

TEMA 4: OPTIMIZACIÓN Y DOCUMENTACIÓN

Nota: Los 6 ejercicios de este documento se debe de entregar en formato pdf, con el enunciado de cada ejercicio y con captura de cada uno de los pasos realizados y una breve explicación. Cuando se solicite añadir un comentario, este comentario debe de ir precedido del nombre del alumno.

1.- Introducción

OBJETIVO: Que el alumno maneje herramientas de control de versiones, herramientas para documentar los programas y herramientas de refactorización

El uso de estas herramientas aseguran que los proyectos mantengan una trazabilidad de todos los cambios que han sufrido a lo largo de su vida, y quien a realizado los cambios. También garantiza que los proyectos estén bien documentados informando de lo que hace cada una de las clases y métodos que los forman, y bien refactorizados simplificando el código de los programas y favoreciendo su lectura, entendimiento y mantenimiento.

2.- Control de versiones

Se define control de versiones como la capacidad de recordar todos los cambios que se realizan tanto en la estructura de directorios como en el contenido de los archivos. Esto es de utilidad cuando se desea recuperar un documento, o una carpeta, o un proyecto en un momento concreto de su desarrollo.

TERMINOLOGÍA

Repositorio: Lugar donde se almacena todos los datos y los cambios. Puede ser un sistema de archivos en un disco duro, un banco de datos, un servidor, etc.

Revisión o versión: Una revisión es una versión concreta de los datos almacenados.

Etiquetar o Rotular (tag). Cuando se crea una versión concreta en un momento determinado del desarrollo de un proyecto se le pone una etiqueta, de forma que se pueda localizar y recuperar en cualquier momento. Las etiquetas permiten identificar de forma fácil revisiones importantes en el proyecto.

Tronco (trunk): Es el tronco o la línea principal de desarrollo de un proyecto.

Rama o ramificar (branch): Las ramas son copias de archivos, carpetas o proyectos. Cuando se crea una rama se crea una bifurcación del proyecto y se crean dos líneas de desarrollo. Son motivos de creación de ramas la creación de nuevas funcionalidades o la corrección de errores.

Desplegar (Checkout): Crear una copia de trabajo del proyecto, o de archivos y carpetas del repositorio en el equipo local. Por defecto se obtiene la última versión, aunque también se puede indicar una versión concreta. Con el checkout se vincula la carpeta de trabajo del equipo local con el repositorio, y se crean los metadatos de control de versiones.

Confirmar (commit o check-in). Se realiza commit cuando se confirma los cambios realizados en local para integrarlos al repositorio.

Exportación (export). Similar a Checkout, pero en esta ocasión no vincula la copia con el repositorio. Es una copia limpia sin los metadatos de control de versiones.

Importación (import). Es la subida de carpetas y archivos del equipo local al repositorio.

Actualizar (update). Se realiza una actualización cuando se desea integrar los cambios realizados en el repositorio en la copia de trabajo local.

Fusión (merge). Una fusión consiste en unir los cambios realizados sobre uno o varios archivos en una única revisión. Se suele realizar cuando hay varias líneas de desarrollo separadas en ramas y en alguna etapa se necesitan fusionar los cambios hechos entre ramas o en una rama con el tronco principal, o viceversa.

Conflicto: Ocurre cuando dos usuarios crean una copia local (Checkout) de la misma versión de un archivo, uno de ellos realiza cambios y envía los cambios (commit) al repositorio, y el otro no actualiza (update) esos cambios y realiza cambios sobre el archivo e intenta enviar luego sus cambios al repositorio. Entonces se produce el conflicto y el sistema no es capaz de fusionar los cambios. Este usuario deberá resolver el conflicto combinando los cambios o eligiendo uno de ellos.

Resolver conflicto: La actuación del usuario para atender un conflicto entre diferentes cambios al mismo documento.

2.1.-Subversion. Ciclo de vida de subversion.

Subversion es una herramienta multiplataforma e código abierto para el control de versiones. Usa una base de datos central, el repositorio, que contiene los archivos cuyas versiones y respectivas historias se controlan. El repositorio actúa como un servidor de ficheros, con la capacidad de recordar todos los cambios que se hacen tanto en sus directorios como en sus ficheros.

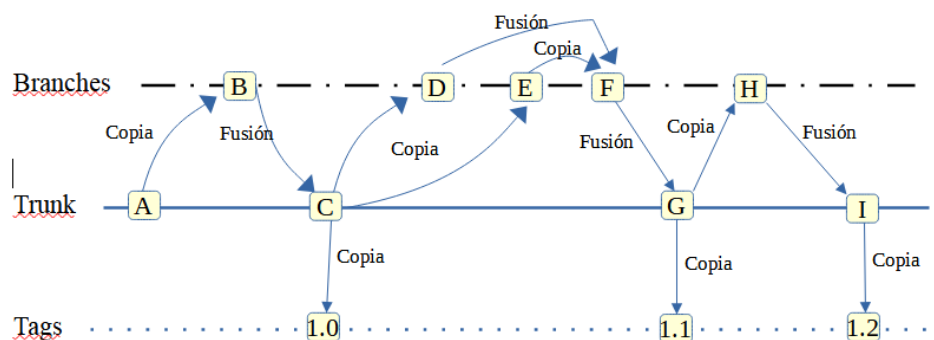
Cuando un proyecto de software es desarrollado por un equipo de personas es indispensable llevar un control bajo un método de trabajo, seguir unas normas y aplicar las buenas prácticas en el uso de las herramientas. Así el proyecto debe verse como un árbol que tiene un tronco (trunk) donde está la línea principal de su desarrollo; que tiene sus ramas (branches) en la que se añadirán nuevas funciones o se corregirán errores; y que además tiene sus etiquetas (tags) para marcar situaciones importantes, o versiones finalizadas. Así la estructura de carpetas recomendada en la creación de proyectos utilizando estas herramientas y la funcionalidad que se le debe dar a cada carpeta dentro del repositorio son las siguientes:

Trunk (tronco): base común para guardar las carpetas del proyecto o trabajo o controlar. Es donde está la versión básica, es decir, la rama de desarrollo principal.

Tags (etiquetas): una etiqueta es una copia del proyecto, de una carpeta o de un archivo que se hace con el objetivo de obtener una versión que no se va a modificar. Deben ser copias del tronco (trunk). Útil para crear versiones ya finalizadas, aquí se guardarán las versiones cerradas

Branches (ramas) : en las ramas se desarrollan versiones que luego se van a publicar. Es una copia del trunk (tronco), de un proyecto , de una carpeta o de un archivo con la intención de modificar sobre ella, para conseguir un producto final diferente y alternativo al original. Es la ramificación del código , es decir, modificaciones de versiones cerradas.

Diagrama en el que se muestra el ciclo de vida de subversión



Descripción del diagrama:

A: Partimos del desarrollo inicial, en el trunk

B: Se crea una rama porque hay que añadir una nueva funcionalidad.

C: Mientras tanto se ha corregido un bug en el tronco principal (C). Una vez que está lista la rama se fusiona en el trunk . Cuando esté lista esta versión libre de bugs y con nueva funcionalidad se crea la primera versión disponible para el público (tag 1.0)

D: Una vez que ha salido la primera versión, se han detectado nuevos bugs, y se necesita añadir nuevas funcionalidades. Entonces se crea una rama para desarrollar una nueva funcionalidad.

E: Se ha creado otra rama porque se necesitan otras funcionalidades.

F: Se realiza la fusión de las dos ramas, primero se realiza una copia de una de ellas y luego se fusiona con la otra.

G: Se incorporan las nuevas funcionalidades al tronco principal. Una vez que está lista esta revisión se crea una nueva versión para el público (tag 1.1)

H: Se han detectado nuevos bugs y se necesita añadir nuevas funcionalidades. Se crea una nueva rama para desarrollar una nueva funcionalidad.

I: Se incorporan las nuevas funcionalidades al tronco principal. Se crea una nueva versión para el público (tag 1.2)

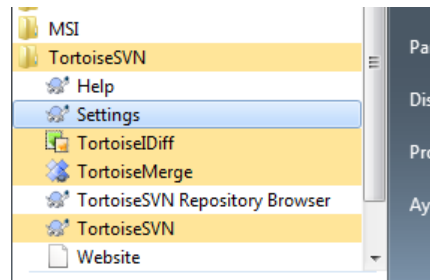
2.2.- Cliente TortoiseSVN

TortoiseSVN es un cliente de código abierto para el sistema de control de versiones subversion

En la web (<http://tortoisesvn.net/downloads.html>) se puede descargar distintas versiones.

Trabajaremos sobre la versión 1.8.12 (<https://osdn.net/projects/tortoisesvn/storage/1.8.12/>)

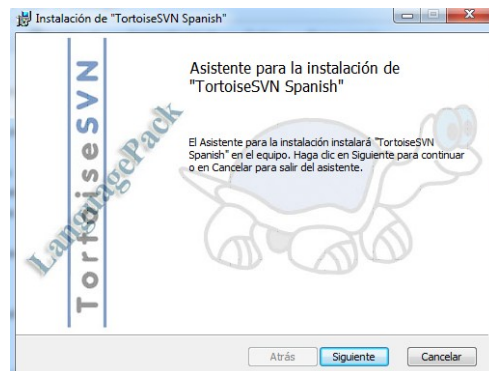
Para la instalación aceptaremos todos los términos de la licencia e indicamos la carpeta donde se va a instalar.



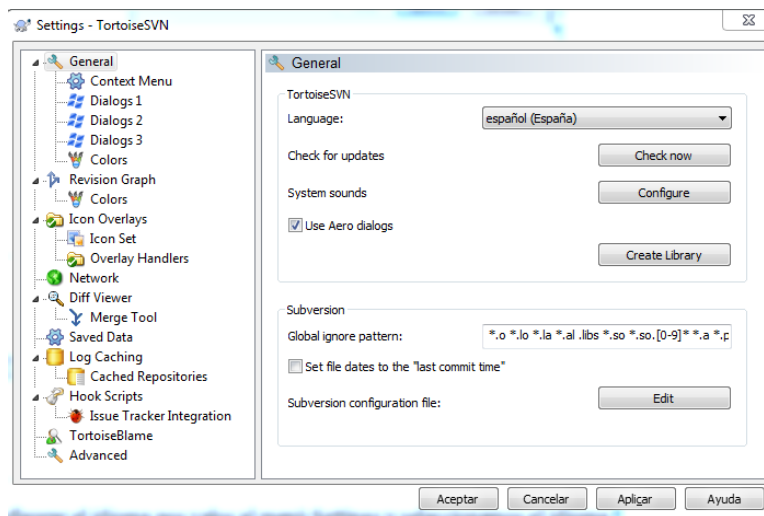
Una vez instalado, ejecutamos el paquete del idioma.

<https://osdn.net/projects/tortoisesvn/storage/1.8.12/Language%20Packs/>

https://osdn.net/projects/tortoisesvn/storage/1.8.12/Language%20Packs/LanguagePack_1.8.12.26645-x64-es.msi/

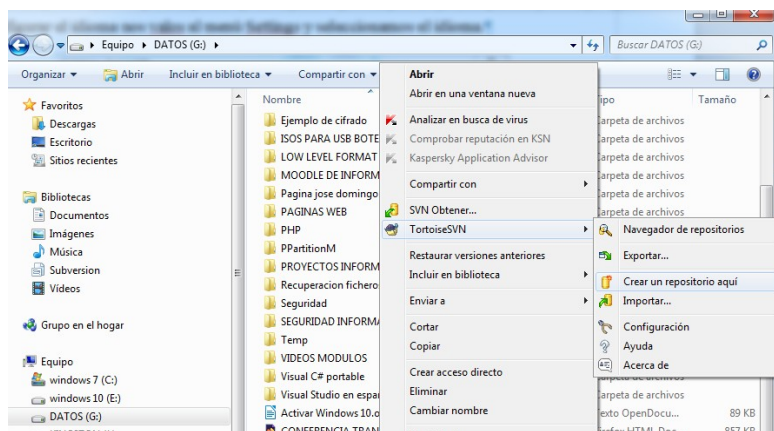


Para configurar el idioma nos vamos al menú Settings y seleccionamos el idioma.

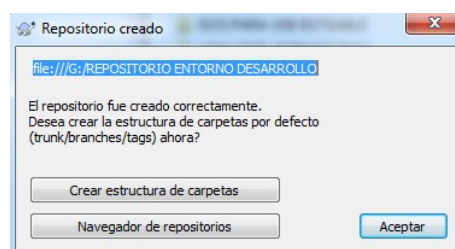


2.2.1.- Creación de un repositorio privado en el PC y controlar los archivos y documentos almacenados en ese repositorio.

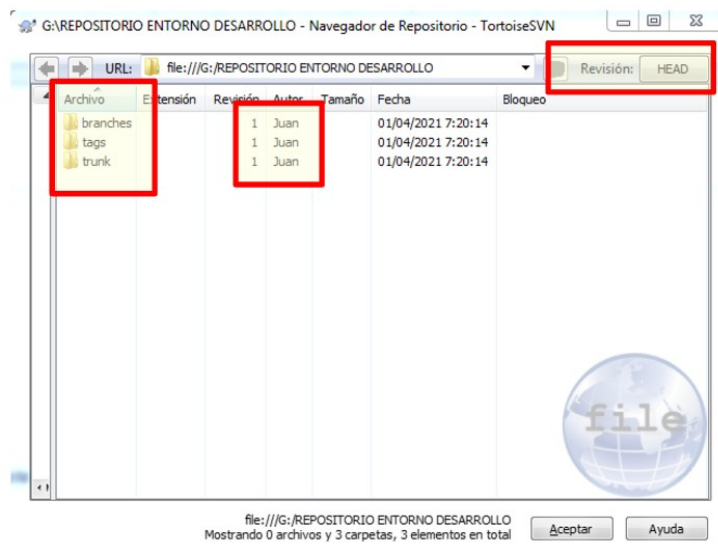
Para crear el repositorio se crea una carpeta en el disco duro , se selecciona y desde el menú contextual (Selección de la carpeta y botón derecho) se elige TortoiseSVN, se accede a la opción “crear un repositorio aquí”.



Una vez creado el repositorio, procedemos a crear la estructura de carpetas (branches, tags, trunks)



Posteriormente hacemos clic en el botón “Navegador de repositorio “ para que se muestre el navegador.



Observaremos la estructura de archivos creada y la revisión inicial que es la 1 (creación del repositorio), esta revisión se incrementará automáticamente cada vez que se hace un cambio en el repositorio.

También podremos acceder a las diferentes revisiones pulsando el botón HEAD que aparece en la parte superior derecha de la ventana

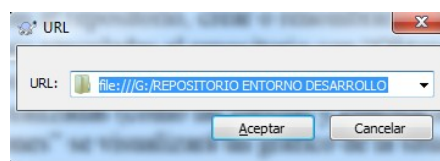
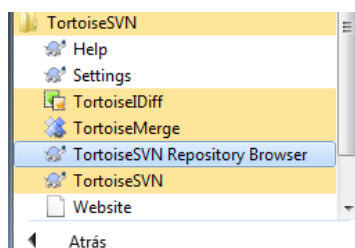
Se podrán crear todos los repositorios que se deseen y desde el navegador se podrán administrar. La estructura de carpetas **trunk-tags-branches**, puede variar dependiendo de la organización que se desee . Se puede organizar un repositorio que contiene muchos proyectos por ramas , o por proyectos.

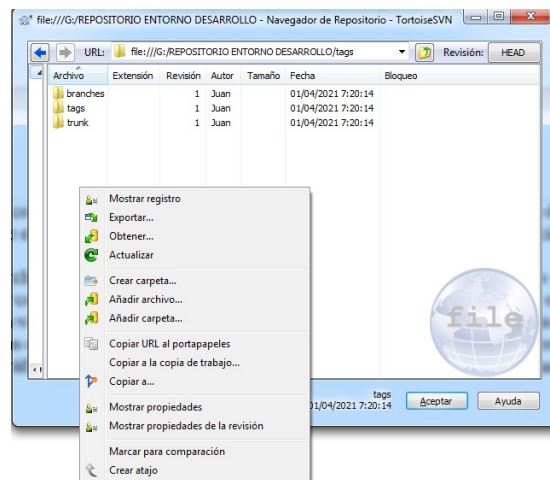
2.2.2.- Operaciones con Tortoise

Para probar la herramienta realiza una copia de varios archivos y carpetas de tu disco para trabajar con ellos.

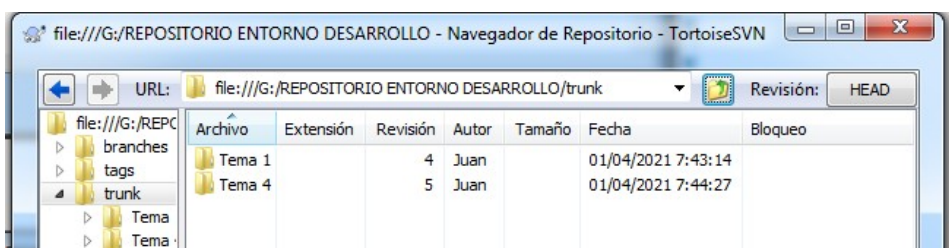
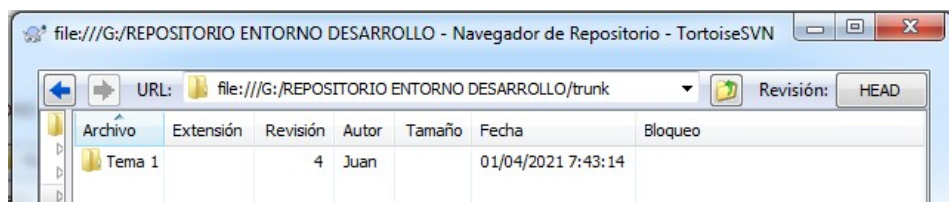
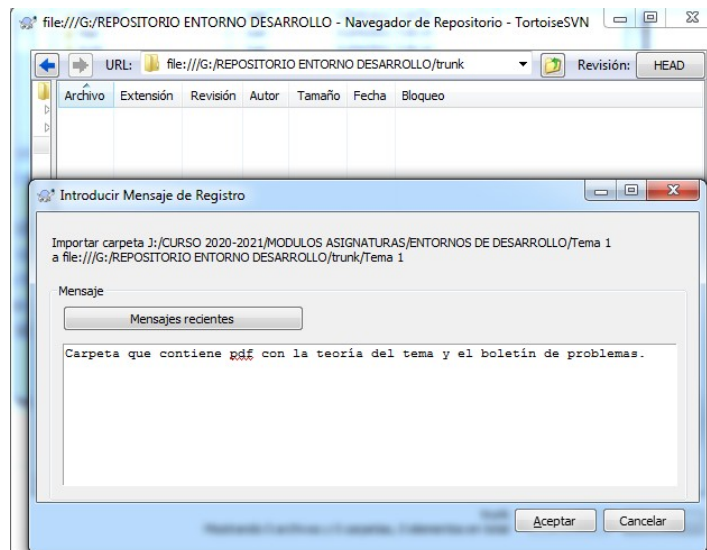
Podemos trabajar con TortoiseSVN dentro y fuera del navegador de repositorio. Dentro del navegador y desde el menú contextual de TortoiseSVN podremos seleccionar las operaciones a realizar.

Podremos subir archivos y carpetas al repositorio, crear o renombrar o eliminar carpetas y archivos, podremos realizar copias de trabajo vinculadas al repositorio con “Obtener (checkout)” o realizar simples copias sin vincular al repositorio con Exportar. Con “Mostrar el registro de revisiones” se mostrarán todas las operaciones realizadas (como las subidas y eliminaciones de archivos y carpetas), con “Gráfico de revisiones” se visualizará un gráfico de la situación seleccionada.





Ejemplo: Subamos una carpeta “Tema 1” dentro de trunk (tronco)

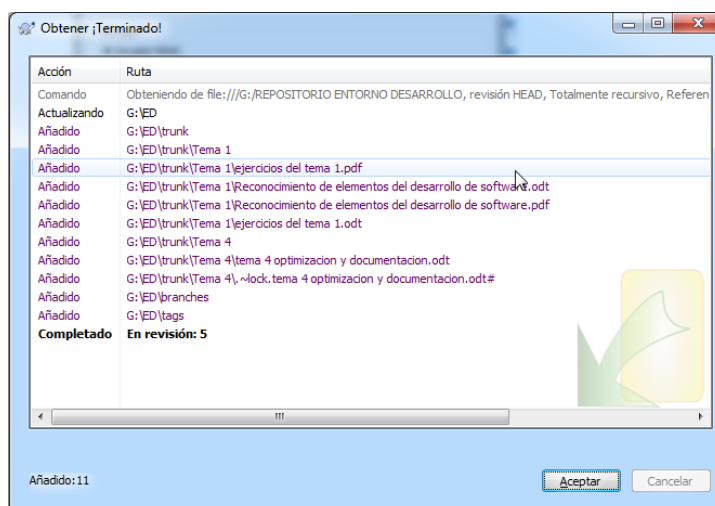
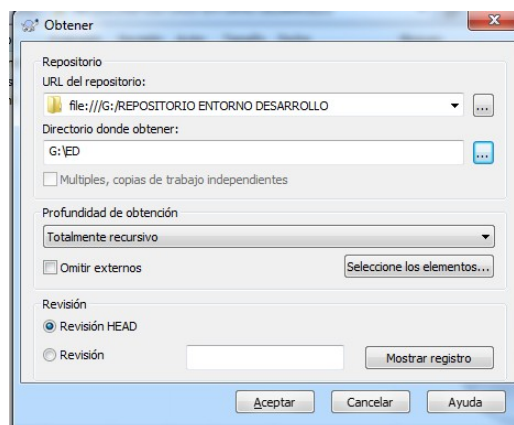
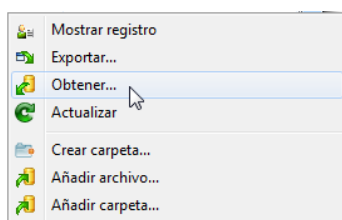


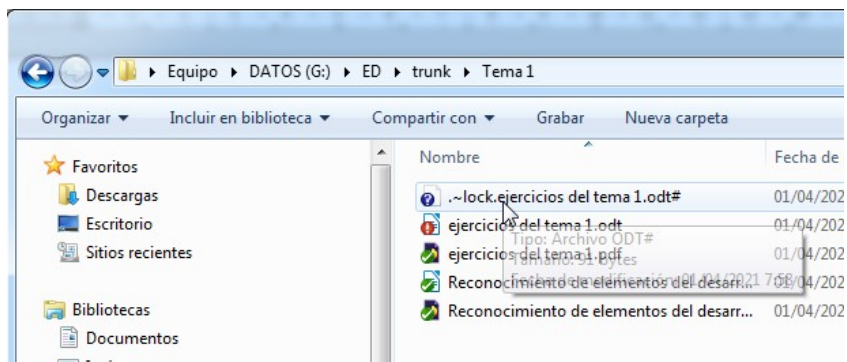
EJERCICIO 1

Desde el navegador de repositorio , selecciona la carpeta “trunk” y sube una carpeta y dos archivos, añade un mensaje a las subidas. Observa las revisiones que se han ido generando. Accede a cada una de ellas desde el botón HEAD , muestra el registro de revisiones, marca cada una de las revisiones y observa en la pantalla el mensaje añadido para esa revisión y los archivos implicados. Visualiza el gráfico de revisiones.

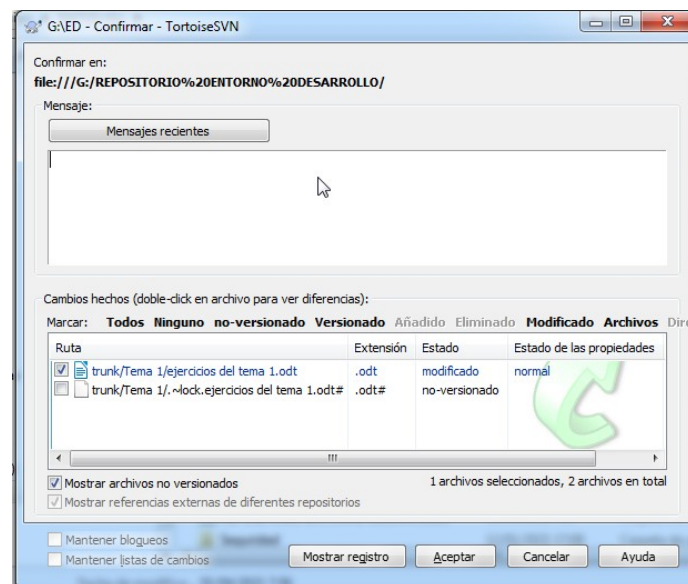
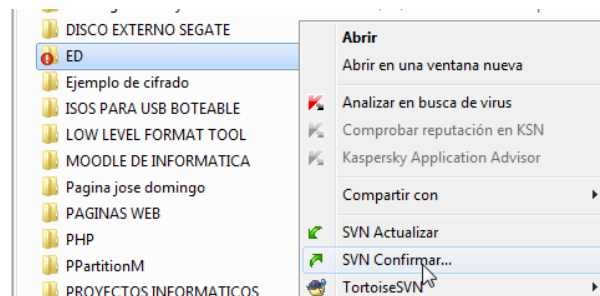
Nota: Realiza captura de pantallas de todo el proceso.

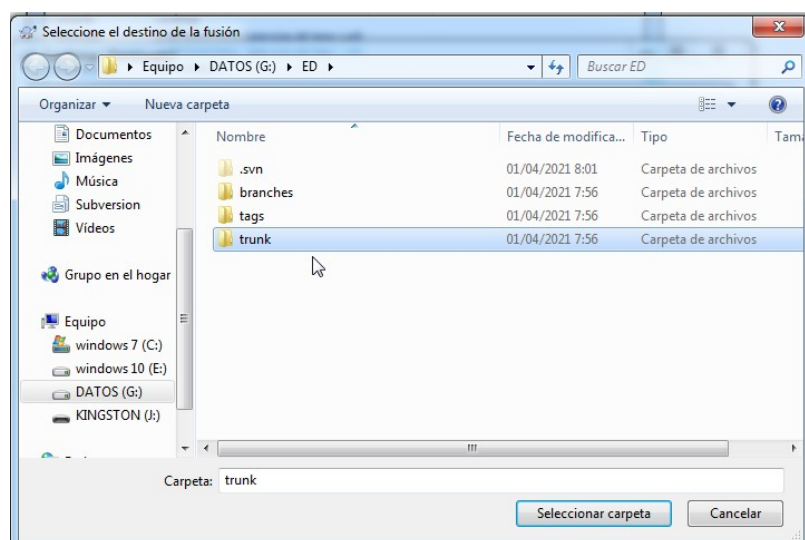
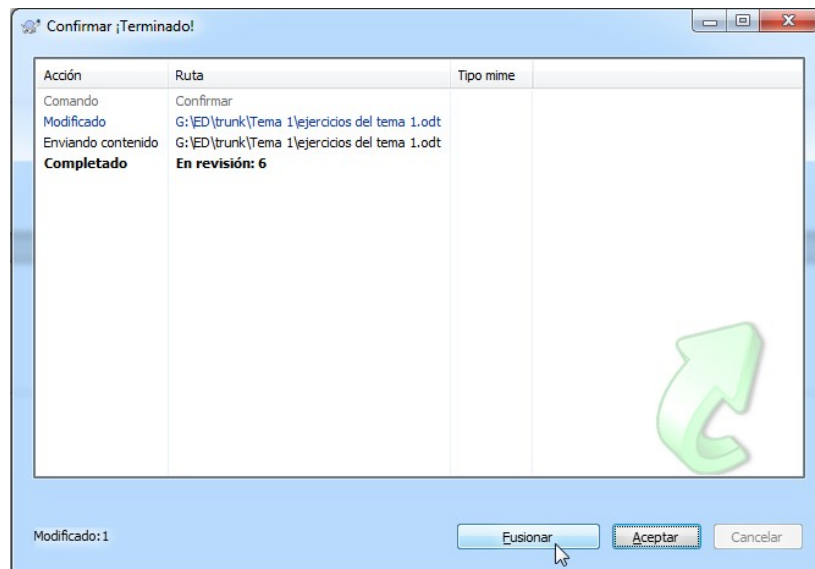
El proceso de trabajo fuera del navegador de repositorios consistirá en hacer CHECKOUT (obtener) para obtener una copia de trabajo en una carpeta local, realizar cambios en esa carpeta local , y confirmar los cambios COMMIT



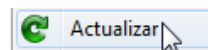


Una vez modificado el fichero , tenemos de confirmar los cambios para subirlos al repositorio.



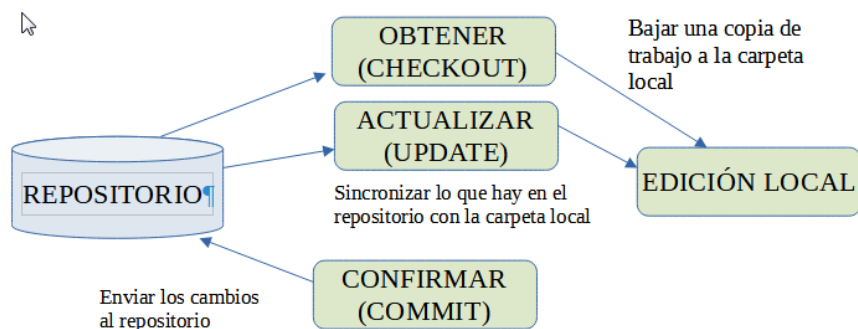


NOTA: Antes de realizar cambios es aconsejable actualizar



con lo que hay en el repositorio , para asegurarse de trabajar con la última revisión . Luego se realizan los cambios, y una vez finalizados se confirman.

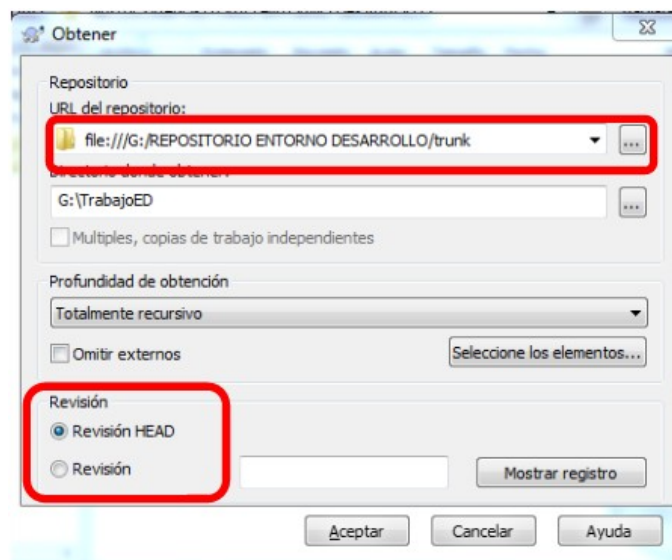
Operaciones básicas de control de versiones

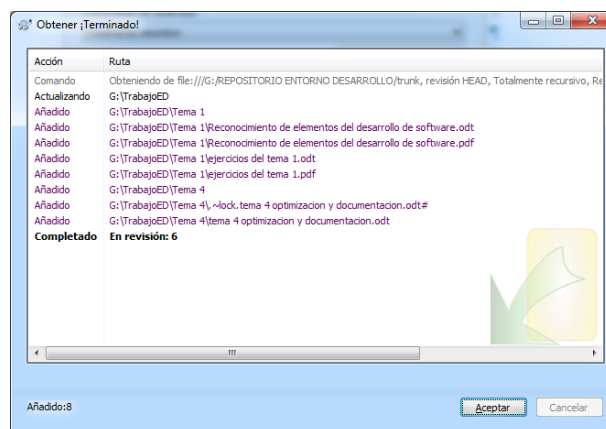


EJERCICIO 2

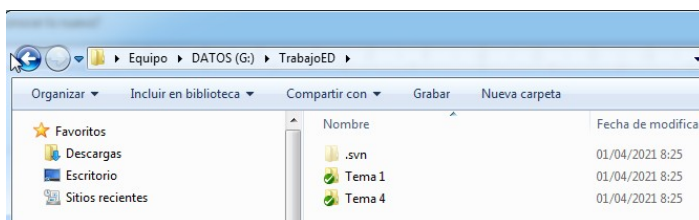
Desde el sistema de archivos, crear una carpeta de trabajo para operar con el repositorio, por ejemplo TrabajoED

Desde el menú contextual hacer CHECKOUT (Obtener) de los archivos y carpetas subidos en la actividad anterior. En la ventana que se muestra se elige la URL del repositorio, también se puede obtener una revisión concreta. Elegir la carpeta “trunk” para descargarnos su contenido.





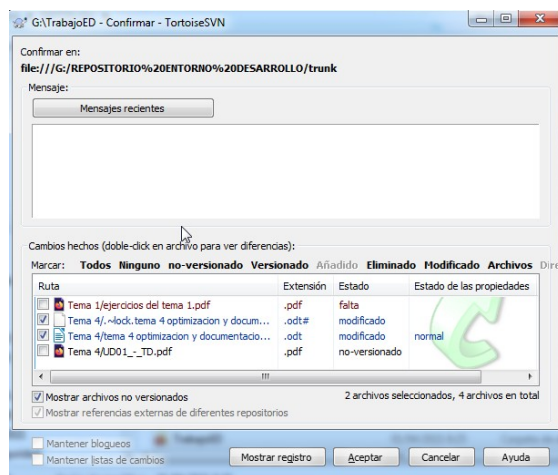
Podemos observar que se crea una carpeta oculta “.svn”

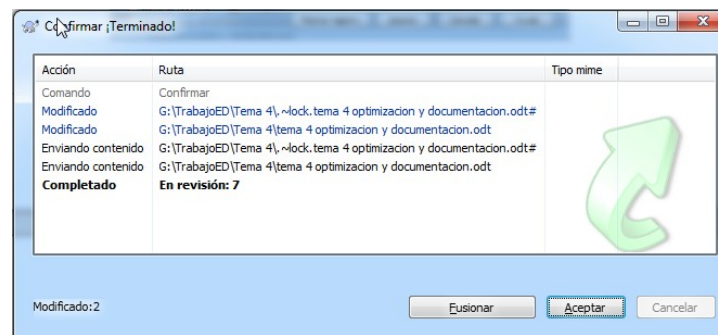


Si no la ves, activa la opción de ver carpetas ocultas.

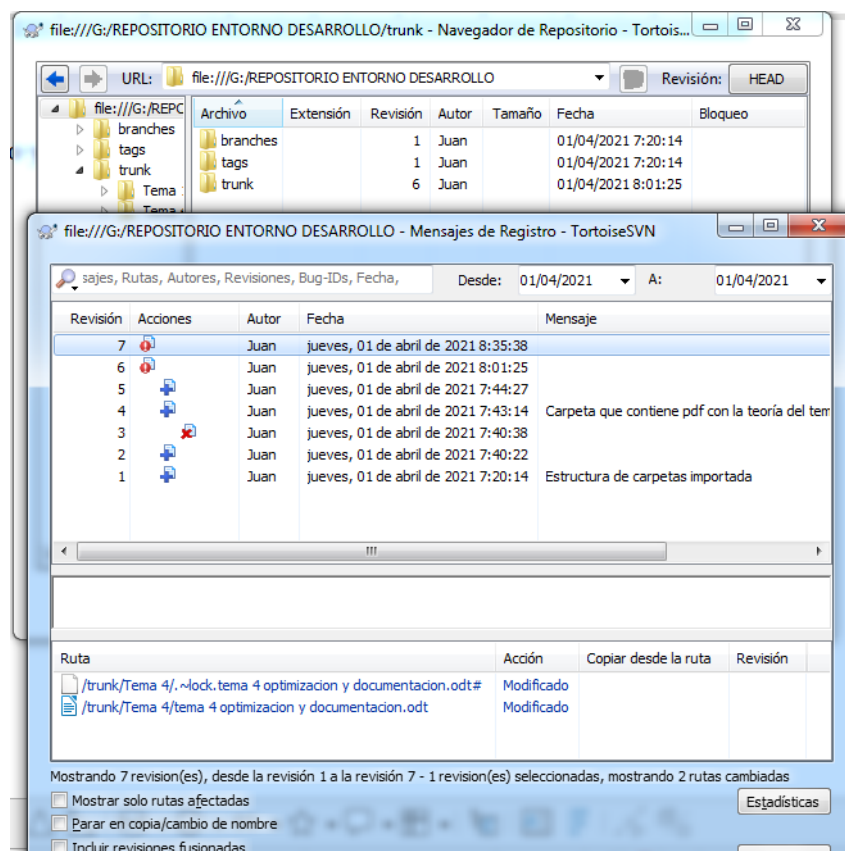
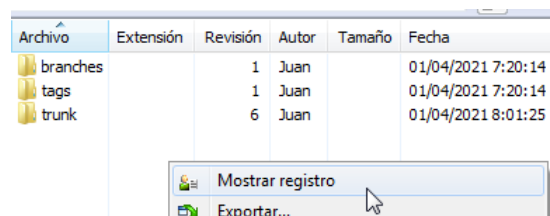
Realiza cambios sobre algún documento, borra alguno de los documentos, copia un nuevo documento a esa carpeta y confirma los cambios , botón derecho del ratón y “Confirmar”

En la ventana de confirmar añade un mensaje y marca los cambios. Observar el estado de los cambios, aparece **modificado** en el documento modificado, **falta** en el documento que se ha borrado y no **versionado** en el documento que hemos añadido . Pulsa aceptar , y a continuación se muestra la ventana con información de los cambios , la nueva revisión y si se desea “Fusionar” los cambios o “Aceptar” , se pulsa Aceptar.





Muestra el registro y observa las modificaciones



Abre el navegador de repositorio y observa las revisiones creadas, accede al registro de revisiones. Observa que el archivo borrado permanece en las revisiones iniciales, muy útil si se desea recuperar de nuevo.

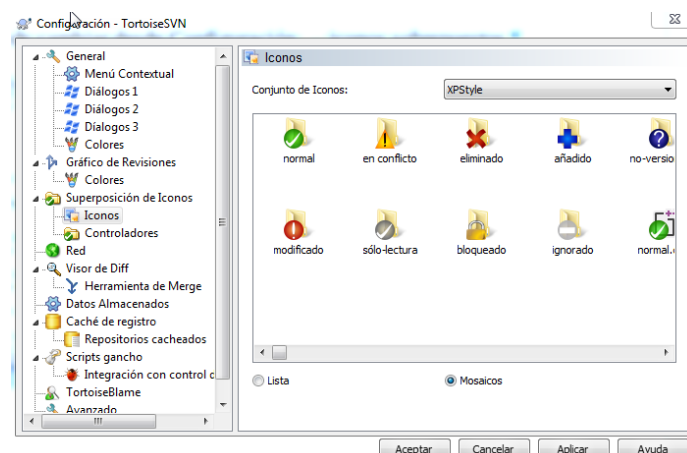
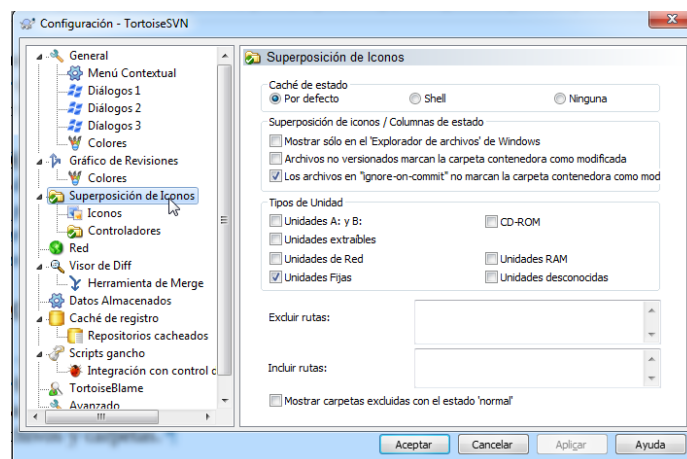
OBSERVACIONES:

Se debe de actualizar (Update) nuestro directorio de trabajo siempre antes de hacer cualquier cambio, para asegurarnos de que estamos modificando la última versión del repositorio. Al actualizar se harán todos los cambios necesarios (añadir / borrar / modificar ficheros y directorios) para que los datos sean idénticos a los del repositorio (en ese momento).








2.2.2.1.- Obteniendo información del estado del repositorio

Mientras se trabaja con la copia de trabajo es necesario saber qué archivos se han cambiado, añadido, borrado, renombrado, o incluso qué archivos han sido cambiados y confirmados por los demás. Para ello aparecerán unos iconos superpuestos que dan información del estado en subversión de los archivos y carpetas.

Esta configuración se puede cambiar desde Configuración → iconos superpuestos.



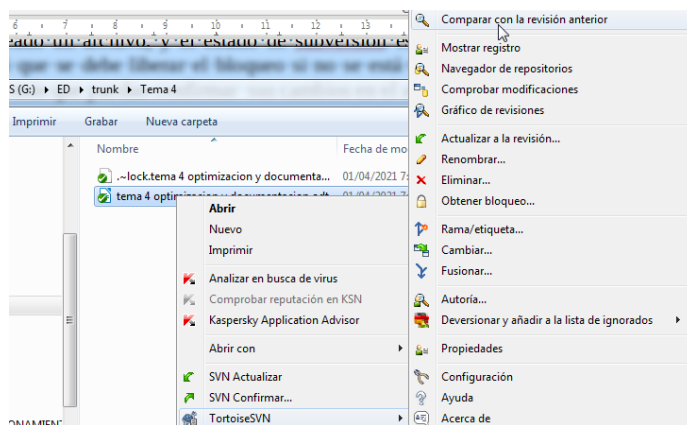
Iconos más comunes:

	Una copia de trabajo recién obtenida y sincronizada indica estado de subversión normal
	Al editar un archivo , el estado cambia a modificado y el icono cambia entonces a una marca de exclamación roja
	Este icono muestra que algunos archivos o carpetas dentro de la carpeta actual se han marcado para ser eliminados del control de versiones, o bien que falta un archivo que está bajo el control de versiones dentro de una carpeta
	El signo “ + “ indica que el archivo o carpeta está programado para ser añadido al control de versiones , se le ha indicado que se añadirá desde el menú “Añadir”.
	Si durante una actualización ocurre un conflicto , el icono cambia a un signo de exclamación amarillo
	Si se ha bloqueado un archivo, y el estado de subversion es normal , este icono recordará que se debe liberar el bloqueo si no se está utilizando para permitir a los demás que puedan confirmar sus cambios en el archivo
	Este icono muestra los archivos y carpeta que no están bajo el control de versiones pero tampoco han sido ignorados. Este icono superpuesto es opcional. Suele aparecer cuando hay conflictos, marca con el icono a las versiones del archivo.

Ejercicio 3

Observa los iconos superpuestos asociados a los archivos y carpetas . Prueba a copiar nuevos archivos en la copia de trabajo y programarlos para subirlos al repositorio con el menú “Añadir”.

Prueba a realizar modificaciones sobre un archivo de la copia de trabajo, y sin validarlo comprueba el archivo con la versión anterior del repositorio, desde el menú , sobre el archivo y elige “Comparar con la revisión anterior” . Observa las diferencias entre uno y otro archivo.



2.2.3- Resolver conflictos

Una vez que se hayan realizado los cambios en la carpeta de trabajo se subirán al repositorio con la opción “Confirmar” . Es conveniente añadir comentarios para recordar el cambio en si.

Si al confirmar se producen **conflictos** es porque alguien realizó cambios en el mismo fichero y lo subió al repositorio con antelación . **Se nos avisará de que la operación ha fallado y no subirá los cambios.**

En este caso , se aconseja que se actualice a la copia del repositorio o que se cancele la operación. Si optamos por la opción de actualizar, se creará una copia de cada archivo en la carpeta de trabajo , y el archivo pasará al estado “**En conflicto**”. Para resolver el conflicto seleccionamos la opción “**Resolver**” del menú.

Ejercicio 4

Simular una situación de conflicto. Crear una carpeta de trabajo nueva (TrabajoED2) y haz Checkout (obtener) de los archivos y carpetas del repositorio que se subieron en las actividades anteriores (antes confirma las dos carpetas de trabajo TrabajoED y TrabajoED2 tenga la misma revisión, confirma los cambios de la carpeta de trabajo y actualízala “commit y update” antes de bajar los datos del repositorio a la nueva carpeta TrabajoED2).

Realiza cambios en un archivo de una de las carpetas de trabajo (TrabajoED), y confírmalos en el repositorio. Realiza cambios en el mismo archivo pero en la otra copia de trabajo (TrabajoED2). Al validar este cambio, nos debe mostrar una ventana indicando que la confirmación falló, que el archivo a modificar está desactualizado porque no coinciden las revisiones, e indica que se debe de actualizar. Al pulsar “Aceptar” pedirá : que se actualice y se vuelve a intentar la subida; o que no se actualice la copia y se cancele. Se pulsa la opción de actualizar y se muestra una ventana en la que se indica que se ha actualizado a la versión del repositorio, pero que el archivo está en conflicto.

Nota: Con los archivos en conflicto la validación no podrá ser realizada , así pues se pulsa “Cancelar” para salir de la validación , el conflicto se debe de resolver de forma manual.

Observa que en la carpeta de trabajo (TrabajoED2) se han creado dos archivos con el icono



Esto indica que hay un conflicto entre archivos, observa que estos archivos se etiquetan añadiendo como extensión el número de revisión (ejemplo .r13 y r14)

Como resolver el conflicto:

Para resolver el conflicto, dentro de la carpeta con el conflicto abrimos el menú (botón derecho) y elegimos resolver , se selecciona el archivo y se pulsa “Aceptar”. El archivo se quedará en estado modificado pendiente de la confirmación al repositorio. Ahora se añadirán los cambios y se validaría. **Se perderían los cambios realizados por el primero que validó.**

Realiza estas tarea con captura de pantalla de cada uno de los pasos.

2.3.- Creación de ramas y etiquetas

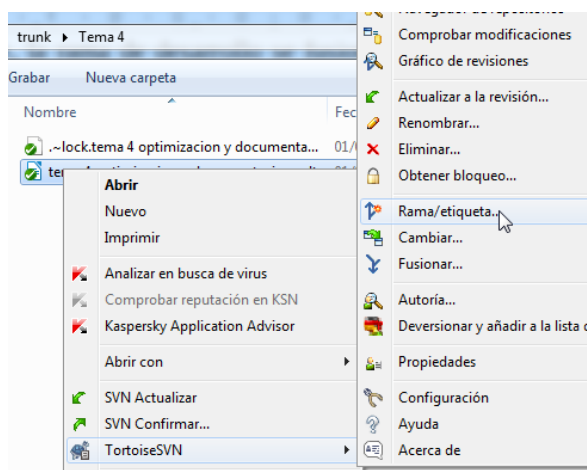
Los sistemas de control de versiones no da la posibilidad de crear líneas separadas de desarrollo para realizar cambios , corregir errores o añadir funcionalidades, **esto se conoce como crear una rama**. Cuando la nueva característica es estable, la rama de desarrollo se fusiona de nuevo en la rama principal (trunk o tronco).

Otra características es la posibilidad de marcar revisiones particulares, para que se pueda en cualquier momento recrear un cierto entorno o compilación; a este proceso se le conoce como “etiquetar”.

Cuando se crea una rama o una etiqueta , no se hacen copias completas del repositorio, sino que se crean vínculos internos, apuntando a una revisión y árbol específicos. Como resultado, las ramas y las etiquetas son muy rápidas de crear y apenas ocupan espacio en el repositorio.

2.3.1.- Crear etiquetas

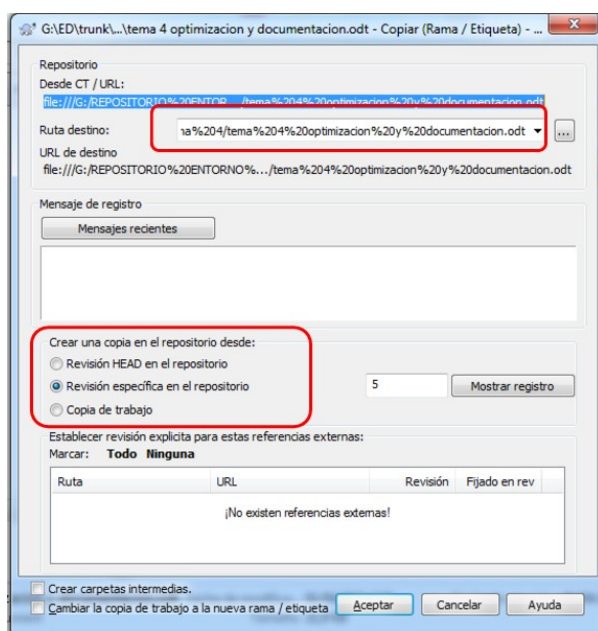
No posicionamos en la carpeta o archivos del repositorio local de trabajo que queramos etiquetar , hacemos clic botón derecho del ratón . A continuación seleccionamos el elemento (Rama/etiqueta) del menú



Hay que indicar el origen y el destino de la etiqueta a crear, y sobre qué elementos se crea la etiqueta o copia.

Antes de crear una rama o una etiqueta hay que crear en el repositorio (desde el navegador del repositorio) una carpeta en branches o en tags (en función de si se crea una rama o una etiqueta) para guardar los archivos y que no cuelguen directamente de branches o tags.

Al crear la etiqueta aparece la siguiente ventana en la que se indica lo siguiente:



- **CT/URL:** Indica la carpeta que se desea etiquetar del repositorio local.
- **Ruta Destino:** Se indica la carpeta destino de la etiqueta. Para ello se selecciona la carpeta creada dentro de tags y se añade un nombre de versión , por ejemplo “Version2”, que creará una carpeta nueva donde se guardarán todos los archivos a etiquetar. En el navegador del repositorio se debe mostrar la carpeta creada dentro de tags, y dentro de la “Versión2” aparecerán los archivos etiquetados .

Para elegir el origen de la copia se dispone de tres opciones:

- **Revisión HEAD en el repositorio:** la nueva etiqueta / rama se copia directamente en el repositorio desde la revisión HEAD del repositorio.
- **Revisión específica en el repositorio:** la nueva etiqueta/rama se copia directamente en el repositorio, eligiendo una versión anterior. Con el botón “Mostrar registro” se pueden ver las revisiones y seleccionarlas . No se transfiere datos desde la copia local de trabajo.
- **Copia de trabajo:** la nueva etiqueta / rama es una copia idéntica de la copia local de trabajo. Si se han cambiado algunos archivos a una revisión anterior en la copia de trabajo, o si se han hecho cambios locales, esto es exactamente lo que irá a la copia.

2.3.2.- Crear ramas

Es similar a la creación de etiquetas . En primer lugar creamos las carpetas de destino dentro de branches desde el “Navegador del repositorio”, poniendo el mismo nombre de la carpeta a enramar. Aunque esta aplicación no hace distinción entre etiquetas y ramas.

La etiquetas se usan generalmente para crear una copia estática de un proyecto en una etapa concreta. Trabajar en una revisión etiquetada (dentro de tag) no es una buena idea . Sin embargo, si necesitamos hacer cambios en una versión etiquetada , la forma de hacerlo es creando primero una nueva rama desde la etiqueta , hacer los cambios en la rama , y luego crear una nueva etiqueta para esta rama. Recuerda que si se modifica una copia de trabajo creada desde una rama y se confirma los cambios, los cambios irán a la rama y no al tronco.

Ejercicio 5

Crea una rama con una carpeta que tengas en el repositorio . Recuerda que antes de crear la rama hay que crear la carpeta dentro de branches con el mismo nombre desde el navegador de repositorio.

Al crear la rama , en “Ruta / Destino “ indica la carpeta creada del repositorio y añade el nombre de la rama , (llámalo RAMA1) . Añade un comentario .

Realiza captura de todo este proceso y de la vista del repositorio una vez creada la rama.

Crea una copia de trabajo en el disco de la rama para realizar modificaciones (menú Obtener), añade algún archivo y modifica algún documento. Confirma los cambios en la rama (menú Confirmar) y añade un comentario.

Realiza capturas de todo el proceso.

2.3.3.- Fusionar ramas

Los cambios se realizan en las ramas, una vez realizados los cambios, estos los fusionamos con las copias de trabajo, y desde las copias de trabajo asociadas al tronco se confirman los cambios a trunk.

La fusión siempre se realiza sobre copias de trabajo.

Para fusionar los cambios realizados en una rama creada de una carpeta de trunk, en la copia de trabajo asociada a trunk, se abre la carpeta sobre la que se creó la rama y dentro de esa carpeta que se ha modificado en la rama , se abre el menú y se elige “Fusionar”. Nos aparecerá dos opciones de fusión:

- 1) ***Fusionar un rango de revisiones.*** Este método se elige cuando se ha hecho una o más revisiones en una rama y se desea recoger los cambios a una rama diferente o a la carpeta de trabajo . En el campo “URL desde “ , se escribirá la URL completa de la carpeta de la rama que contiene los cambios que se desea cargar en la copia de trabajo. También se podrá elegir un rango de revisiones.

- 2) **Fusionar dos árboles diferentes.** Este método se usa cuando se desea fusionar las diferencias de dos ramas distintas en la copia de trabajo. Lo que se le indica a la aplicación es que haga los cambios necesarios para ir desde la *URL y revisión inicial del rango a fusionar*, por ejemplo la versión actual (HEAD) del trunk, hasta la URL y revisión del fin del rango a fusionar, por ejemplo los cambios de una rama , y que aplique esos cambios a la copia de trabajo (que es la asociada a trunk). El resultado final es hacer que la copia del tronco sea igual a la de la rama.

Unas opciones del asistente de fusión son:

- Profundidad de fusión: indica los niveles que debe bajar la fusión en la copia de trabajo
- Forzar la fusión: se usa para evitar conflictos de árbol en los que un borrado entrante afecta a archivos que, o están modificados localmente o ni siquiera están versionados.

NOTA: Si al realizar la fusión aparecen muchos errores o conflictos es posible volver a la situación anterior seleccionando “Revertir” del menú.

Ejercicio 6

Antes de fusionar actualiza la carpeta de trabajo con la versión del repositorio. Se supone que las revisiones de trunk son anteriores a las de la rama.

Fusionar la rama creada en el ejercicio 5 con la opción “Fusionar un rango de revisiones” . Sigue el asistente y antes de aceptar la fusión pulsa el botón “Probar fusión” para comprobar posibles conflictos . Si ocurren conflictos en algún archivo marca la opción de posponer la resolución para los archivos en conflictos.

Confirma los cambios en el repositorio añadiendo comentarios (incluye tu nombre en el comentario).

Realiza más cambios sobre la rama, añadiendo un archivo, modificando otro y eliminando otro.

Confirma los cambios de la rama en el repositorio , añadiendo un comentario (incluye tu nombre en el comentario).

Realiza la fusión con la opción “Fusionar dos árboles diferentes”, desde la revisión que se encuentra en trunk hasta la revisión de la rama.

Confirma los cambios añadiendo comentarios (incluye tu nombre en el comentario).

Nota: Realiza capturas de todo el proceso