtener cambios hechos por otros usuarios

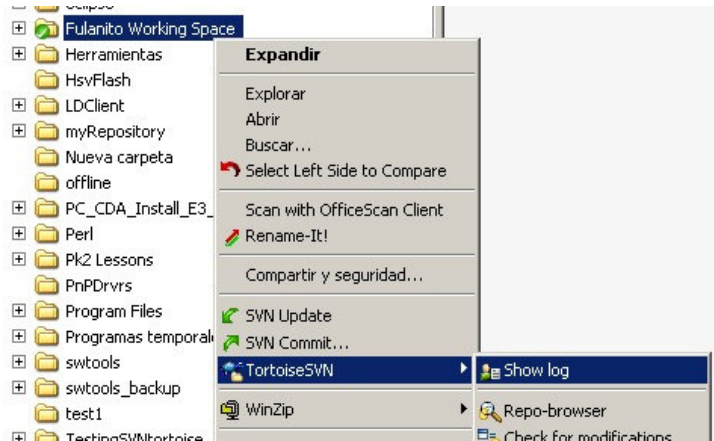
1. Crear una nueva "work area" en otra ubicación. En esta nueva "work area" hacer modificaciones en algunos de los archivos y hacer "commit" para actualizar el repositorio.
2. En el "work area" Fulanito work space, ejecutar la operación "SNV Update".
3. Verificar que los cambios hechos en la otra "work area" se han actualizado en el "work area" de Fulanito work space.

5.6 Comparar Versiones / Lista de cambios (Change Log)

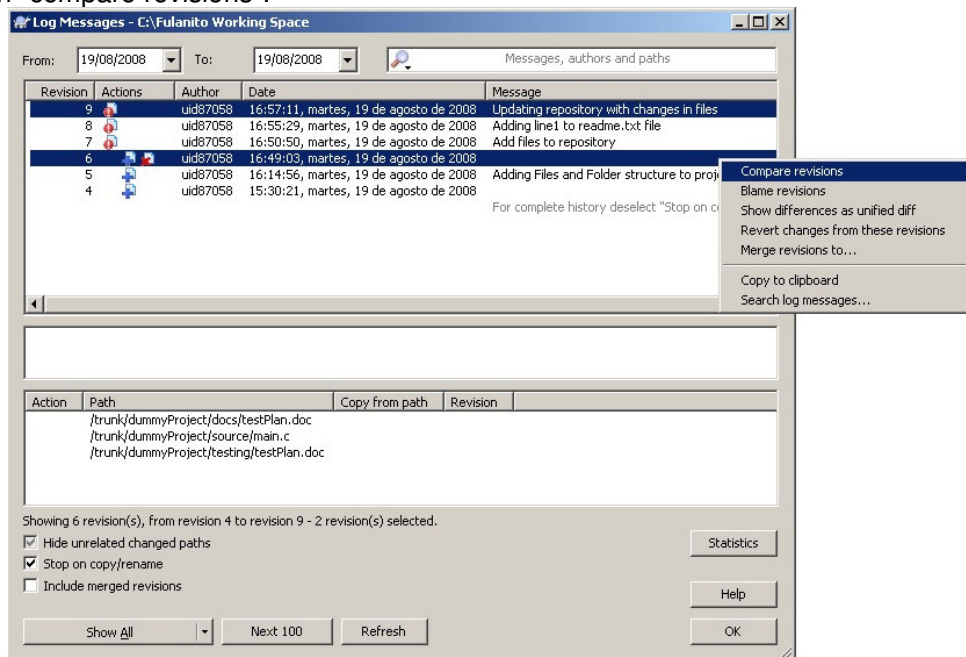
Se pueden ver los cambios hechos entre revisiones tanto a nivel carpetas como a nivel archivos.

Para ver cambios entre revisiones de carpetas:

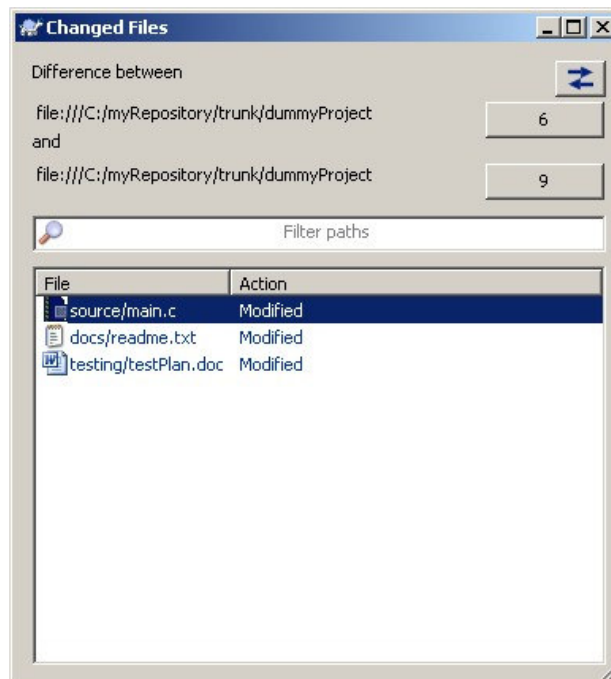
1. sobre una carpeta bajo control de versiones con TortoiseSVN hacer click con botón secundario, después en el menú contextual ir a TortoiseSVN -> Show Log.



2. Aparecerá una ventana "Log messages" aquí podremos ver una lista de las revisiones aplicables a la carpeta seleccionada y los comentarios (que hemos puesto) para cada revisión. Para ver que cambios se realizaron entre revisiones solo es necesario seleccionar 2 revisiones de las enlistadas en el Log y con click secundario, sobre el menú contextual, hacer click en "compare revisions".



3. Aparecerá una ventana mostrando las diferencias. Como esta comparación es a nivel carpeta los cambios harán referencia a archivos agregados, modificados, eliminados, etc.



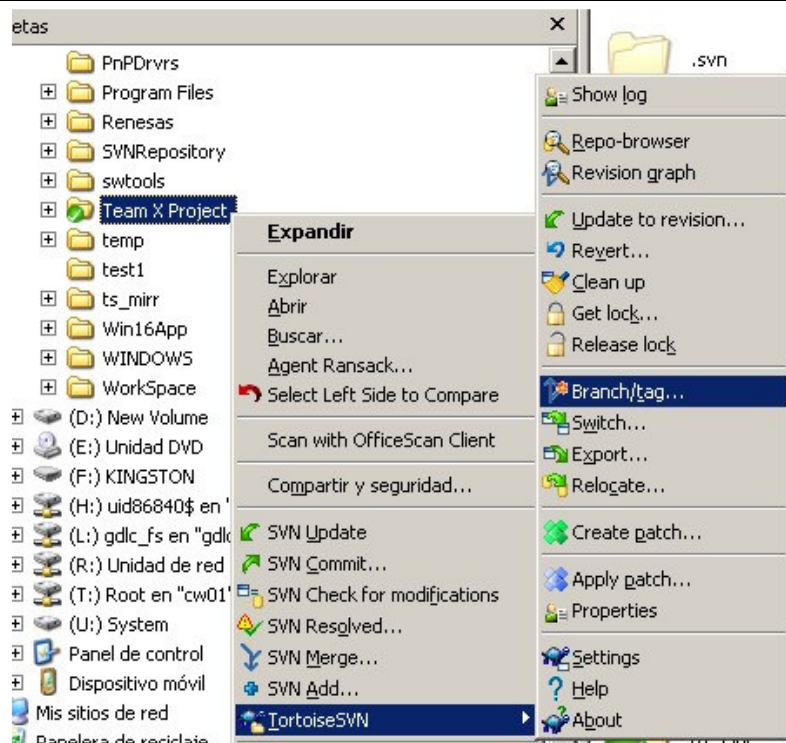
Cambios entre revisiones de archivos:

5.7 Creación de Rama (Branching / Tagging)

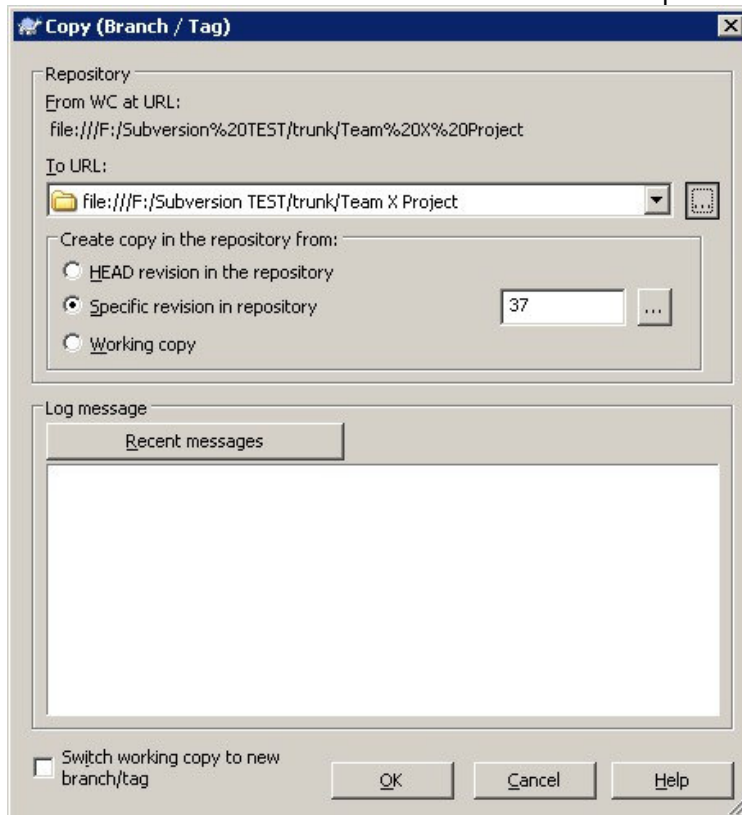
Hacer una rama (Branch) significa hacer una copia de un proyecto el cual también se desarrollará o modificará en forma paralela al proyecto original. Esto es muy útil cuando se requiere agregar o probar funcionalidades a un proyecto y también da la facilidad de poder desarrollar en paralelo.

En los siguientes pasos se muestra como crear una rama:

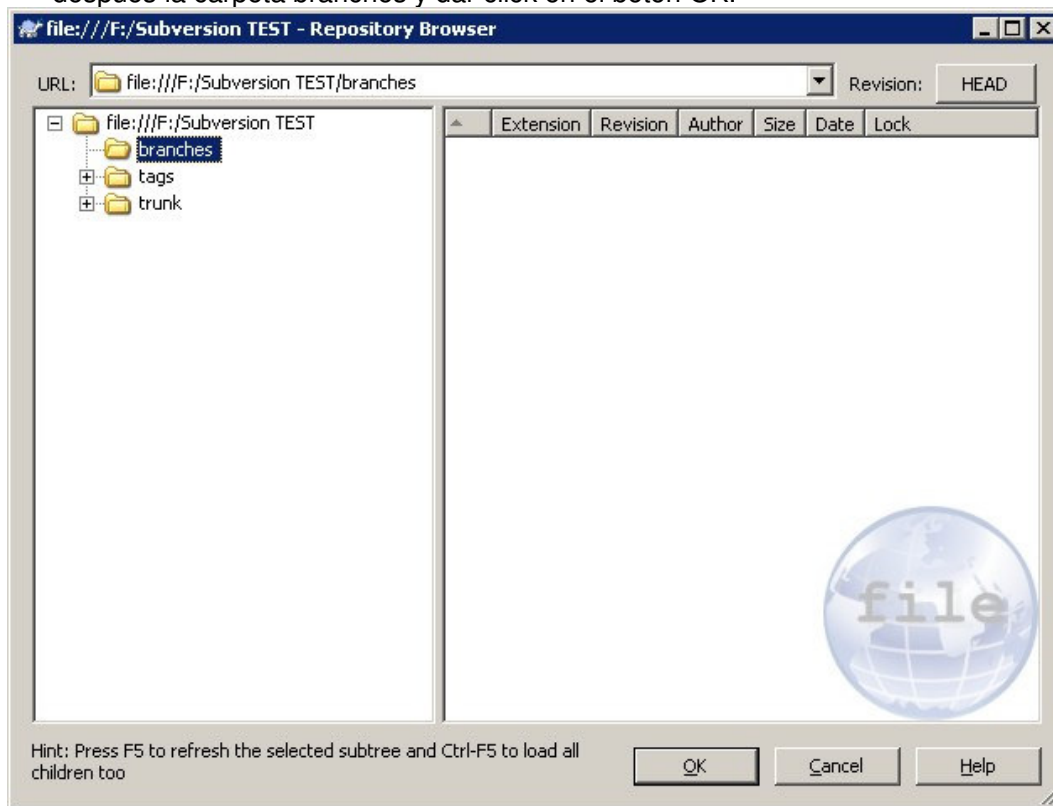
1. Seleccionar la carpeta (proyecto) del que se quiere hacer la rama y en el explorador dar click derecho y seleccionar TortoiseSVN->Branch/Tag...



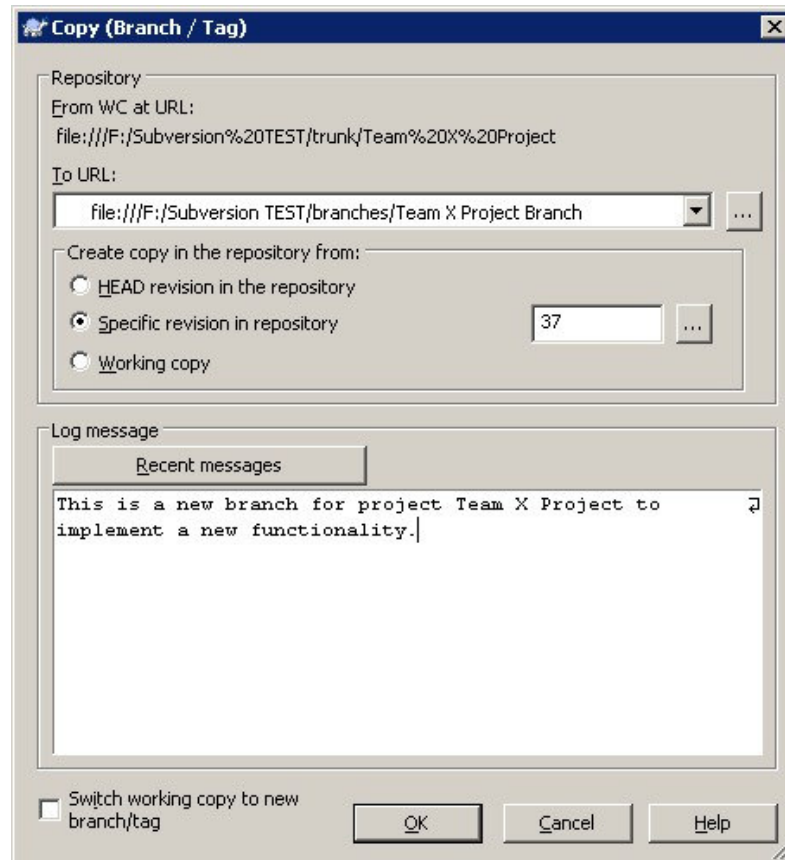
2. Aparecerá el siguiente cuadro de diálogo de donde se seleccionará la carpeta destino de la rama. Cambiar el Path de To URL: seleccionando el cuadro con los puntos suspensivos "..."



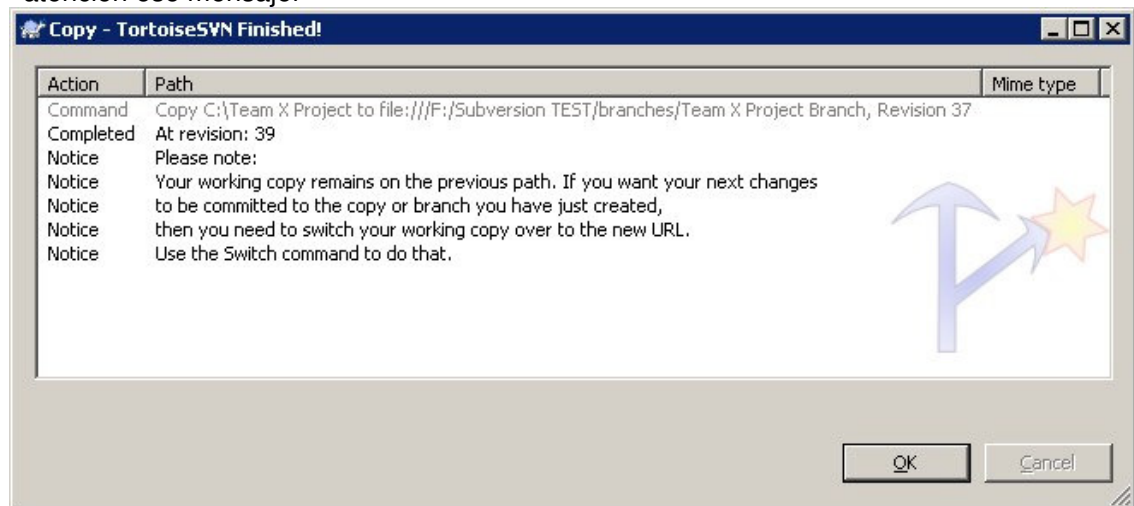
3. En la siguiente ventana se tiene que dar doble click en el repositorio (carpeta de hasta arriba en el cuadro izquierdo) para poder ver las carpetas que contiene el repositorio y seleccionar después la carpeta branches y dar click en el botón OK.



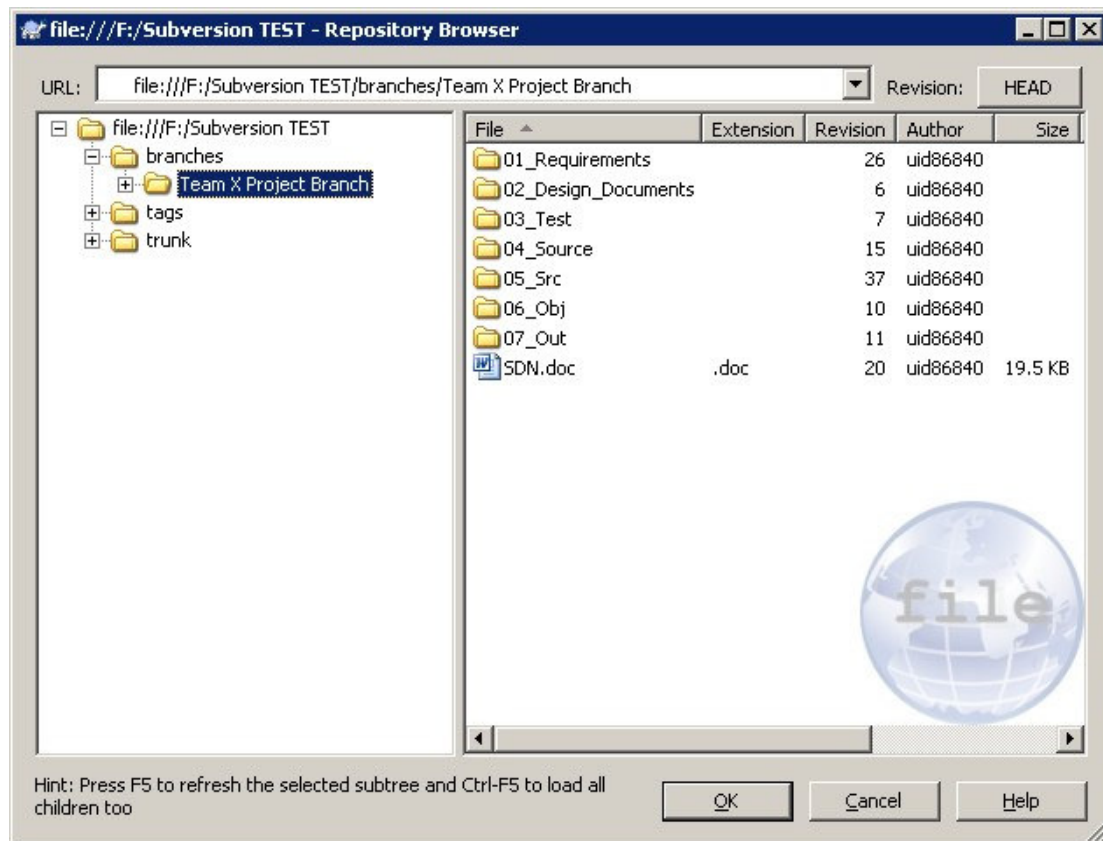
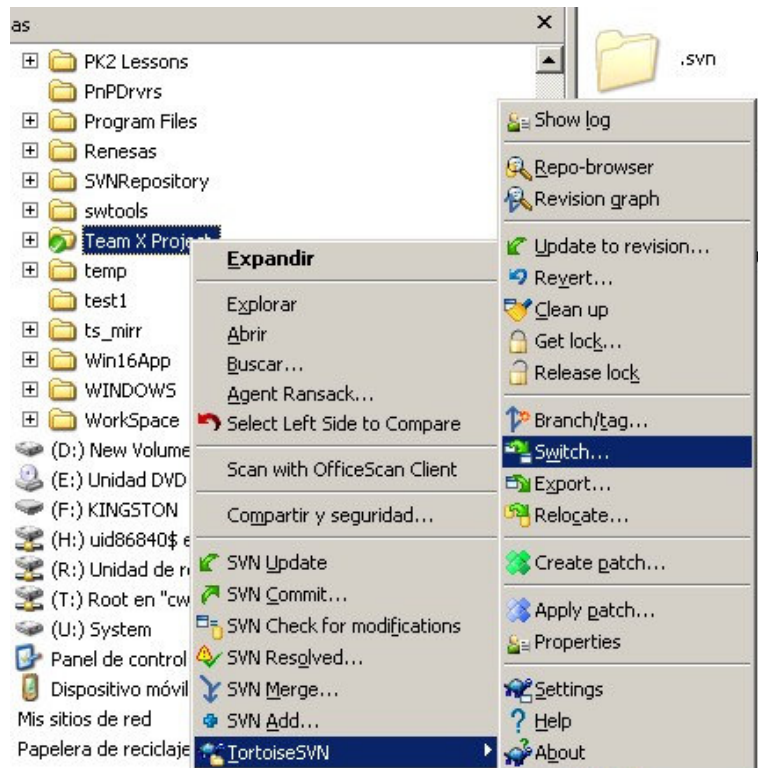
4. En la siguiente ventana agregar al Path una carpeta al final por ejemplo agregar: "/Team X Project Branch", que será la carpeta donde queremos que haga la copia de nuestro proyecto en Trunk. No olviden poner algún comentario relevante respecto a esta rama. Después presionar el botón OK.

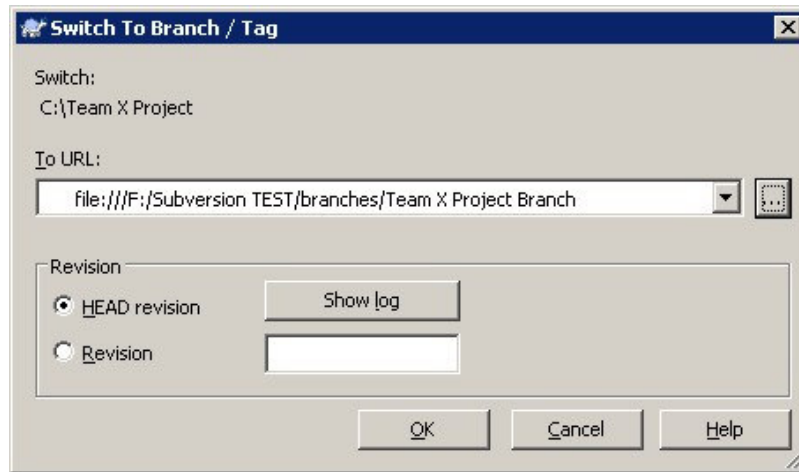


5. Te aparecerá un cuadro de diálogo similar al mostrado en la siguiente imagen. ¡Lean con atención ese mensaje!

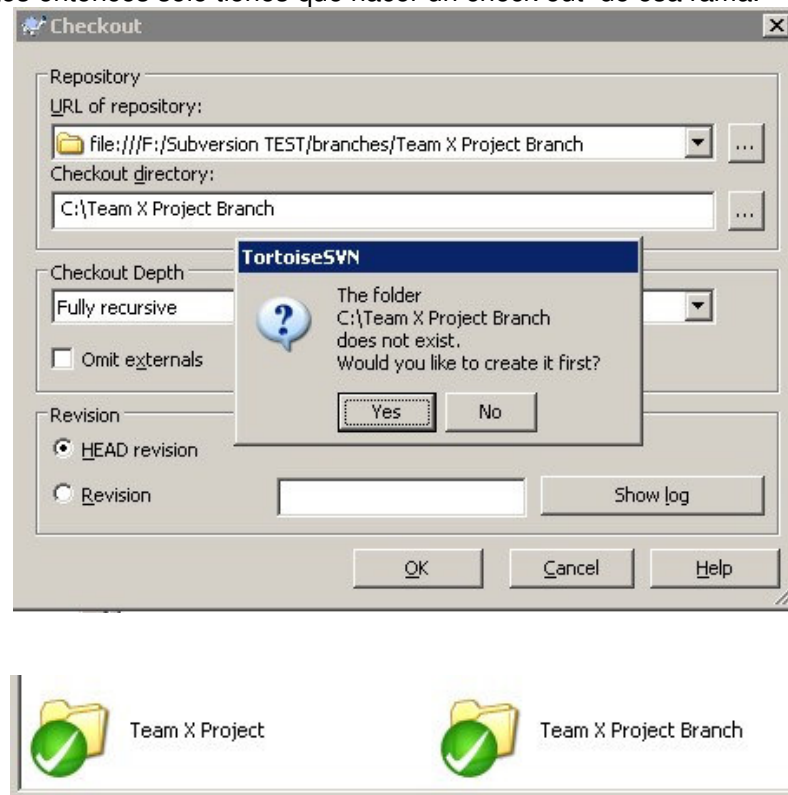


6. ¡Ahora ya tienes una rama de tu proyecto creada! Si quieres trabajar sobre ella entonces tienes que hacer un Switch en el "working copy" que estás usando o hacer un "check out" de tu rama en otro lugar. En las siguientes imagenes se muestra como hacer un switch de tu working copy solo recuerda seleccionar el path del branch que quieras hacer el switch.



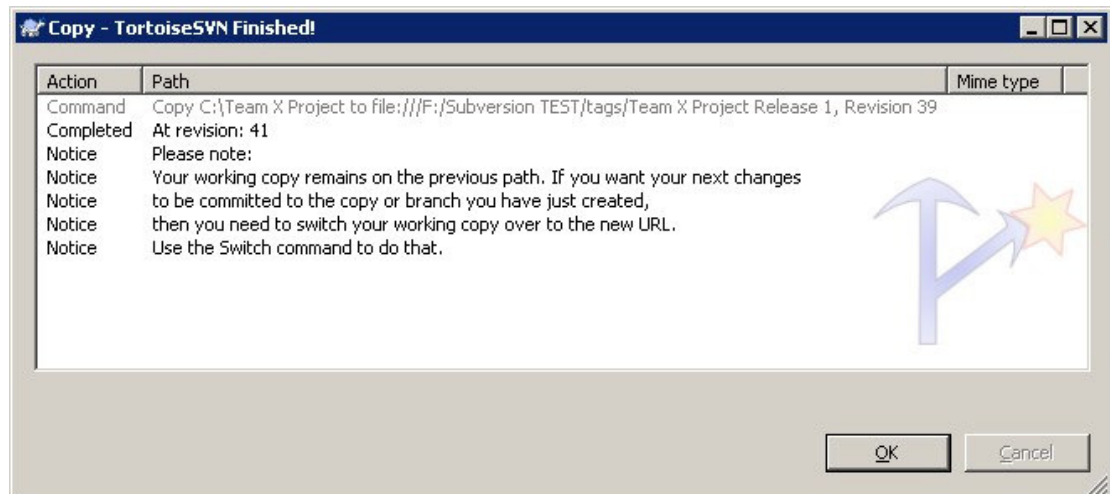
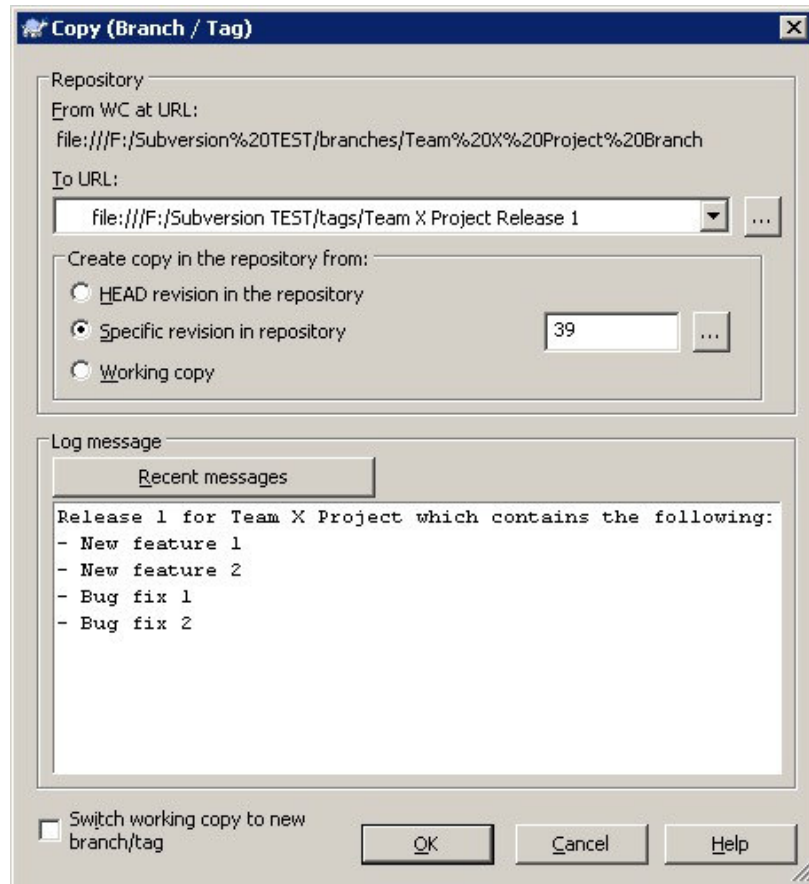


7. Si quieres crear otro working copy (working area) de el branch para poder desarrollar en los dos proyectos entonces solo tienes que hacer un check out de esa rama.



Una etiqueta (Tag) se usa para hacer una copia del proyecto en una versión específica que queremos que sea vista como una entrega o "release". En este "Tag" no se debe usar para continuar con el desarrollo sino usar la rama "trunk" de donde se tomó el "Tag" o en una rama.

1. Para crear una etiqueta (Tag) se usa el mismo procedimiento que para crear una rama solo que se tiene que seleccionar un nombre distinto (que tenga significado) y se tendrá que poner en la carpeta "Tags" que se encuentra en el repositorio. Ver imagen siguiente como referencia.

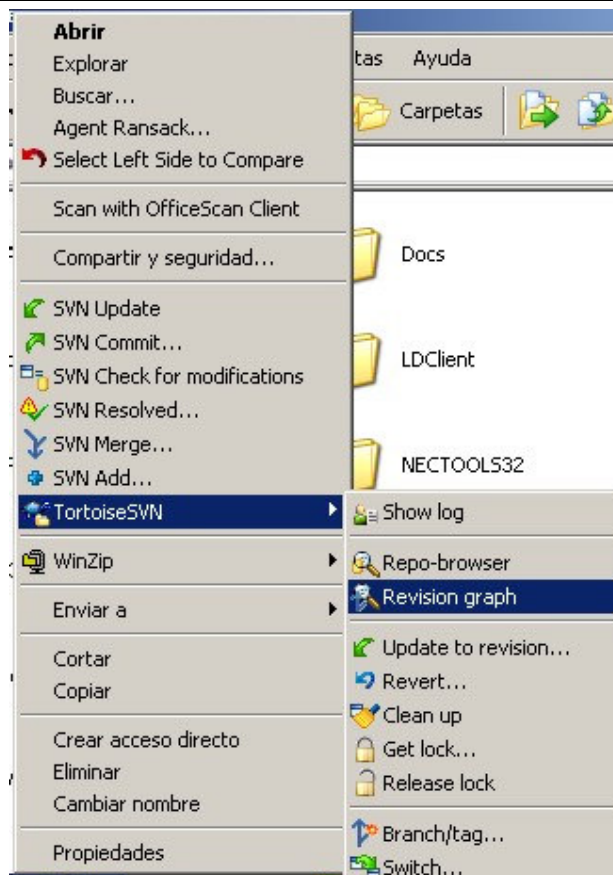


2. ¡Listo! Has creado una etiqueta. Recuerda que no debes hacer un "commit" a partir de un working copy de una Tag.

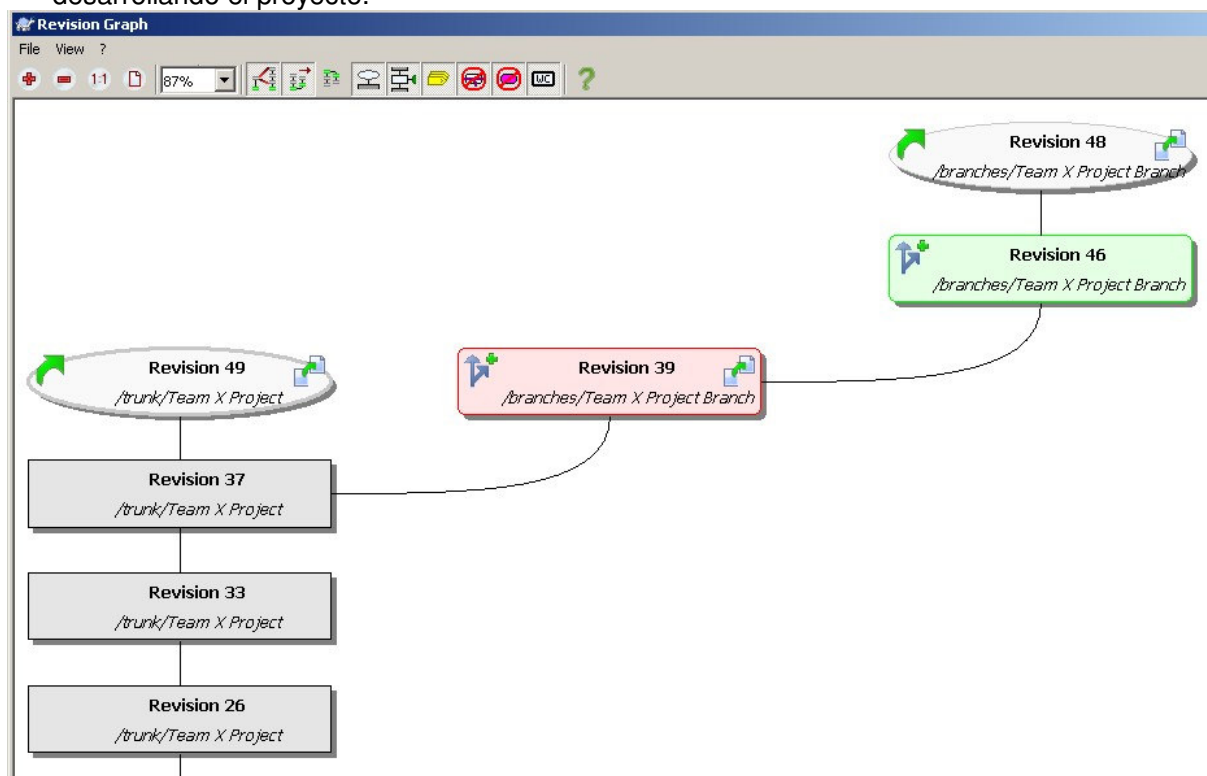
5.8 Gráfica de revisiones



Ahora que tenemos una rama y una etiqueta vamos a ver cual es su relación entre ellas mediante una gráfica de revisiones. Para esto sigan los siguientes pasos:

1. Click derecho->TortoiseSVN->Revision graph. (hacerlo sobre una de las versiones)



2. Aparecerá una ventana similar a la siguiente. La utilidad de esto es ver como se ha ido desarrollando el proyecto.



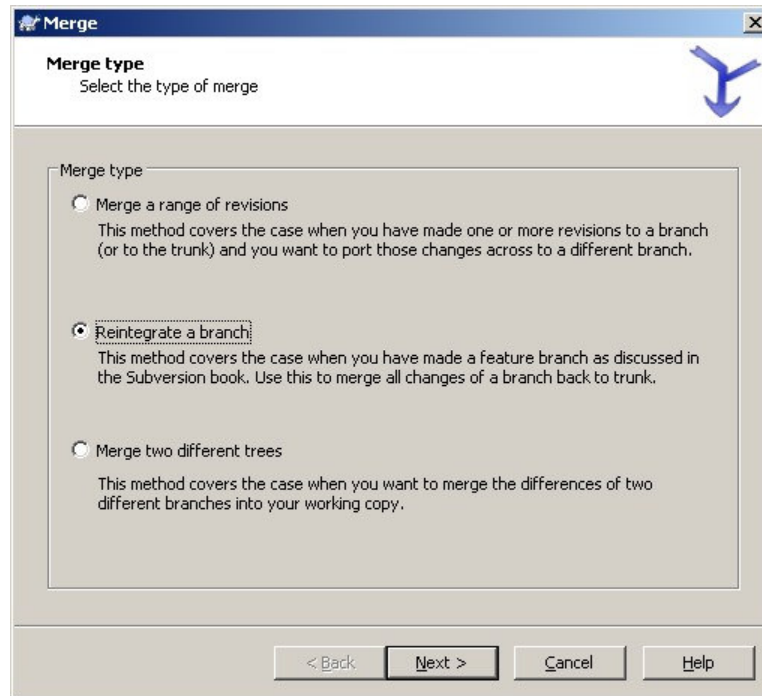
3. El símbolo  significa que ahí se ha hecho una etiqueta (Tag).
4. El símbolo  significa que está es una rama.
5. Esta ventana tiene varias opciones para visualizar de diferentes maneras, intenta ver las diferentes opciones que ofrece.
6. También puedes seleccionar dos versiones diferentes y revisar cuales son las diferencias.

5.9 Integrar modificaciones (Merge)

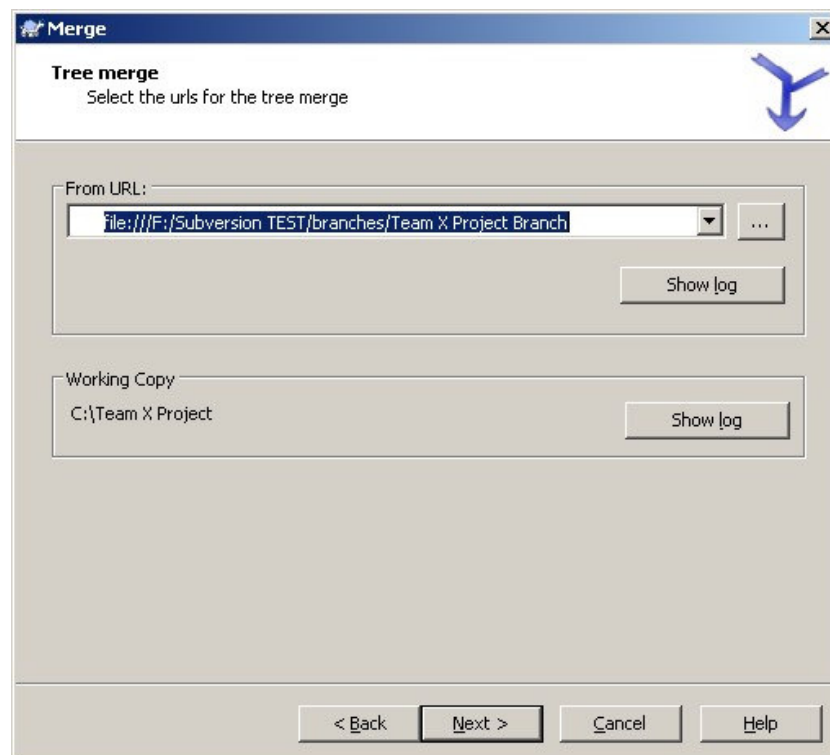
Ya que se han hecho cambios en la rama y se tiene un avance, podría quererse incluir esos cambios al trunk para seguir con el desarrollo del programa. Para incluir los cambios se tiene que integrar (hacer merge) del branch al trunk. Ten cuidado incluir los cambios puesto que podrías sobre escribir cambios hechos en el trunk. Para hacer un merge a continuación se detallan los pasos:

1. Sobre la rama trunk (el destino de la integración) da click derecho->SVN Merge y luego selecciona la opción "Reintegrate a branch", dale next

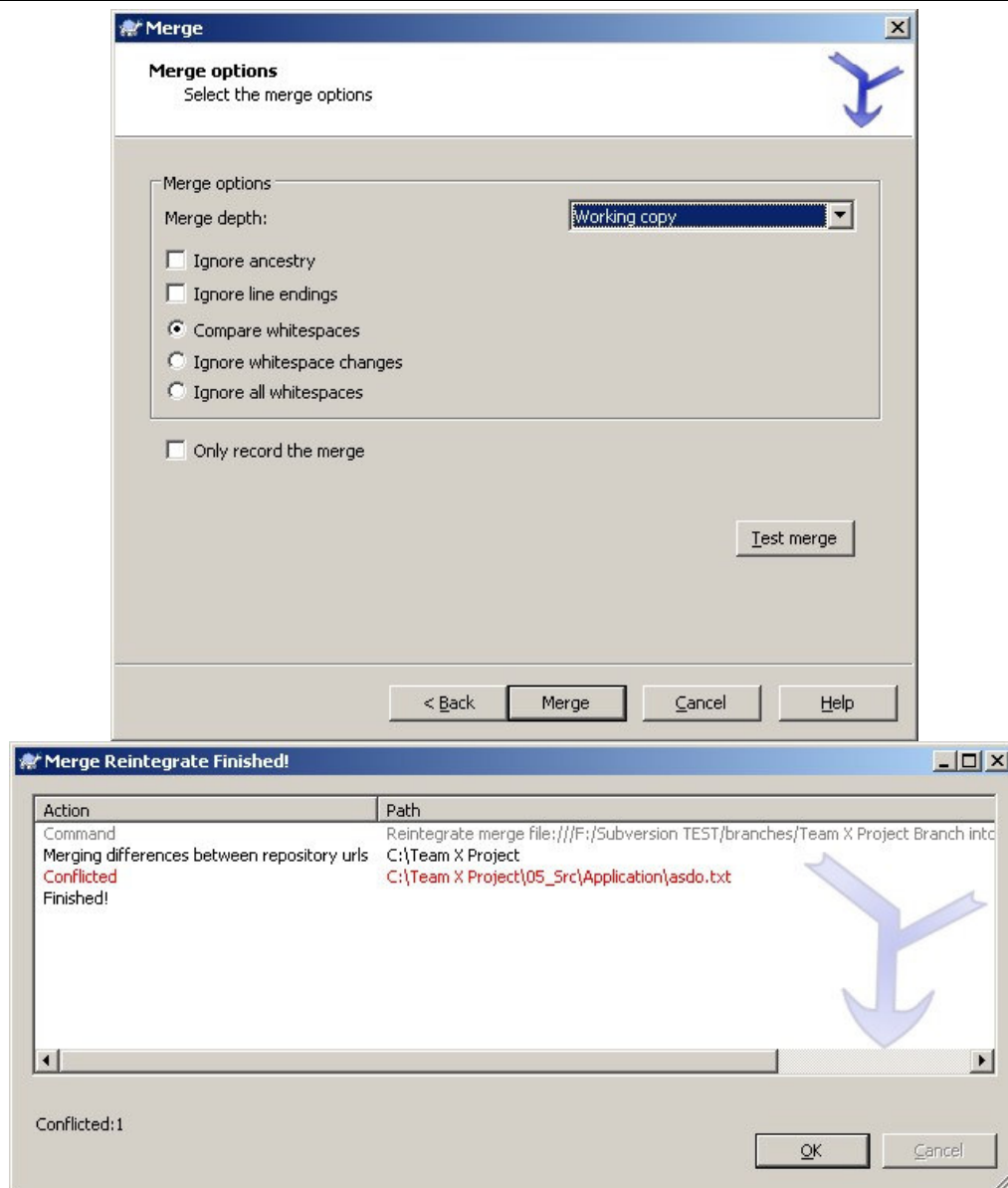




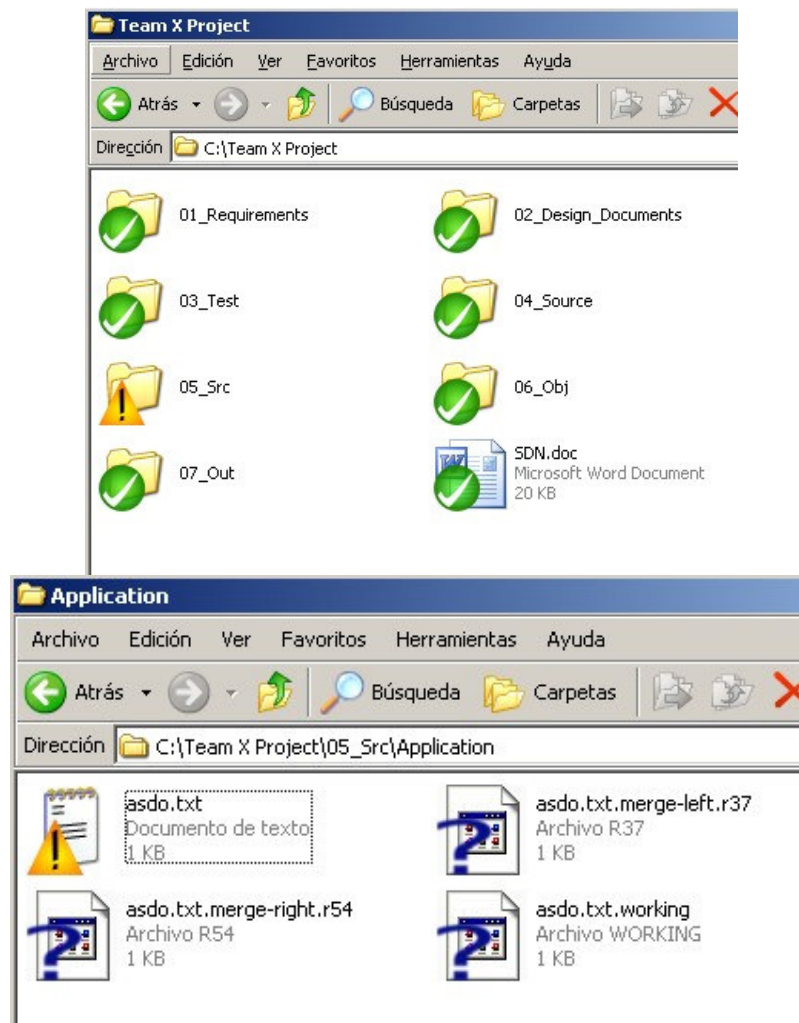
2. Selecciona de donde quieres integrar (de que rama quieres traer los cambios) y después selecciona "next".



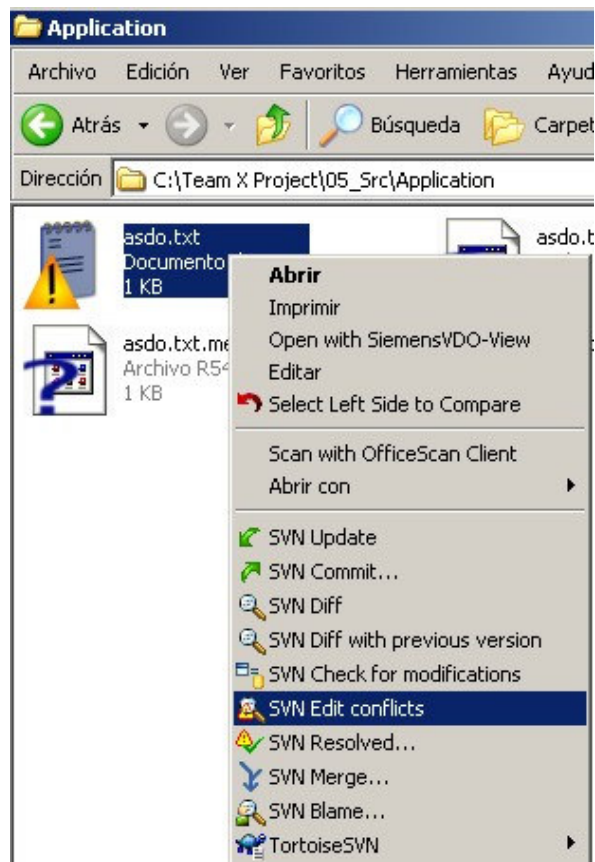
3. En la siguiente ventana presiona "Test merge" para ver una vista previa de como hará el merge y cuales archivos están en conflicto (archivos que cambiaron tanto en el branch como en el trunk).



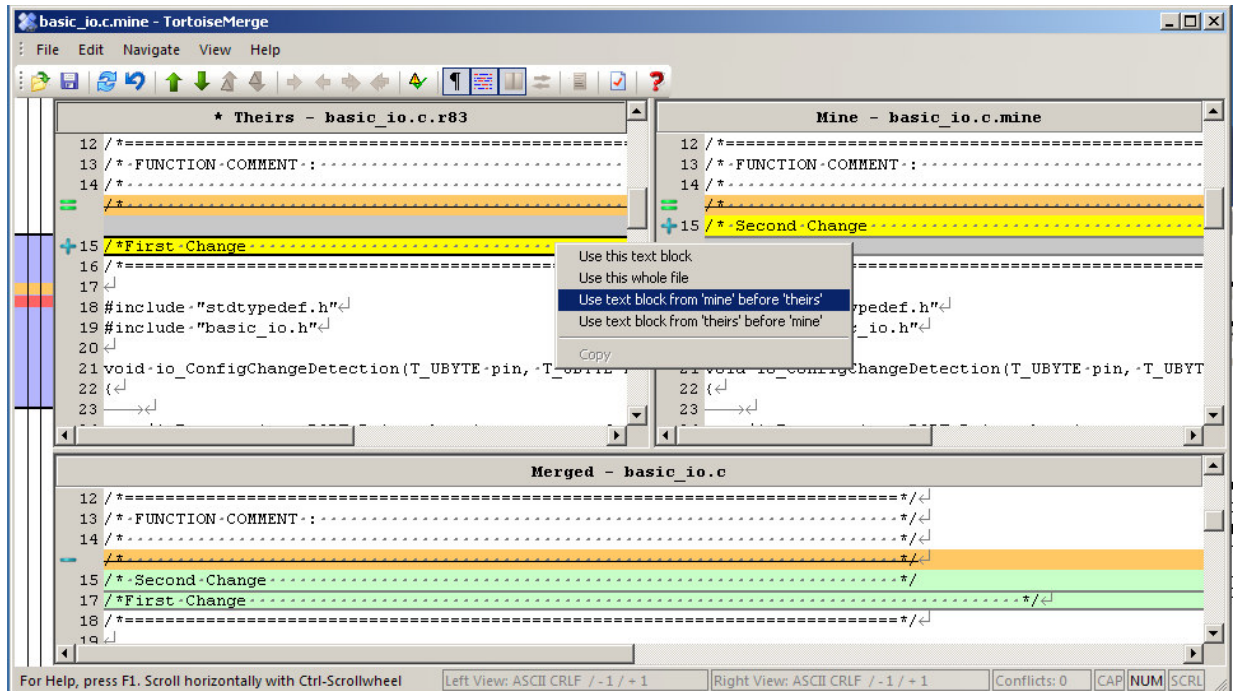
4. Después presiona Ok y en la ventana anterior haz el merge, te aparecerá la misma ventana que el "test merge" sin embargo ahora si hará el merge. Si hubo algún archivo con conflicto se verá reflejado en las carpetas/archivos del working copy.



5. Para resolver el conflicto, se tiene que ir al archivo y dar click derecho y seleccionar SVN edit conflict.



6. Aparecerá una pantalla similar a la mostrada abajo, donde del lado superior izquierdo se muestra una versión del archivo en conflicto y del lado superior derecho se ve la otra versión, se resaltan las diferencias, las cuales se tienen que resolver. En la parte inferior aparece el archivo que resultará de combinar las dos versiones de archivos para incluir un cambio u otro selecciona la línea resaltada y da click derecho y selecciona una de las opciones según los cambios que quieras incluir. Es importante que cuando se haga el merge sepan que hace cada cambio para no sobrescribir un cambio con otro.



Una vez hecho esto ya no habrá conflicto en el archivo y podrás "subirlo" al "servidor" con un commit.

5.10 Simular Actividades de Equipo antes de crear su proyecto

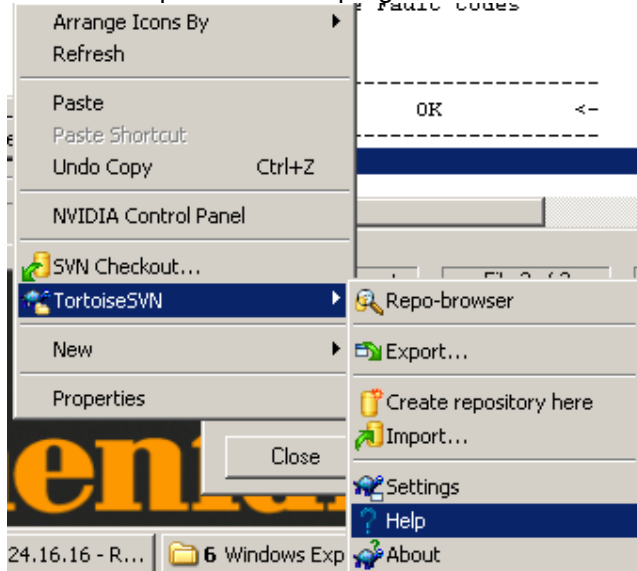
Practiquen con un proyecto piloto o muestra y revisen los demás comandos y opciones que tienen la herramienta para que practiquen antes de crear el proyecto a entregar.

Para mayor información puede consultar las siguientes paginas:

<http://tortoisesvn.tigris.org>

<http://tortoisesvn.net/support>

Recuerden que también el programa tiene una buena sección de ayuda



Para su proyecto real ya no necesitarán crear otro repositorio pero sí crear un proyecto nuevo como se muestra en la sección 5.3.