TortoiseSVN

Un cliente de Subversion para Windows

Versión 1.6.16

Stefan Küng
Lübbe Onken
Simon Large

TortoiseSVN: Un cliente de Subversion para Windows: Versión 1.6.16

por Stefan Küng, Lübbe Onken, y Simon Large
Traducción: Fernando P. Najera Cano (yo@FernandoNajera.com)

publicado 2011/01/21 21:21:17 (r20750)

Tabla de contenidos

Lista de figuras

Lista de tablas

Prefacio

- ¿Trabaja en equipo?
- ¿Alguna vez le ha ocurrido que estaba trabajando en un fichero, y alguien más también estaba trabajando en ese mismo fichero al mismo tiempo? ¿Perdió sus cambios en ese fichero por ese motivo?
- ¿Alguna vez ha grabado un fichero, y luego deseó deshacer los cambios que había hecho? ¿Alguna vez ha querido ver cómo estaba un fichero hace tiempo?
- ¿Alguna vez ha encontrado un error en su proyecto y ha querido saber cuándo se introdujo ese error en sus ficheros?

Si ha respondido “sí” a alguna de las preguntas anteriores, ¡entonces TortoiseSVN está hecho para usted! Siga leyendo para saber cómo puede TortoiseSVN ayudarle en su trabajo. No es tan dificil.

1. Audiencia

Este libro está escrito para usuarios informáticos que quieren usar Subversion para manejar sus datos, pero no están cómodos usando el cliente de línea de comandos para hacerlo. Dado que TortoiseSVN es una extensión del shell de Windows, se asume que el usuario está familiarizado con el Explorador de windows y sabe cómo usarlo.

2. Guía de Lectura

Este Prefacio le ofrece una breve explicación sobre el proyecto TortoiseSVN, la comunidad de gente que trabaja en él, y las condiciones de licencia para utilizarlo y distribuirlo.

La Capítulo 1, *Introducción* expone qué es TortoiseSVN, lo que hace, de dónde viene y las bases para instalarlo en su PC.

En los Capítulo 2, *Conceptos básicos de control de versiones* le ofrecemos una breve introducción al sistema de control de revisiones *Subversion* que es la base de TortoiseSVN. Está prestado de la documentación del proyecto Subversion y explica las diferentes formas de control de versiones y cómo funciona Subversion.

El capítulo sobre el Capítulo 3, *El repositorio* explica cómo se prepara un repositorio local, algo útil para probar Subversion y TortoiseSVN utilizando un único PC. También explica algo sobre la administración de repositorios, que también es relevante para los repositorios que se encuentran en un servidor. También hay una sección aquí sobre cómo preparar un servidor, si lo necesita.

La Capítulo 4, *Guía de uso diario* es la sección más importante, ya que le explica todas las características principales de TortoiseSVN y cómo utilizarlas. Tiene la forma de un tutorial, empezando con obtener una copia de trabajo, modificarla, confirmar los cambios, etc. Luego avanza a temas más avanzados.

Capítulo 5, *El programa SubWCRev* es un programa adicional que se incluye con TortoiseSVN, y que puede extraer información de su copia de trabajo y escribirla en un fichero. Esto es útil para incluir información de compilación en sus proyectos.

La sección Apéndice B, *¿Cómo...?* responde algunas preguntas frecuentes sobre la realización de tareas que no están cubiertas explícitamente en ninguna otra parte.

La sección Apéndice D, *Automatizando TortoiseSVN* le muestra cómo puede invocar los diálogos GUI de TortoiseSVN desde la línea de comandos. Esto es útil para scripts donde se necesite la interacción del usuario.

La Apéndice E, *Referencia cruzada del interface de línea de comandos* le muestra una correlación entre los comandos de TortoiseSVN y sus equivalentes en el cliente de línea de comandos de Subversion `svn.exe`.

3. ¡TortoiseSVN es gratis!

TortoiseSVN es gratis. No tiene que pagar por él, y puede usarlo para lo que quiera. Está desarrollado bajo la Licencia Pública General GNU (GPL).

TortoiseSVN es un proyecto de Código Abierto. Eso significa que tiene acceso completo de lectura al código fuente de este programa. Puede verlo en este enlace *http://code.google.com/p/tortoisesvn/source/browse/*. Se le pedirá un usuario y contraseña. El usuario es `guest`, y la contraseña debe dejarse en blanco. La versión más reciente (donde estamos trabajando) se encuentra bajo `/trunk/`, y las versiones ya lanzadas están bajo `/tags/`.

4. Comunidad

Tanto TortoiseSVN como Subversion se desarrollan por una comunidad de gente que trabaja en estos proyectos. Provienen de diferentes países alrededor de todo el mundo y trabajan juntos para crear programas estupendos.

5. Reconocimientos

Tim Kemp
: por fundar el proyecto TortoiseSVN

Stefan Küng
: por el duro trabajo de llevar a TortoiseSVN a lo que es hoy

Lübbe Onken
: por los bonitos iconos, el logo, la caza de errores, por traducir y administrar traducciones

Simon Large
: por ayudar con la documentación y en la caza de bugs

El libro de Subversion
: por la gran introducción a Subversion y su capítulo 2 que hemos copiado aquí

El proyecto Tigris Style
: por algunos de los estilos que están siendo reutilizados en esta documentación

Nuestros colaboradores
: por los parches, informes de errores y nuevas ideas, y por ayudar a otros respondiendo preguntas de nuestra lista de correo.

Nuestros donantes
: por la cantidad de horas de entretenimiento con la música que nos enviaron

6. Terminología usada en este documento

Para hacer más facil la lectura de la documentación, los nombres de todas las pantallas y menús de TortoiseSVN están remarcados en un tipo de letra diferente. Por ejemplo, el Diálogo de Registro.

Las opciones de menú se indican con una flecha. TortoiseSVN → Mostrar Registro significa: seleccione *Mostrar Registro* desde el menú contextual *TortoiseSVN*.

Donde aparezca un menú contextual local dentro de uno de los diálogos de TortoiseSVN, se mostrará así: Menú Contextual → Grabar como ...

Los botones del interfaz de usuario se indican como este: Pulse OK para continuar.

Las acciones del usuario se indican en negrita. Alt+A: pulse la tecla **Alt** en su teclado, y mientras la mantiene pulsada, pulse también la tecla **A**. Arrastre-con-botón-derecho: pulse el botón derecho del ratón, y mientras lo mantiene pulsado, *arrastre* los ítems a su nuevo destino.

La salida del sistema y la entrada por teclado se indica con una fuente también `diferente`.

Importante

Las notas importantes están marcadas con un icono.

Sugerencia

Trucos que le facilitan la vida.

Atención

Lugares donde debe tener cuidado con lo que hace.

Aviso

Donde hay que tener un cuidado extremo, porque puede ocurrir corrupción de datos u otras cosas horribles si se ignoran estas advertencias.

Capítulo 1. Introducción

El control de versiones es el arte de manejar cambios en la información. Ha sido desde siempre una herramienta crítica para los programadores, quienes típicamente emplean su tiempo haciendo pequeños cambios al software y luego deshaciendo o comprobando esos cambios al día siguiente. Imagine un equipo de estos programadores trabajando concurrentemente - ¡y quizás también simultáneamente en los mismos ficheros! - y podrá ver por qué se necesita un buen sistema para *manejar el caos potencial*.

1.1. ¿Qué es TortoiseSVN?

TortoiseSVN es un cliente gratuito de código abierto para el sistema de control de versiones *Subversion*. Esto es, TortoiseSVN maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un *repositorio* central. El repositorio es prácticamente lo mismo que un servidor de ficheros ordinario, salvo que recuerda todos los cambios que se hayan hecho a sus ficheros y directorios. Esto permite que pueda recuperar versiones antiguas de sus ficheros y examinar la historia de cuándo y cómo cambiaron sus datos, y quién hizo el cambio. Esta es la razón por la que mucha gente piensa que Subversion, y los sistemas de control de versiones en general, son una especie de “máquinas del tiempo”.

Algunos sistemas de control de versiones también son sistemas de manejo de configuración del software (SCM). Estos sistemas están diseñados específicamente para manejar árboles de código fuente, y tienen muchas características que son específicas para el desarrollo de software - tales como el entendimiento nativo de los lenguajes de programación, o proporcionan herramientas para compilar software. Subversion, sin embargo, no es uno de estos sistemas; es un sistema general que puede ser utilizado para manejar *cualquier* colección de ficheros, incluyendo código fuente.

1.2. Historia de TortoiseSVN

En 2002, Tim Kemp se dio cuenta que Subversion era un sistema de control de versiones muy bueno, pero le faltaba un buen cliente GUI. La idea de tener un cliente de Subversion integrado en el shell de Windows se inspiró por el cliente similar que ya existía para CVS llamado TortoiseCVS.

Tim estudió el código fuente de TortoiseCVS y lo utilizó como base de TortoiseSVN. Entonces inició el proyecto, registró el dominio `tortoisesvn.org` y puso el código fuente en línea. Durante ese tiempo, Stefan Küng estaba buscando un sistema de control de versiones bueno y gratuito, y encontró Subversion y el código fuente de TortoiseSVN. Como TortoiseSVN todavía no estaba listo para usarse, se unió al proyecto y empezó a programar. Pronto reescribió la mayor parte del código existente y empezó a añadir comandos y características, hasta el punto de que no quedó nada del código original.

Según se fue estabilizando Subversion, atrajo más y más usuarios que también empezaron a utilizar TortoiseSVN como su cliente de Subversion. Los usuarios de TortoiseSVN se incrementaron rápidamente (y aún crecen día a día). Entonces Lübbe Onken se ofreció a ayudar con algunos iconos más vistosos y un logo para TortoiseSVN. Y también se encarga de la página web y de administrar las traducciones.

1.3. Características de TortoiseSVN

¿Qué hace de TortoiseSVN tan buen cliente de Subversion? Aquí hay una pequeña lista de sus características.

Integración con el shell de Windows

TortoiseSVN se integra perfectamente en el shell de Windows (por ejemplo, el explorador). Esto significa que puede seguir trabajando con las herramientas que ya conoce. ¡Y que no tiene que cambiar a una aplicación diferente cada vez que necesite las funciones del control de versiones!

Y ni siquiera está obligado a usar el Explorador de Windows. Los menús contextuales de TortoiseSVN también funcionan en otros administradores de archivos, y en el diálogo Fichero/Abrir

que es común a la mayoría de aplicaciones estándar de Windows. Sin embargo, debe tener en cuenta que TortoiseSVN está desarrollado con la mirada puesta en hacerle extensión del Explorador de Windows. Por este motivo, puede que en otras aplicaciones la integración no sea tan completa y que, por ejemplo, los iconos sobreimpresionados en las carpetas no se muestren.

Iconos sobreimpresionados
: El estado de cada carpeta y fichero versionado se indica por pequeños iconos sobreimpresionados. De esta forma, puede ver fácilmente el estado en el que se encuentra su copia de trabajo.

Fácil acceso a los comandos de Subversion
: Todos los comandos de Subversion están disponibles desde el menú contextual del explorador. TortoiseSVN añade su propio submenú allí.

Dado que TortoiseSVN es un cliente de Subversion, también queremos enseñarle algunas de las características del propio Subversion:

Versionado de carpetas
: CVS sólo controla la historia de ficheros individuales, pero Subversion implementa un sistema "virtual" de ficheros versionados que sigue la pista de los cambios en todos los árboles de directorios en el tiempo. Los ficheros *y* los directorios están versionados. Como resultado, hay comandos reales en el lado del cliente como **mover** y **copiar** que operan en ficheros y directorios.

Confirmaciones atómicas
: Una confirmación o bien entra en el repositorio completamente, o no entra en absoluto. Esto permite a los desarrolladores construir y confirmar cambios como unidades lógicas.

Metadatos versionados
: Cada fichero y directorio tiene un conjunto invisible de "propiedades" adjuntos. PUede inventarse y almacenar cualquier par de clave/valor que desee. Las propiedades se versionan en el tiempo, igual que el contenido de los ficheros.

Elección de capas de red
: Subversion tiene una noción abstracta del acceso al repositorio, haciéndo que la gente pueda implementar nuevos mecanismos de red fácilmente. El "avanzado" servidor de red de Subversion es un módulo para el servidor web Apache, que habla una variante de HTTP llamada WebDAV/DeltaV. Esto dota a Subversion una gran ventaja en estabilidad e interoperatividad, y proporciona varias características importantes gratis: autentificación, autorización, compresión de la transmisión y navegación del repositorio, por ejemplo. También está disponible un proceso servidor de Subversion independiente. Este servidor habla un protocolo propio que puede encapsularse fácilmente sobre ssh.

Manejo de datos consistente
: Subversion expresa las diferencias entre ficheros usando un algoritmo de diferenciación binario, que funciona exactamente igual tanto en ficheros de texto (legibles por los humanos) como en ficheros binarios (que no son legibles por nosotros). Ambos tipos de ficheros se almacenan igualmente comprimidos en el repositorio, y las diferencias se transmiten en ambas direcciones por la red.

Etiquetado y creación de ramas eficiente
: El coste de crear una rama o una etiqueta no necesita ser proporcional al tamaño del proyecto. Subversion crea ramas y etiquetas símplemente copiando el proyecto, utilizando un mecanismo similar a los vínculos duros. Por tanto estas operaciones llevan un tiempo pequeño y constante, y muy poco espacio en el repositorio.

Extensibilidad
: Subversion no tiene lastre histórico; está implementado como una colección de librerías C compartidas con APIS bien definidas. Esto hace que Subversion sea extremadamente mantenible y se pueda utilizar por otras aplicaciones y lenguajes.

1.4. Instalando TortoiseSVN

1.4.1. Requerimientos del sistema

TortoiseSVN se ejecuta en Windows 2000 SP2, Windows XP o superiores. Windows 98, Windows ME y Windows NT4 ya no se soportan desde TortoiseSVN 1.2.0, pero aún puede descargar las versiones más antiguas si realmente las necesita.

Si encuentra algún problema durante o después de la instalación de TortoiseSVN, por favor visite primero Apéndice A, *Preguntas más frecuentes (FAQ)*.

1.4.2. Instalación

TortoiseSVN viene con un instalador fácil de utilizar. Haga doble click en el fichero de instalación y siga las instrucciones. El instalador se encargará del resto.

Importante

Necesita privilegios de Administrador para instalar TortoiseSVN.

1.4.3. Packs de idiomas

El interfaz de usuario de TortoiseSVN se ha traducido a muchos idiomas distintos, por lo que es posible que pueda descargar un pack de idioma que se ajuste a sus necesidades. Puede encontrar los packs de idioma en nuestra *página de estado de las traducciones* [http://tortoisesvn.net/translation_status]. Y si aún no hay un pack de idioma disponible, por qué no unirse al equipo y enviar su propia traducción ;-)

Cada pack de idioma está empaquetado como un instalador `.exe`. Sólo tiene que ejecutar el programa de instalación y seguir las instrucciones. La siguiente vez que reinicie, tendrá disponible la traducción.

1.4.4. Corrector ortográfico

TortoiseSVN incluye un corrector ortográfico que le permite comprobar sus mensajes de registro en las confirmaciones. Esto es especialmente útil si el idioma del proyecto no es su lengua materna. El corrector ortográfico utiliza los mismos diccionarios que *OpenOffice* [http://openoffice.org] y *Mozilla* [http://mozilla.org].

El instalador automáticamente añade los diccionarios de inglés de EE.UU. y de Reino Unido. Si desea tenerlos en otros idiomas, la opción más sencilla es símplemente instalar uno de los packs de idiomas de TortoiseSVN. Eso instalará los ficheros de diccionario adecuados junto con el interface de usuario de TortoiseSVN en ese idioma. La siguiente vez que reinicie, el diccionario también estará disponible.

O puede instalar los diccionarios usted mismo. Si tiene OpenOffice o Mozilla instalados, puede copiar esos diccionarios, que se encuentran en las carpetas de instalación de esas aplicaciones. Si no, deberá descargar los ficheros de diccionario necesarios desde *http://wiki.services.openoffice.org/wiki/Dictionaries*

Una vez que tenga los ficheros de diccionario, seguramente deberá cambiarlos de nombre para que los nombres de los ficheros sólo contengan los caracteres del idioma. Por ejemplo:

- `es_ES.aff`
- `es_ES.dic`

Luego sólo tiene que copiarlos en la subcarpeta `bin` de la carpeta de instalación de TortoiseSVN. Normalmente ésta será `C:\Archivos de programa\TortoiseSVN\bin`. Si no desea jugar con

la subcarpeta `bin`, también puede poner los ficheros del corrector ortográfico en `C:\Archivos de programa\TortoiseSVN\Languages`. Si esa carpeta no está ahí, tendrá que crearla primero. La siguiente vez que inicie TortoiseSVN, podrá utilizar el corrector ortográfico.

Si instala múltiples diccionarios, TortoiseSVN utilizará estas reglas para seleccionar cuál utilizar.

1. Compruebe la configuración `tsvn:projectlanguage`. Lea Sección 4.17, “Configuración del proyecto” para encontrar información sobre cómo se establecen propiedades de proyecto.

2. Si no se ha establecido un idioma del proyecto, o ese idioma no está instalado, inténtelo con el idioma que corresponde al de Windows.

3. Si no funciona el idioma exacto de Windows, pruebe el idioma “Base”, por ejemplo, `es_MX` (Español-México) se transformaría en `es_ES` (Español).

4. Si nada de lo anterior funciona, entonces el idioma por defecto es el Inglés, que se incluye con la instalación estándar.

Capítulo 2. Conceptos básicos de control de versiones

Este capítulo es una versión ligeramente modificada del mismo capítulo en el libro de Subversion. Una versión en línea del libreo de Subversion está disponible aquí: *http://svnbook.red-bean.com/*.

Este capítulo es una introducción corta e informal a Subversion. Si el control de versiones es nuevo para usted, este capítulo es definitivamente para usted. Empezamos con una discusión de los conceptos generales de control de versiones, nos hacemos camino dentro de las ideas específicas que hay tras Subversion, y mostramos algunos ejemplos sencillos de Subversion en acción.

Incluso aunque los ejemplos en este capítulo muestran a gente compartiendo colecciones de código fuente de programas, tenga en cuenta que Subversion puede manejar cualquier collección de ficheros - no está limitado a ayudar a los programadores de ordenadores.

2.1. El repositorio

Subversion es un sistema centralizado para compartir información. En su núcleo está un *repositorio*, que es un almacén central de datos. El respositorio almacena información en forma de un *árbol de ficheros* - una jerarquía típica de ficheros y directorios. Cualquier número de *clientes* se conectan al repositorio, y luego leen o esriben esos ficheros. Al escribir datos, el cliente hace que la información esté disponible para los otros; al leer los datos, el cliente recibe la información de los demás.

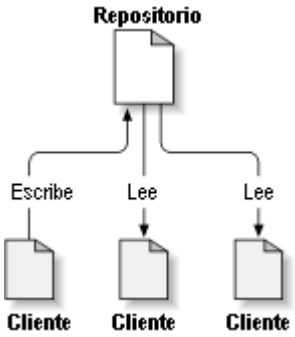


Figura 2.1. Un sistema típico cliente/servidor

¿Y ésto por qué es interesante? Por ahora, eso suena a la definición típica de un servidor de ficheros típico. Y de hecho, el repositorio *es* una clase de servidores de ficheros, pero no el habitual. Lo que hace al repositorio de Subversion especial es que *recuerda todos los cambios* que alguna vez se hayan escrito en él: cada cambio en cada fichero, e incluso los cambios en el propio árbol de directorios, como el añadir, borrar o reorganizar ficheros y directorios.

Cuando un cliente lee datos de un repositorio, normalmente ve únicamente la última versión del árbol de ficheros. Pero el cliente también tiene la capacidad de ver estados *previos* del sistema de ficheros. Por ejemplo, un cliente puede hacer preguntas históricas, como “¿qué contenía este directorio el último miércoles?”, o “¿quién fue la última persona que cambió este fichero, y qué cambios hizo?” Esta es la clase de preguntas que forman el corazón de cualquier *sistema de control de versiones*: son sistemas que están diseñados para guardar y registrar los cambios a los datos a lo largo del tiempo.

2.2. Modelos de versionado

Todos los sistemas de control de versiones tienen que resolver los mismos problemas fundamentales: ¿cómo permitirá el sistema compartir información entre usuarios, pero evitando que ellos accidentalmente

se pisen unos a otros? Es demasiado sencillo que los usuarios accidentalmente sobreescriban los cambios del otro en el repositorio.

2.2.1. El problema de compartir ficheros

Considere este escenario: suponga que tiene dos compañeros de trabajo, Harry y Sally. Cada uno decide editar el mismo fichero del repositorio a la vez. Si Harry graba sus cambios en el repositorio primero, el posible que (unos momentos después) Sally pueda accidentalmente sobreescribirlos con su propia versión nueva del fichero. Mientras que la versión del fichero de Harry no se ha perdido para siempre (porque el sistema recuerda cada cambio), cualquier cambio que Harry hizo *no estará* en la versión nueva del fichero de Sally, porque para empezar ella nunca vió los cambios de Harry. El trabajo de Harry está aún efectivamente perdido - o al menos falta en la última versión del fichero - y probablemente por accidente. ¡Esta es una situación que definitivamente tenemos que evitar!

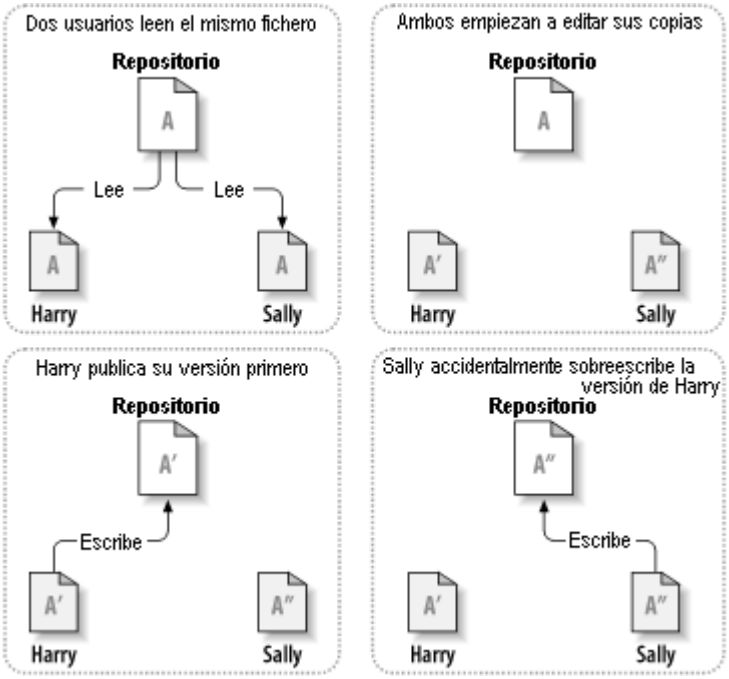


Figura 2.2. El problema a evitar

2.2.2. La solución bloquear-modificar-desbloquear

Muchos sistemas de control de versiones utilizan un modelo *bloquear-modificar-desbloquear* para enfrentarse a este problema, que es una solución muy simple. En estos sistemas, el repositorio sólo permite que una persona cambie un fichero al mismo tiempo. Harry primero debe *bloquear* el fichero antes de que pueda empezar a hacer cambios en él. Bloquear un fichero se parece mucho a tomar prestado un libro de la biblioteca; si Harry ha bloqueado un fichero, entonces Sally no puede hacer ningún cambio en él. Si ella intenta bloquear el fichero, el repositorio le denegará la petición. Todo lo que ella puede hacer es leer el fichero, y esperar a que Harry termine sus cambios y libere su bloqueo. Después de que Harry desbloquee el fichero, se acabó su turno, y ahora Sally puede bloquear y editar.

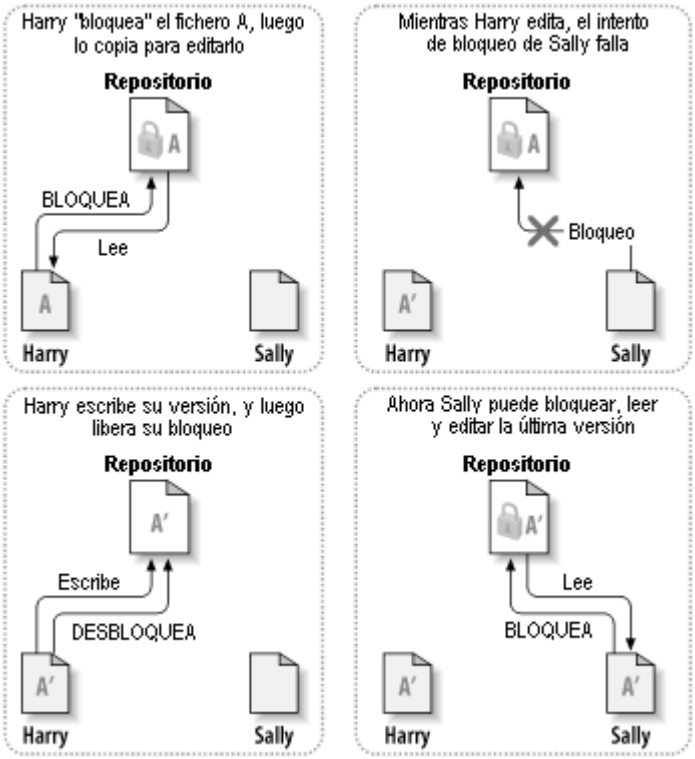


Figura 2.3. La solución bloquear-modificar-desbloquear

El problema con el modelo bloquear-modificar-desbloquear es que es un poco restrictivo, y a menudo se convierte en una calle cortada para los usuarios:

- *El bloqueo causa muchos problemas administrativos.* A veces Harry bloqueará un fichero y luego se olvidará de ello. Mientras tanto, dado que Sally está aún esperando para editar el fichero, sus manos están atadas. Y Harry se va de vacacioens. Ahora Sally tiene que buscar a un administrador para que libere el bloqueo de Harry. La situación acaba causando un montón de retraso y pérdida de tiempo innecesarios.

- *El bloqueo puede causar procesos en serie innecesarios.* ¿Qué ocurre si Harry está editando el inicio de un fichero de texto, y Sally simplemente quiere cambiar la parte final del mismo fichero? Esos cambios no se superponen en absoluto. Ellos podrían fácilmente editar el fichero de forma simultánea, y no habría ningún daño, asumiendo que los cambios se fusionaran correctamente. No hay necesidad de que se turnen en esta situación.

- *El bloqueo puede causar una falsa sensación de seguridad.* Imagine que Harry bloquea y edita el fichero A, mientras Sally simultáneamente bloquea y edita el fichero B. Pero suponga que A y B dependen uno del otro, y que los cambios hechos a cada uno son semánticamente incompatibles. De repente A y B ya no funcionan juntos. El sistema de bloqueo no tiene forma de prevenir este problema - sin embargo, de alguna forma dió una sensación de falsa seguridad. Es fácil para Harry y Sally imaginar que al bloquear los ficheros, cada uno está empezando una tarea segura y aislada, y por tanto les inhibe de discutir sus cambios incompatibles en un momento temprano.

2.2.3. La solución copiar-modificar-fusionar

Subversion, CVS y otros sistemas de control de versiones utilizan un modelo *copiar-modificar-fusionar* como alternativa al bloqueo. En este modelo, el cliente de cada usuario lee el repositorio y crea una *copia de trabajo* personal del fichero o del proyecto. Luego, los usuarios trabajan en paralelo, modificando sus copias privadas. Finalmente, las copias privadas se fusionan juntas en una nueva versión final. El sistema de control de versiones a menudo ofrece ayuda en la fusión, pero al final la persona es la responsable de hacer que ocurra correctamente.

Aquí hay un ejemplo. Digamos que tanto Harry como Sally crean copias de trabajo del mismo proyecto, copiado del repositorio. Ellos trabajan concurrentemente, y hacen los cambios al mismo fichero A dentro de sus copias. Sally es la primera en grabar sus cambios en el repositorio. Cuando Harry intenta grabar sus cambios más tarde, el repositorio le informa que su fichero A está *desactualizado*. En otras palabras, que el fichero A en el repositorio ha cambiado de alguna forma desde la última vez que lo copió. Por lo que Harry le pide a su cliente que *fusione* cualquier nuevo cambio del repositorio dentro de su copia de trabajo del fichero A. Lo más seguro es que los cambios de Sally no se superpongan a los suyos; por lo que una vez que ambos conjuntos de cambios se han integrado, él graba su copia de trabajo de nuevo en el repositorio.

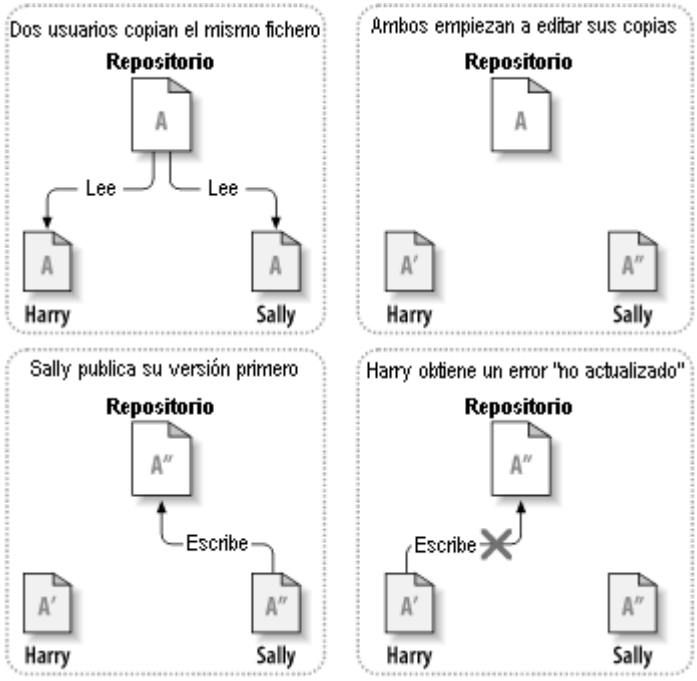


Figura 2.4. La solución copiar-modificar-fusionar

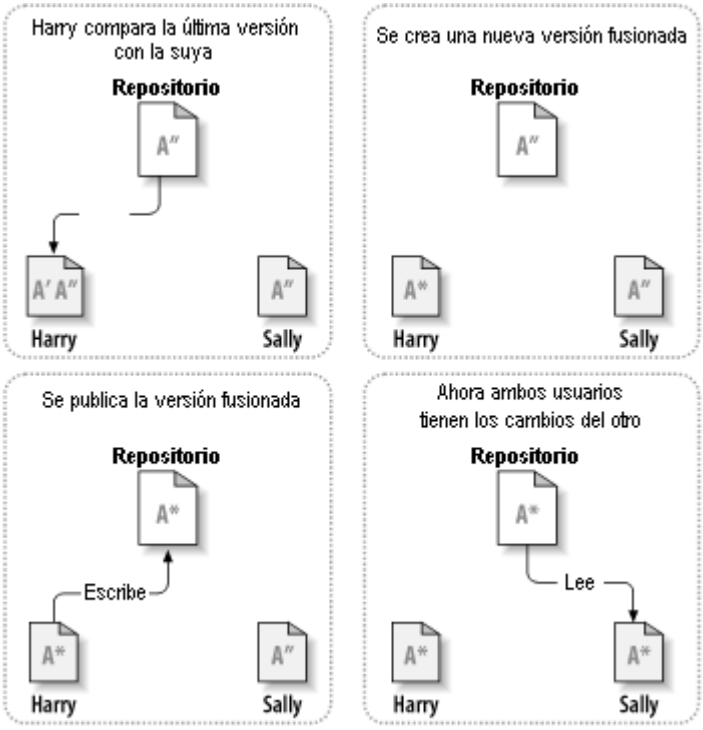


Figura 2.5. ...Copiar-modificar-fusionar continuado

¿Pero qué ocurre si los cambios de Sally *sí* se superponen a los cambios de Harry? ¿Qué hacemos entonces? La situación se denomina un *conflicto*, y normalmente no es mucho problema. Cuando Harry le pide a su cliente que fusione los últimos cambios del repositorio en su copia de trabajo, su copia del fichero A se marca de alguna forma como que está en un estado de conflicto: él será capaz de ver ambos conjuntos de cambios conflictivos, y manualmente podrá elegir entre ellos. Tenga en cuenta que el software no puede resolver conflictos automáticamente; sólo los humanos son capaces de entender y hacer las elecciones necesarias de forma inteligente. Una vez que Harry haya resuelto manualmente los cambios que se superponían (¡quizás discutiendo el conflicto con Sally!), puede volcar de forma segura el fichero fusionado al repositorio.

El modelo copiar-modificar-fusionar puede sonar un poco caótico, pero en la práctica, funciona extremadamente bien. Los usuarios pueden trabajar en paralelo, sin que tengan que esperar nunca uno por otro. Cuando trabajan en los mismos ficheros, resulta que la mayoría de los cambios concurrentes no se superponen en absoluto; los conflictos no son frecuentes. Y el tiempo que lleva resolver conflictos es mucho menor que el tiempo perdido por un sistema bloqueante.

Al final, todo se reduce a un factor crítico: la comunicación entre usuarios. Cuando los usuarios se comunican de forma pobre, aumentan los conflictos sintácticos y semánticos. No hay sistema capaz de forzar a los usuarios a comunicarse perfectamente, y no hay sistema que pueda detectar conflictos semánticos. Por lo que no hay motivo para que se le prometa falsamente que un sistema con bloqueos prevendrá de alguna forma los conflictos; en la práctica, el bloqueo parece inhibir la productividad más que otra cosa.

Hay una situación común donde el modelo bloquear-modificar-desbloquear resulta mejor, y es cuando tiene ficheros no-fusionables. Por ejemplo si su repositorio contiene algunas imágenes gráficas, y dos personas cambian la imagen a la vez, no hay forma de fusionar esos cambios. O Harry o Sally perderán sus cambios.

2.2.4. ¿Qué hace Subversion?

Subversion utiliza la solución copiar-modificar-mezclar por defecto, y en muchos casos esto es todo lo que necesitará. Sin embargo, desde la Versión 1.2, Subversion también admite bloqueo de ficheros, por

lo que si tiene ficheros no-fusionables, o si simplemente está forzado a una política de bloqueo por la dirección, Subversion seguirá teniendo las características que necesita.

2.3. Subversion en acción

2.3.1. Copias de trabajo

Ya ha oído hablar sobre las copias de trabajo; ahora le demostraremos cómo las crea y las utiliza el cliente de Subversion.

Una copia de trabajo de Subversion es un árbol de directorios ordinario en su sistema local, conteniendo una colección de ficheros. Puede editar estos ficheros como desee, y si son ficheros de código fuente, puede compilar su programa de la forma habitual. Su copia de trabajo es su propia área de trabajo privada: Subversion nunca incorporará los cambios de otra gente, ni hará que sus cambios estén disponibles para los demás, a menos que se lo pida explícitamente.

Después de que haya hecho algunos cambios en los ficheros dentro de su copia de trabajo y haya verificado que funcionan correctamente, Subversion le provee de comandos para *publicar* sus cambios para los demás que trabajan con usted en su proyecto (escribiendo en el repositorio). Si los demás publican sus propios cambios, Subversion le provee de comandos para fusionar esos cambios dentro de su copia de trabajo (leyendo desde el repositorio).

Una copia de trabajo también contiene algunos ficheros extra, creados y mantenidos por Subversion, para ayudarse a llevar a cabo esos comandos. En particular, cada directorio detnro de su copia de trabajo contiene un subdirectorio llamado `.svn`, también conocido como el *directorio administrativo* de la copia de trabajo. Los ficheros dentro de los directorios administrativos ayudan a Subversion a reconocer qué ficheros contienen cambios no publicados, y qué ficheros están desactualizados respecto al trabajo de los demás.

Un repositorio típico de Subversion a menudo contiene los ficheros (o el código fuente) de varios proyectos; usualmente, cada proyecto es un subdirectorio en el árbol de ficheros del repositorio. Con esta disposición, una copia de trabajo de un usuario normalmente corresponderán a un subárbol particular del repositorio.

Por ejemplo, suponga que tiene un repositorio que contiene dos proyectos de software.

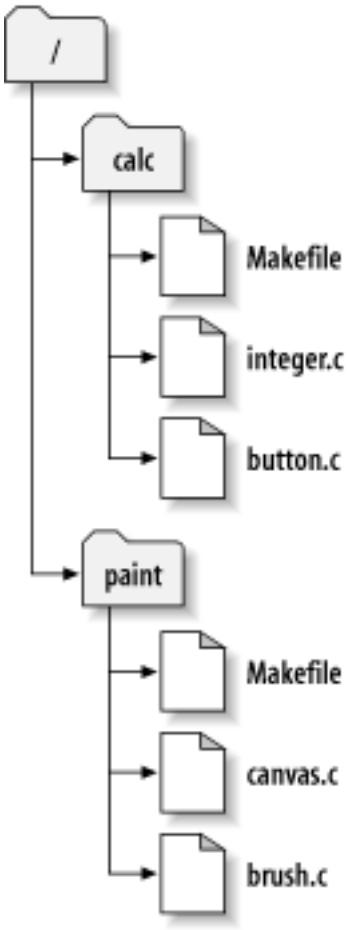


Figura 2.6. El sistema de ficheros del repositorio

En otras palabras, el directorio raíz del repositorio tiene dos subdirectorios, `paint` y `calc`.

Para obtener una copia de trabajo, primero debe *obtener* algún subárbol del repositorio. (El término *obtener* puede sonar como que tenga algo que ver con el bloqueo o la reserva de recursos, pero no es cierto; simplemente crea una copia privada del proyecto para usted).

Suponga que ha hecho cambios a `button.c`. Dado que el directorio `.svn` recuerda la fecha de modificación y los contenidos originales del fichero, Subversion puede decirle que ha cambiado el fichero. Sin embargo, Subversion no hace públicos sus cambios hasta que explícitamente se lo pida. El acto de publicar sus cambios se conoce más comúnmente como *confirmar* (o *enviar*) los cambios al repositorio.

Para publicar sus cambios para los demás, puede utilizar el comando de Subversion **commit**.

Ahora que sus cambios a `button.c` se han confirmado en el respositorio, si cualquier otro usuario obtiene una copia de trabajo de `/calc`, verán sus cambios en la última versión del fichero.

Suponga que tiene un colaborador, Sally, que obtuvo una copia de trabajo de `/calc` al mismo tiempo que usted. Cuando ha confirmado sus cambios en `button.c`, la copia de trabajo de Sally se queda sin cambios; Subversion sólo modifica las copias de trabajo cuando lo pide el usuario.

Para poner al día su proyecto, Sally puede pedirle a Subversion *actualizar* su copia de trabajo, utilizando el comando de Subversion **actualizar**. Esto incorporará sus cambios en la copia de trabajo de Sally, junto con cualquier otro que se haya confirmado desde que ella lo obtuvo.

Tenga en cuenta que Sally no necesita especificar qué ficheros actualizar; Subversion utiliza la información en el directorio `.svn`, y más información desde el repositorio, para decidir qué ficheros deben ponerse al día.

2.3.2. URLs de repositorio

Los repositorios de Subversion pueden ser accedidos por muchos métodos diversos - en discos locales, o a través de varios protocolos de red. La ruta de un repositorio es siempre, sin embargo, una URL. El esquema URL indica el método de acceso:

Esquema	Método de acceso
`file://`	Acceso directo al repositorio en el disco local o de red.
`http://`	Acceso utilizando el protocolo WebDAV a un servidor Apache configurado para Subversion.
`https://`	Lo mismo que `http://`, pero con encriptación SSL.
`svn://`	Acceso TCP/IP sin autentificación utilizando un protocolo personalizado a un servidor `svnserve`.
`svn+ssh://`	Acceso TCP/IP autentificado y encriptado utilizando un protocolo propio a un servidor `svnserve`.

Tabla 2.1. URLs de acceso al repositorio

En su mayoría, las URLs de Subversion utilizan la sintaxis estándar, permitiendo que se especifiquen nombres de servidor y números de puertos como parte de la URL. El método de acceso `file://` se utiliza normalmente para el acceso local, aunque puede utilizarse con rutas UNC a equipos en red. La URL por lo tanto toma la forma `file://nombredeequipo/ruta/al/repositorio`. Para la máquina local, la parte `nombredeequipo` de la URL debe estar o ausente o ser `localhost`. Por esta razón, las rutas locales normalmente aparecen con tres barras, `file:///ruta/al/repositorio`.

Además, los usuarios del esquema `file://` en las plataformas Windows necesitarán utilizar una sintaxis “estándar” no oficial para acceder a los repositorios que están en la misma máquina, pero en una letra de unidad diferente de la unidad actual del cliente. Cualquiera de las siguientes sintaxis de URL funcionarán, donde `X` es la unidad en la que reside el repositorio:

```
file:///X:/ruta/al/repositorio
...
file:///X|/ruta/al/repositorio
...
```

Tenga en cuenta que las URLs utilizan las barras hacia delante (las de dividir) incluso aunque la forma nativa (no-URL) de una ruta en Windows utiliza las barras contrarias.

Puede acceder con seguridad a un repositorio FSFS utilizando una carpeta compartida de red, pero *no* puede acceder a un repositorio BDB de esta forma.

Aviso

No cree o acceda a un repositorio Berkeley DB en una unidad de red compartida. *No* puede existir en un sistema de archivos remoto. Ni siquiera si tiene la unidad de red mapeada a una letra de unidad. Si intenta usar Berkeley DB en una unidad de red compartida, los resultados son imprevisibles - puede ver desde el principio errores misteriosos, o pueden pasar meses antes de que descubra que su base de datos del repositorio está corrupta de una forma inimaginable.

2.3.3. Revisiones

Una operación **svn commit** puede publicar los cambios de cualquier número de ficheros y carpetas como una única transacción atómica. En su copia de trabajo, puede cambiar el contenido de los ficheros, crear,

borrar, renombrar y copiar ficheros y directorios, y luego confirmar el conjunto completo de cambios como una unidad.

En el repositorio, cada confirmación se trata como una transacción atómica: o bien todos los cambios de la confirmación se llevan a cabo, o bien ninguno de ellos se realiza. Subversion aplica esta atomicidad en caso de errores en el programa, errores del sistema, problemas de red, y otras acciones del usuario.

Cada vez que el repositorio acepta una confirmación, crea un nuevo estado del árbol de ficheros, llamado *revisión*. A cada revisión se le asigna un número natural único, un número mayor que la revisión anterior. La revisión inicial de un repositorio recién creado se numera como cero, y consiste únicamente en un directorio raíz vacío.

Una buena forma de visualizar el repositorio es como una serie de árboles. Imagine una fila de números de revisiones, empezando en 0, de izquierda a derecha. Cada número de revisión tiene un árbol colgando debajo, y cada árbol es una “foto” de cómo estaba el repositorio tras cada confirmación.

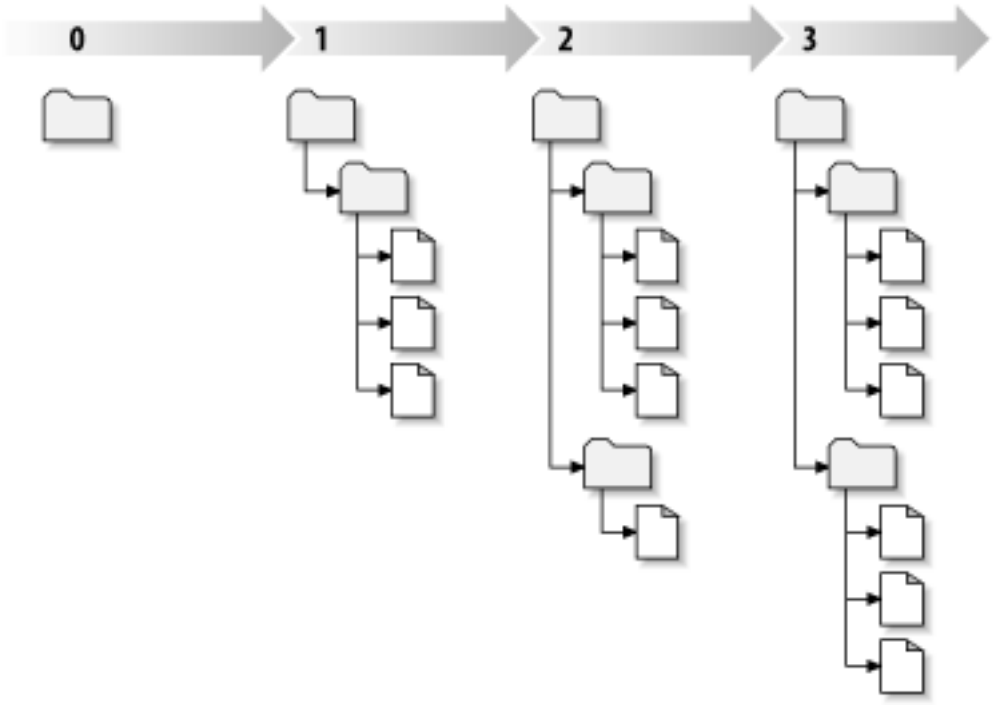


Figura 2.7. El repositorio

> **Números globales de revisión**
>
> Al contrario que la mayoría del resto de sistemas de control de versiones, los números de revisión de Subversion se aplican a *árboles completos*, no a los ficheros individuales. Cada número de revisión selecciona un árbol entero, un estado particular del repositorio tras algún cambio confirmado. Otra forma de verlo es pensar que la revisión N representa el estado del repositorio tras la confirmación N-ésima. Cuando un usuario de Subversion habla de la "revision 5 de `foo.c`", realmente quieren decir "`foo.c` tal y como estaba en la revisión 5". ¡Tenga en cuenta que, en general, las revisiones N y M de un fichero *no* tienen por qué ser diferentes!

Es importante que tenga en cuenta que las copias de trabajo no siempre se corresponden a una única revisión en el repositorio; pueden contener ficheros de varias revisiones.Por ejemplo, suonga que obtiene una copia de trabajo de un repositorio cuya revisión más reciente es la 4:

```
calc/Makefile:4
```

```
integer.c:4
button.c:4
```

En este momento, esta copia de trabajo corresopnde exactamente a la revisión 4 en el respositorio. Sin embargo, suponga que ha hecho cambios al fichero `button.c`, y confirme ese cambio. Asumiendo que no se haya llevado a cambio ninguna otra confirmación, su confirmación creará la revisión 5 en el repositorio, y su copia de trabajo quedará así:

```
calc/Makefile:4
     integer.c:4
     button.c:5
```

Suponga que, en este punto, Sally hace un cambio a `integer.c`, creando la revisión 6. Si utiliza **svn update** para actualizar su copia de trabajo, obtendrá ésto:

```
calc/Makefile:6
     integer.c:6
     button.c:6
```

Los cambios de Sally a `integer.c` aparecerán en su copia de trabajo, y su cambio estará aún presente en `button.c`. En este ejemplo, el texto de `Makefile` es idéntico en las revisiones 4, 5, y 6, pero Subversion marcará su copia de trabajo de `Makefile` con la revisión 6 para indicar que aún está actualizado. Por lo que, después de que haga una actualización limpia en la parte superior de su copia de trabajo, generalmente obtendrá exactamente una revisión del repositorio.

2.3.4. Cómo se unen las copias de trabajo al repositorio

Por cada fichero en un directorio de trabajo, Subversion grabará dos piezas esenciales de información en el área administrativa `.svn/`:

- en qué revisión se basa su fichero de trabajo (lo que se denomina la *revisión de trabajo*), y
- una fecha que indica cuándo se actualizó por última vez la copia local por el repositorio.

Con esta información, hablando con el repositorio, Subversion puede decirle en cuál de los siguientes cuatro estados está un fichero de trabajo:

Sin cambios, y actualizado
: El fichero no se ha cambiado en el directorio de trabajo, y no se han confirmado cambios a ese fichero en el repositorio desde su revisión de trabajo. Una **confirmación** de ese fichero no hará nada, y una **actualización** de ese fichero no hará nada.

Cambiado localmente, y actualizado
: El fichero ha sido cambiado en el directorio de trabajo, y no se ha confirmado ningún cambio a ese fichero en el repositorio desde su revisión base. Hay cambios locales que no se han confirmado en el repositorio, por lo que al **confirmar** el fichero se conseguirá publicar sus cambios, y al **actualizar** el fichero no se realizará nada.

Sin cambios, y desactualizado
: El fichero no ha sido cambiado en el directorio de trabajo, pero ha sido cambiado en el repositorio. El fichero deberá ser actualizado en algún momento, para actualizarlo con la revisión pública. Un comando **confirmar** sobre el fichero no hará nada, y al **actualizar** el fichero se traerán los últimos cambios a su copia de trabajo.

Cambiado localmente, y desactualizado
: El fichero se ha cambiado tanto en el directorio de trabajo como en el repositorio. Un comando **confirmar** sobre el fichero fallará con un error *desactualizado*. El fichero debería actualizarse primero; al **actualizar** se intentará fusionar los cambios públicos con los cambios locales. Si

Subversion no puede completar la fusión de una forma plausible automáticamente, le dejará al usuario la tarea de resolver el conflicto.

2.4. Sumario

Hemos cubierto un número de conceptos fundamentales de Subversion en este capítulo:

- Hemos introducido las nociones de un repositorio central, la copia de trabajo del cliente, y la lista de árboles de revisiones del repositorio.
- Hemos visto algunos ejemplos simples sobre cómo dos colaboradores pueden utilizar Subversion para publicar y recibir los cambios de uno a otro, utilizando el modelo 'copiar-modificar-fusionar'.
- Hemos hablado un poco sobre la forma en la que Subversion controla y maneja la información en una copia de trabajo.

Capítulo 3. El repositorio

Sin importar el protocolo que use para acceder a sus repositorios, siempre necesita crear al menos un repositorio. Esto puede hacerse o bien con el cliente de línea de comandos de Subversion o con TortoiseSVN.

Si todavía no ha creado un repositorio de Subversion, ahora es el momento de hacerlo.

3.1. Creación de repositorios

Puede crear un repositorio bajo el sistema FSFS o bien utilizando el anterior formato Berkeley Database (BDB). El formato FSFS es generalmente más rápido y más fácil de administrar, y funciona en unidades de red compartidas y en Windows 98 sin problemas. El formato BDB fue considerado durante un tiempo más estable simplemente porque se ha utilizado durante más tiempo, pero dado que FSFS se usa desde hace varios años, ese argumento ahora es más bien débil. Lea *Eligiendo un almacén de datos* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.planning.html#svn.reposadmin.basics.backends] en el libro de Subversion si desea más información.

3.1.1. Creando un repositorio con el cliente de línea de comandos

1. Cree una carpeta vacía con el nombre SVN (por ejemplo, `D:\SVN\`), que se usará como la raíz de todos sus repositorios.
2. Cree otra carpeta `MiNuevoRepositorio` dentro de `D:\SVN\`
3. Abra un símbolo del sistema (o ventana MS-DOS), vaya a `D:\SVN\` y escriba

```
svnadmin create --fs-type bdb MiNuevoRepositorio
```

o

```
svnadmin create --fs-type fsfs MiNuevoRepositorio
```

Ahora ya tiene un nuevo repositorio que se encuentra en `D:\SVN\MiNuevoRepositorio`.

3.1.2. Creando el repositorio con TortoiseSVN

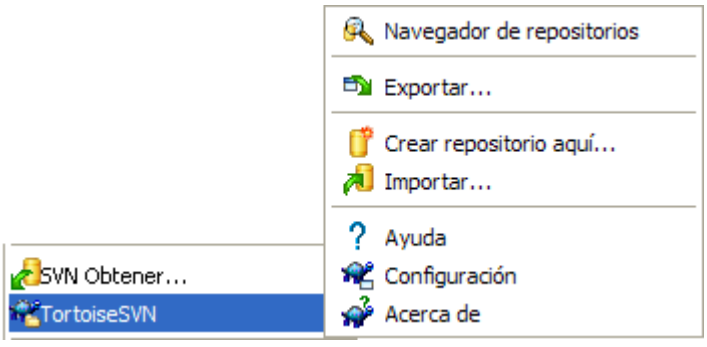


Figura 3.1. El menú de TortoiseSVN para carpetas no versionadas

1. Abra el explorador de Windows

2. Cree una nueva carpeta y llámela por ejemplo `SVNRepositorio`

3. Haga click con el botón derecho sobre la carpeta recién creada y seleccione TortoiseSVN → Crear Repositorio aquí....

   Entonces se creará un repositorio dentro de la nueva carpeta. *¡¡¡No edite los ficheros!!!*. Si obtiene algún error asegúrese de que la carpeta esté vacía y que no esté protegida contra escritura.

Sugerencia

TortoiseSVN ya no ofrece la opción de crear repositorios BDB, aunque aún puede utilizar el cliente de línea de comandos para crearlos. Los repositorios FSFS son generalmente más sencillos de mantener, y también nos facilita el mantenimiento de TortoiseSVN debido a los problemas de compatibilidad entre las diferentes versiones de BDB.

Las versiones futuras de TortoiseSVN no soportarán el acceso `file://` a repositorios BDB debido a estos problemas de compatibilidad, aunque por supuesto sí soportará este formato de repositorio cuando se acceda utilizando un servidor a través de los protocolos `svn://`, `http://` o `https://`. Por esta razón, le recomendamos encarecidamente que cualquier nuevo repositorio al que deba accederse utilizando el protocolo `file://` se cree como FSFS.

Por supuesto también le recomendamos que no utilice el acceso `file://` en absoluto, salvo para propósitos de pruebas locales. Utilizar un servidor es más seguro y confiable para todo lo que no sean proyectos de un único desarrollador.

3.1.3. Acceso local al repositorio

Para acceder a su repositorio local, necesita la ruta a esa carpeta. Recuerde que Subversion espera todas las rutas de repositorios con el formato `file:///C:/RepositorioSVN/`. Tenga en cuenta el uso de las barras de dividir.

Para acceder a un repositorio que se encuentre en una unidad de red compartida puede o bien utilizar mapeado de unidades, o bien usar la ruta UNC. El formato de una ruta UNC es `file://NombreDelServidor/ruta/al/repositorio/`. Observe que sólo hay 2 barras invertidas aquí.

Antes de SVN 1.2, las rutas UNC tenían que estar dadas en la forma más oscura `file:///\NombreDelServidor/ruta/al/repositorio`. Esta forma aún se puede utilizar, pero no está recomendada.

Aviso

No cree o acceda a un repositorio Berkeley DB en una unidad de red compartida. *No* puede existir en un sistema de archivos remoto. Ni siquiera si tiene la unidad de red mapeada a una letra de unidad. Si intenta usar Berkeley DB en una unidad de red compartida, los resultados son imprevisibles - puede que aparezcan extraños errores desde un primer momento, o pueden pasar meses antes de que descubra que la base de datos del repositorio está corrupta de una forma casi imperceptible.

3.1.4. Accediendo a un repositorio en una unidad de red

Aunque en teoría es posible poner un repositorio FSFS en una unidad de red y hacer que varios usuarios accedan a él utilizando el protocolo `file://`, esto realmente *no* está recomendado. De hecho, nosotros lo desaconsajemos *encarecidamente*, y no soportamos este uso.

En primer lugar, está dando a todos los usuarios acceso directo de escritura en el repositorio, por lo que cualquier usuario podría accidentalmente borrar el repositorio completo o hacerlo inutilizable de cualquier otra forma.

En segundo lugar, no todos los protocolos de compartición de ficheros de red soportan el bloqueo que Subversion necesita, por lo que puede encontrar que su repositorio se corrompe. Puede que no ocurra al principio, pero un día dos usuarios intentarán acceder al repositorio al mismo tiempo.

En tercer lugar, también debe establecer los permisos de los ficheros. Puede ser sencillo en una unidad de red nativa de Windows, pero en SAMBA esto es particularmente dificil.

El acceso `file://` está pensado únicamente para el acceso local por un único usuario, en particular para testeos y depuraciones. Cuando desee compartir el repositorio, *realmente* necesita configurar un servidor de forma apropiada, y realmente no es tan dificil como pueda pensar. Lea Sección 3.5, “Accediendo al repositorio” para obtener indicaciones sobre cómo elegir y configurar un servidor.

3.1.5. Organización del repositorio

Antes de que importe sus datos al repositorio, primero debería pensar cómo quiere organizar sus datos. Si utiliza uno de los patrones recomendados lo tendrá luego mucho más fácil.

Hay algunas formas estándar y recomendadas de organizar un repositorio. La mayoría de la gente crea un directorio `trunk` (tronco) para alojar la “línea principal del desarrollo, un directorio branches (ramas) para que contenga las copias/ramas, y un directorio tags (etiquetas) para contener las copias/etiquetas. Si un repositorio s”

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

Si un repositorio contiene múltiples proyectos, la gente a menudo indexa por ramas:

```
/trunk/paint
/trunk/calc
/branches/paint
/branches/calc
/tags/paint
/tags/calc
```

...o por proyecto:

```
/paint/trunk
/paint/branches
/paint/tags
/calc/trunk
/calc/branches
/calc/tags
```

Indexar por proyecto tiene sentido si los proyectos no están muy relacionados y cada uno se obtiene de forma individual. Para proyectos relacionados donde puede querer obtener todos los proyectos de golpe, o donde los proyectos están unidos y forman un paquete de distribución único, a menudo es mejor indexar por rama. De esta forma sólo tendrá un tronco para obtener, y las relaciones entre sub-proyectos se ven más fácilmente.

Si adopta una aproximación de primer nivel `/trunk /tags /branches`, no es necesario decir que tendrá que copiar el tronco completo para cada rama y etiqueta, y de alguna forma esta estructura ofrece la mayor flexibilidad.

Para proyectos que no tienen que ver, puede preferir utilizar repositorios separados. Cuando confirma los cambios, lo que cambia es el número de revisión del repositorio completo, no el número de revisión del proyecto. Si tiene 2 proyectos que no tengan que ver compartiendo un repositorio, eso puede llevar a que ocurran grandes lagunas en los números de revisión. Los proyectos Subversion y TortoiseSVN aparecen en el mismo servidor, pero son dos repositorios totalmente separados que permiten un desarrollo independiente, y no hay confusión sobre los números de compilación.

Por supuesto, tiene libertad para ignorar estos patrones comunes. Puede crear cualquier variación, la que mejor le venga a usted o a su equipo. Recuerde que cualquiera que sea la que elija, no es una elección inamovible. Puede reorganizar su repositorio en cualquier momento. Dado que las ramas y las etiquetas son directorios normales, TortoiseSVN puede mover o renombrarlas como desee.

Cambiar de una disposición a otra es sólo una cuestión de ejecutar una serie de movimientos en el lado del servidor; si no le gusta la forma en la que están organizadas las cosas en el repositorio, sólo tiene que ir moviendo los directorios.

Así que si no ha creado todavía una estructura básica de carpetas dentro de su repositorio, debería hacerlo ahora. Hay dos formas de conseguirlo. Si simplemente desea crear una estructura `/trunk /tags / branches`, puede utilizar el visor de repositorios para crear las tres carpetas (en tres confirmaciones distintas). Si desea crear una jerarquía más profunda, es más fácil crear primero una estructura de carpetas en el disco e importarla en una única confirmación, por ejemplo:

1. cree una nueva carpeta en su disco duro
2. cree la estructura de carpetas de primer nivel dentro de esa carpeta - ¡no ponga ningún fichero allí todavía!
3. importe esta estructura en el repositorio via click con el botón derecho en la carpeta y seleccionando TortoiseSVN → Importar.... Esto importará su carpeta temporal a la raíz del repositorio para crear la estructura básica del repositorio.

Tenga en cuenta que el nombre de la carpeta que está importando no aparece en el repositorio, sólo sus contenidos. Por ejemplo, cree la siguiente estructura de carpetas:

```
C:\Temp\New\trunk
C:\Temp\New\branches
C:\Temp\New\tags
```

Importe `C:\Temp\New` en la raíz del repositorio, que entonces contendrá:

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

3.2. Copia de seguridad del Repositorio

Cualquiera que sea el tipo de repositorio que use, es de vital importancia que mantenga copias de seguridad regulares, y que verifique la copia. Si el servidor falla, puede ser capaz de acceder a la versión más reciente de sus ficheros, pero sin el repositorio toda su historia se perderá para siempre.

La manera más sencilla (pero no recomendada) es tan simple como copiar la carpeta del repositorio a un medio de backup. Sin embargo, tiene que estar absolutamente seguro de que no hay ningún proceso accediendo a los datos. En este contexto, acceder significa *cualquier* tipo de acceso. En un repositorio BDB se escribe incluso cuando la operación sólo parece que necesite leer, como obtener el estado. Si se accede a su repositorio durante la copia (se deja un navegador web abierto, WebSVN, etc.) la copia puede que no valga para nada.

El método recomendado es ejecutar

```
svnadmin hotcopy ruta/al/repositorio ruta/al/backup --clean-logs
```

para crear una copia del repositorio de forma segura. Entonces hacer una copia de seguridad de la copia. La opción `--clean-logs` no es necesaria, pero quita cualquier fichero de log redundante cuando hace una copia de seguridad de un repositorio BDB; lo que puede ahorrar algo de espacio.

La herramienta `svnadmin` se instala automáticamente cuando instala el cliente de línea de comandos de Subversion. Si está instalando las herramientas de línea de comandos en un PC con Windows, la mejor forma de hacerlo es descargar la versión con Windows installer. Está comprimido de forma más eficiente que la versión `.zip`, por lo que la descarga es menor, y se encarga de establecer las rutas por usted. Puede descargar la última versión del cliente de línea de comandos desde *http://subversion.apache.org/getting.html*.

3.3. Scripts gancho en el lado del servidor

Un script gancho es un programa que se activa por algún evento del repositorio, tal como la creación de una nueva revisión o la modificación de una propiedad no versionada. A cada gancho se le proporciona suficiente información para decirle qué evento es, sobre qué destino(s) se está operando, y el nombre de usuario de la persona que lanzó el evento. Dependiendo de la salida del gancho o del estado de salida, el programa gancho puede continuar la acción, detenerla, o suspenderla de alguna forma. Por favor lea *Scripts gancho* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.create.html#svn.reposadmin.create.hooks] en el Libro de Subversion donde encontrará todos los detalles sobre la forma en la que están implementados los ganchos.

Estos scripts gancho se ejecutan por el servidor que hospeda el repositorio. TortoiseSVN también le permite configurar scripts ganchos del lado del cliente que se ejecutan localmente en ciertos eventos. Para más información consulte Sección 4.30.8, “Scripts gancho del lado del cliente”.

Puede encontrar scripts de ganchos en el directorio `hooks` del repositorio. Estos scripts de ejemplo son válidos para servidores Unix/Linux pero necesitan modificarse si su servidor está basado en Windows. El gancho puede ser un fichero batch o un ejecutable. El siguiente ejemplo muestra un fichero batch que puede ser usado para implementar un ganch pre-revprop-change.

```
rem Solo se permite cambiar mensajes de registro.
if "%4" == "svn:log" exit 0
echo No se puede cambiar la property '%4' >&2
exit 1
```

Tenga en cuenta que cualquier cosa que mande a la salida estándar se descartará. Si desea que aparezca un mensaje en el diálogo Confirmación Rechazada debe enviarlo a la salida de error. En un fichero batch esto se consigue usando `>&2`

3.4. Enlaces de obtener

Si desea hacer disponible su repositorio de Subversion a los demás, puede que desee incluir un vínculo a él desde su sitio web. Una forma de hacer esto más accesible es incluir un *enlace de obtener* para otros usuarios de TortoiseSVN.

Cuando se instala TortoiseSVN, se registra un nuevo protocolo `tsvn:`. Cuando un usuario de TortoiseSVN pulsa en un enlace con ese protocolo, aparece el diálogo obtener automáticamente con la URL del repositorio ya escrita.

Para incluir dicho enlace en su propia página HTML, necesita añadir un código que será similar a este:

```
<a href="tsvn:http://proyecto.dominio.org/svn/trunk">
</a>
```

Por supuesto será más bonito si añade una imagen adecuada. Puede utilizar el *logotipo de TortoiseSVN* [http://tortoisesvn.tigris.org/images/TortoiseCheckout.png] o puede proporcionar su propia imagen.

```
<a href="tsvn:http://proyecto.dominio.org/svn/trunk">
<img src=TortoiseCheckout.png></a>
```

También puede hacer el enlace apuntando a una revisión en concreto, por ejemplo

```
<a href="tsvn:http://proyecto.dominio.org/svn/trunk?100">
</a>
```

3.5. Accediendo al repositorio

Para utilizar TortoiseSVN (o cualquier otro cliente de Subversion), necesita un lugar donde poner sus repositorios. Puede o bien almacenar sus repositorios de forma local y acceder a ellos utilizando el protocolo `file://`, o puede ponerlos en un servidor y acceder a ellos con los protocolos `http://` o `svn://`. Los dos protocolos de servidor también pueden estar encriptados. Utilice `https://` o `svn+ssh://`, o puede utilizar `svn://` con SASL.

Si está utilizando un servicio de hospedaje público como *Google Code* [http://code.google.com/hosting/] o si su servidor ha sido ya preparado por alguna otra persona entonces no hay nada más que necesite hacer. Continúe en Capítulo 4, *Guía de uso diario*.

Si no tiene un servidor y trabaja solo, o si está simplemente evaluando Subversion y TortoiseSVN aisladamente, los repositorios locales son probablemente su mejor elección. Cree sencillamente un repositorio en su propio PC como se describe más arriba en Capítulo 3, *El repositorio*. Puede saltar el resto de este capítulo e ir directamente a Capítulo 4, *Guía de uso diario* para averiguar cómo empezar a utilizarlo.

Si estaba pensando en configurar un repositorio multiusuario en una unidad compartida de red, piénselo de nuevo. Lea Sección 3.1.4, “Accediendo a un repositorio en una unidad de red” para averiguar por qué creemos que eso es una mala idea. Preparar un servidor no es tan dificil como suena, y le dará mayor confiabilidad y probablemente también mejor velocidad.

Las siguientes secciones son una guía paso a paso sobre cómo puede configurar un servidor en una máquina Windows. Por supuesto también puede preparar un servidor en una máquina Linux, pero eso queda fuera del alcance de esta guía. Puede encontrar información más detallada sobre las diferentes opciones para montar un servidor de Subversion, y cómo elegir la mejor arquitectura para su situación, en el libro de Subversion bajo *Configuración del servidor* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.html].

3.6. Servidor Basado en Svnserve

3.6.1. Introducción

Subversion incluye Svnserve - un servidor ligero y autónomo que utiliza un protocolo propio sobre una conexión TCP/IP ordinaria. Es ideal para las instalaciones más pequeñas, o donde no se pueda utilizar un servidor completo Apache.

En la mayoría de los casos svnserve es más fácil de instalar y se ejecuta más rápido que el servidor basado en Apache, aunque no tiene algunas de las características más avanzadas. Y ahora que se incluye soporte SASL es fácil hacerlo seguro también.

3.6.2. Instalando svnserve

1. Obtenga la última versión de Subversion desde *http://subversion.apache.org/getting.html*. Alternativamente obtenga un instalador pre-empaquetado desde CollabNet en *http://www.collab.net/downloads/subversion*. Este instalador pondrá svnserve como un servicio de Windows, y también incluye algunas de las herramientas que necesitará si va a utilizar SASL para la seguridad.

2. Si ya tiene una versión de Subversion instalada, y svnserve se está ejecutando, necesitará pararlo antes de continuar.

3. Ejecute el instalador de Subversion. Si ejecuta el instalador en su servidor (recomendado) puede saltarse el paso 4.

4. Abra el explorador de Windows, vaya al directorio de instalación de Subversion (normalmente`C:\Archivos de programa\Subversion`) y en el directorio `bin`, localice los ficheros `svnserve.exe`, `intl3_svn.dll`, `libapr.dll`, `libapriconv.dll`, `libapriutil.dll`, `libdb*.dll`, `libeay32.dll` y `ssleay32.dll` - copie estos ficheros, o simplemente copie todo el contenido del directorio `bin`, a un directorio de su servidor, por ejemplo `c:\svnserve`

3.6.3. Ejecutando svnserve

Ahora que svnserve está instalado, necesitará ejecutarlo en su servidor. La forma más sencilla es ejecutar lo siguiente desde una ventana DOS o bien crear un acceso directo de Windows:

```
svnserve.exe --daemon
```

svnserve ahora se iniciará esperando peticiones entrantes en el puerto 3690. La opción --daemon le dice a svnserve que se ejecute como un servicio, por lo que continuará ejecutándose hasta que manualmente se le mande terminar.

Si aún no ha creado un repositorio, siga las instrucciones dadas en la instalación del servidor basado en Apache Sección 3.7.4, “Configuración”.

Para comprobar que svnserve está funcionando, utilice TortoiseSVN → Navegador para ver un repositorio.

Asumiendo que su repositorio está en `c:\repos\TestRepo`, y que su servidor se llama `localhost`, introduzca:

```
svn://localhost/repos/TestRepo
```

cuando le pregunte el navegador de repositorios.

También puede incrementar la seguridad y ahorrar tiempo introduciendo URLs con svnserve utilizando la opción --root para cambiar la ruta raíz y restringir el acceso a un directorio concreto del servidor:

```
svnserve.exe --daemon --root unidad:\ruta\al\repositorio
```

Utilizando el test anterior como guía, se debería ejecutar svnserve así:

```
svnserve.exe --daemon --root c:\repos
```

Y en TortoiseSVN la URL del navegador de repositorios se acorta así:

```
svn://localhost/TestRepo
```

Tenga en cuenta que la opción --root también se necesita si su repositorio está en una unidad o partición diferente del lugar donde está svnserve en su servidor.

Svnserve podrá servir un número de repositorios arbitrario. Simplemente colóquelos en algún lugar bajo la carpeta raíz que acaba de definir, y acceda a ellos utilizando una URL relativa a esa raíz.

Aviso

No cree o acceda a un repositorio Berkeley DB en una unidad de red compartida. *No* puede existir en un sistema de archivos remoto. Ni siquiera si tiene la unidad de red mapeada a una letra de unidad. Si intenta usar Berkeley DB en una unidad de red compartida, los resultados son imprevisibles - puede ver desde el principio errores misteriosos, o pueden pasar meses antes de que descubra que su base de datos del repositorio está corrupta de una forma inimaginable.

3.6.3.1. Ejecutar svnserve como un servicio

Ejecutar svnserve como un usuario normalmente no es la mejor forma de hacerlo. Eso significa que siempre debe haber un usuario con la sesión iniciada en su servidor, y que debe acordarse de iniciarlo de nuevo tras cada reinicio. Una manera mejor es ejecutar svnserve como un servicio de Windows. Desde la versión 1.4 de Subversion, svnserve puede instalarse como un servicio de Windows nativo.

Para instalar svnserve como un servicio de Windows nativo, ejecute el siguiente comando todo en una línea para crear un servicio que se arranque automáticamente cuando se inicie Windows.

```
sc create svnserve binpath= "c:\svnserve\svnserve.exe --service
    --root c:\repos" displayname= "Subversion" depend= tcpip
    start= auto
```

Si cualquiera de las rutas incluye espacios, tendrá que utilizar comillas (escapadas) en la ruta, como en este ejemplo:

```
sc create svnserve binpath= "
    \"C:\Archivos de programa\Subversion\bin\svnserve.exe\"
    --service --root c:\repos" displayname= "Subversion"
    depend= tcpip start= auto
```

También puede añadir una descripción después de crear el servicio; esta descripción se mostrará en el Administrador de Servicios de Windows.

```
sc description svnserve "Servidor de Subversion (svnserve)"
```

Tenga en cuenta el formato no muy usual de la línea de comandos utilizado por `sc`. En los pares `key= value` no debe haber espacio entre la clave y el `=` pero sí debe haber un espacio antes del valor.

Sugerencia

Microsoft ahora recomienda que los servicios se ejecuten o con la cuenta de Servicio Local o con la cuenta de Servicio de Red. Tiene más información en *The Services and Service Accounts Security Planning Guide* [http://www.microsoft.com/technet/security/topics/serversecurity/serviceaccount/default.mspx]. Para crear el servicio bajo la cuenta de Servicio Local, añada lo siguiente al ejemplo anterior.

> `obj= "NT AUTHORITY\LocalService"`
>
> Tenga en cuenta que deberá proporcionar a la cuenta de Servicio Local los permisos apropiados tanto para Subversion como para sus repositorios, y también a cualquier aplicación que utilicen los scripts gancho. El grupo por defecto para esto se llama "LOCAL SERVICE".

Una vez que haya instalado el servicio, necesitará ir al administrador de servicios para arrancarlo (sólo por esta vez; arrancará automáticamente cuando se reinicie el servidor).

Para obtener información más detallada, refiérase a *Soporte de servicio de Windows para Svnserve* [http://svn.collab.net/repos/svn/trunk/notes/windows-service.txt].

Si ha instalado una versión anterior de svnserve utilizando el programa `SVNService`, y ahora quiere utilizar el soporte nativo, deberá desregistrar el programa `SVNService` como servicio (¡recuerde parar el servicio antes!). Símplemente utilice el comando

```
svnservice -remove
```

para eliminar la entrada del registro del servicio.

3.6.4. Autentificación básica con svnserve

La configuración por defecto de svnserve proporciona acceso anónimo de sólo-lectura. Esto significa que puede utilizar una URL de tipo `svn://` para obtener y actualizar, o utilizar el navegador de repositorios en TortoiseSVN para ver el repositorio, pero no podrá confirmar ningún cambio.

Para permitir acceso de escritura en un repositorio, necesitará editar el fichero `conf/svnserve.conf` en el directorio de su repositorio. Este fichero controla la configuración del servicio svnserve, y también contiene información útil.

Puede habilitar el acceso anónimo para escritura simplemente poniendo:

```
[general]
anon-access = write
```

Sin embargo, no sabrá quién ha hecho cambios en el repositorio, dado que la propiedad `svn:author` estará vacía. Tampoco podrá controlar quién puede hacer cambios en el repositorio. ¡Esta es una configuración algo arriesgada!

Una forma de conseguir esto es crear una base de datos de contraseñas:

```
[general]
anon-access = none
auth-access = write
password-db = ficherodeusuarios
```

Donde `ficherodeusuarios` es un fichero que existe en el mismo directorio que `svnserve.conf`. Este fichero también puede estar en cualquier otro sitio de su sistema de ficheros (útil cuando tiene múltiples repositorios que necesitan los mismos derechos de acceso) y puede ser referenciado utilizando una ruta absoluta, o una ruta relativa del directorio `conf`. Si incluye una ruta, debe estar escrita `/a/la/forma/de/unix`. No funcionará si utiliza \ o letras de unidades. El `ficherodeusuarios` debería tener una estructura como ésta:

```
[users]
usuario = contraseña
...
```

Este ejemplo denegaría cualquier acceso a los usuarios no autentificados (anónimos), y daría acceso de lectura-escritura a los usuarios listados en `ficherodeusuarios`.

Sugerencia

Si mantiene múltiples repositorios utilizando la misma base de datos de contraseñas, la utilización de un dominio de autentificación le hará la vida más fácil a los usuarios, dado que TortoiseSVN puede almacenar en caché sus credenciales para que sólo tenga que introducirlas una vez. Se puede encontrar más información en el libro de Subversion, específicamente en las secciones *Crear un fichero 'users' y un dominio* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.svnserve.html#svn.serverconfig.svnserve.auth.users] y *Caché de credenciales de cliente* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.netmodel.html#svn.serverconfig.netmodel.credcache]

3.6.5. Mejor seguridad con SASL

3.6.5.1. ¿Qué es SASL?

La Capa de seguridad y autenticación simple Cyrus (Cyrus Simple Authentication and Security Layer) es un software de código abierto escrito por la Universidad de Carnegie Mellon. Añade capacidades genéricas de autenticación y encriptación a cualquier protocolo de red, y desde Subversion 1.5 y posteriores, tanto el servidor svnserve como el cliente TortoiseSVN saben cómo hacer uso de esta biblioteca.

Para una discusión más completa de las opciones disponibles, debería leer el libro de Subversion en la sección *Utilizando svnserve con SASL* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.svnserve.html#svn.serverconfig.svnserve.sasl]. Si simplemente está buscando una forma sencilla para poner autenticación y encriptación en un servidor Windows, para que su repositorio pueda accederse de forma segura a través del gran y malvado Internet, siga leyendo.

3.6.5.2. Autentificación SASL

Para activar los mecanismos específicos SASL en el servidor, necesitará hacer tres cosas. Primero, cree una sección `[sasl]` en el fichero `svnserve.conf` de su repositorio, con este par de clave-valor:

```
use-sasl = true
```

En segundo lugar, cree un fichero llamado `svn.conf` en un lugar conveniente - típicamente en el directorio donde está instalado Subversion.

En tercer lugar, cree dos nuevas entradas de registro para indicar a SASL dónde encontrar las cosas. Cree una clave de registro llamada `[HKEY_LOCAL_MACHINE\SOFTWARE\Carnegie Mellon \Project Cyrus\SASL Library]` y ponga dos nuevos valores de cadena dentro: `SearchPath` establecido a la carpeta que contiene los plug-ins `sasl*.dll` (normalmente en la carpeta de instalación de Subversion), y `ConfFile` establecido a la carpeta que contiene el fichero `svn.conf`. Si ha utilizado el instalador CollabNet, estas claves de registro ya habrán sido creadas por usted.

Edite el fichero `svn.conf` para que contenga lo siguiente:

```
pwcheck_method: auxprop
auxprop_plugin: sasldb
mech_list: DIGEST-MD5
sasldb_path: C:\TortoiseSVN\sasldb
```

La última línea muestra el lugar de la base de datos de autenticación, que es un fichero llamado `sasldb`. Este puede estar en cualquier sitio, pero una elección conveniente es la carpeta padre del repositorio. Asegúrese de que el servicio svnserve tiene acceso de lectura sobre este fichero.

Si svnserve ya está ejecutándose, necesitará reiniciarlo para asegurarse de que lee la configuración actualizada.

Ahora que todo está preparado, todo lo que necesita es crear algunos usuarios y contraseñas. Para hacerlo, necesitará el programa `saslpasswd2`. Si ha usado el instalador CollabNet, ese programa estará en el diretorio de instalación. Utilice un comando como este:

```
saslpasswd2 -c -f C:\TortoiseSVN\sasldb -u realm username
```

La opción `-f` establece el lugar de la base de datos, `realm` debe ser el mismo valor que ha definido en el fichero `svnserve.conf` de su repositorio, y el nombre de usuario es exactamente lo que se supone que es. Tenga en cuenta que el realm no admite espacios en blanco.

Puede listar los nombres de usuarios almacenados en la base de datos utilizando el programa `sasldblistusers2`.

3.6.5.3. Encriptación SASL

Para habilitar o deshabilitar diferentes niveles de encriptación, puede poner dos valores en el fichero `svnserve.conf` de su repositorio:

```
[sasl]
use-sasl = true
min-encryption = 128
max-encryption = 256
```

Las variables `min-encryption` y `max-encryption` controlan el nivel de encriptación demandado por el servidor. Para deshabilitar la encriptación completamente, establezca ambos valores a 0. Para habilitar la suma simple de comprobación (por ejemplo, previene alteraciones y garantiza la integridad de los datos sin encriptación), establezca ambos valores a 1. Si desea permitir (pero no requerir) encriptación, establezca el valor mínimo a 0 y el valor máximo a alguna longitud de bits. Para requerir encriptación incondicional, establezca ambos valores a números mayores de 1. En nuestro ejemplo anterior, requerimos a los clientes encriptación de al menos 128-bits, pero no más de 256-bits.

3.6.6. Autentificación con svn+ssh

Otra forma de autentificar a los usuarios con un servidor basado en svnserve es utilizar un shell seguro (SSH) para encapsular las peticiones a través suyo. No es tan sencillo de preparar como SASL, pero puede ser útil en algunos casos.

Con esta aproximación, svnserve no se ejecuta como un servicio, en cambio, el shell seguro inicia svnserve por usted, ejecutándolo como el usuario autentificado SSH. Para habilitar esto, necesita un servicio de shell seguro en su servidor.

En Apéndice G, *Asegurando Svnserve utilizando SSH* tiene un método básico para preparar su servidor. Puede encontrar otros temas SSH en el FAQ buscando “SSH”.

Puede encontrar más información sobre svnserve en el libro *Control de versiones con Subversion* [http://svnbook.red-bean.com].

3.6.7. Autorización basada en rutas con svnserve

Empezando con Subversion 1.3, svnserve soporta el mismo esquema de autorización basada en rutas que está disponible en el módulo `mod_authz_svn` de Apache. Necesita editar el fichero `conf/svnserve.conf` dentro del directorio de su repositorio y añadir una línea refiriéndose a su fichero de autorización.

```
[general]
authz-db = authz
```

Aquí, `authz` es un fichero que debe existir y que define los permisos de acceso. Puede utilizar un fichero separado por cada repositorio, o utilizar el mismo fichero para varios repositorios. Si desea una descripción del formato del fichero, lea Sección 3.7.6, “Autorización basada en rutas”.

3.7. Servidor basado en Apache

3.7.1. Introducción

La configuración más flexible de todas las instalaciones de servidor posibles para Subversion es la que se basa en Apache. Aunque es un poco más complicada de preparar, ofrece beneficios que otros servidores no pueden dar:

WebDAV
: El servidor de Subversion basado en Apache utiliza el protocolo WebDAV que se utiliza por muchos otros programas. Por ejemplo, podría montar dicho repositorio como una “Carpeta web” en el explorador de Windows y luego acceder a ella como cualquier otra carpeta en su sistema de ficheros.

Navegando por el repositorio
: Puede apuntar su navegador a la URL del repositorio y navegar por sus contenidos sin tener un cliente de Subversion. Esto da acceso a sus datos a un mayor círculo de usuarios.

Autentificación
: Puede utilizar cualquier mecanismo de autentificación que Apache soporte, incluyendo SSPI y LDAP.

Seguridad
: Dado que Apache es muy estable y seguro, automáticamente obtendrá la misma seguridad para su repositorio. Esto incluye la encriptación SSL.

3.7.2. Instalando Apache

La primera cosa que necesita antes de instalar Apache es un ordenador con Windows 2000, Windows XP con SP1, Windows 2003, Vista o Server 2008.

Aviso

Por favor tenga en cuenta que utilizar Windows XP sin el Service Pack 1 corrompe datos de la red y por tanto ¡podría corromper su repositorio!

1. Descargue la última versión del servidor web Apache desde *http://httpd.apache.org/download.cgi*. Asegúrese de que descarga la versión 2.2.x - ¡las versiones 1.3.xx no servirán!

   El instalador MSI de Apache se puede encontrar haciendo click en `other files` (otros ficheros), y luego navegando a `binaries/win32`. Puede que quiera seleccionar el fichero MSI `apache-2.2.x-win32-x86-openssl-0.9.x.msi` (el que incluye OpenSSL).

2. Una vez que tenga el instalador de Apache2 puede hacer doble click en él y le guiará a través del proceso de instalación. Asegúrese de que ha introducido la URL del servidor correctamente (si no tiene un nombre DNS para su servidor introduzca la dirección IP). Es recomendable que instale Apache *para Todos los usuarios, en el Puerto 80, como un Servicio*. Nota: si ya tiene IIS u otro programa ejecutándose que escuche en el puerto 80 la instalación puede fallar. Si esto ocurre, vaya al directorio de Archivos de programa, `\Apache Group\Apache2\conf` y localice el fichero `httpd.conf`. Edite dicho fichero para cambiar `Listen 80` por un puerto libre, por ejemplo, `Listen 81`. Luego reinicie la instalación - esta vez debería terminar sin problemas.

3. Ahora compruebe si el servidor web Apache funciona correctamente apuntando desde su navegador web a la dirección `http://localhost/` - debería aparecer un sitio web preconfigurado.

Atención

Si decide instalar Apache como un servicio, queda avisado de que por defecto se ejecutará con la cuenta de sistema local. Sería una práctica más segura que creara una cuenta separada para que Apache se ejecutara bajo ella.

Asegúrese de que la cuenta en el servidor bajo la que se ejecuta Apache tenga una entrada explícita en la lista de control de acceso del directorio del repositorio (click con el botón derecho en el directorio | propiedades | seguridad), con control total. Si no lo hace así, los usuarios no podrán confirmar sus cambios.

Incluso si Apache se ejecuta como sistema local, aún así necesitará dicha entrada (que en este caso debería ser la cuenta SYSTEM).

Si Apache no tiene este permiso configurado, sus usuarios tendrán mensajes de error "Acceso denegado", que se mostrarán en el registro de errores de Apache como error 500.

3.7.3. Instalando Subversion

1. Descarge la última versión de los binarios de Subversion para Apache y Windows 32. Asegúrese de obtener la versión correcta para integrarla en su versión de Apache, porque si no obtendrá un oscuro mensaje de error cuando intente reiniciar. Si tiene Apache 2.2.x acuda a *http://subversion.tigris.org/servlets/ProjectDocumentList?folderID=8100*.

2. Ejecute el instalador de Subversion y siga las instrucciones. Si el instalador de Subversion reconoce que ha instalado Apache, habrá casi terminado. Si no puede encontrar un servidor de Apache entonces tendrá que realizar algunos pasos adicionales.

3. Utilizando el explorador de Windows, vaya al directorio de instalación de Subversion (normalmente `C:\Archivos de programa\Subversion`) y busque los ficheros `/httpd/mod_dav_svn.so` y `mod_authz_svn.so`. Copie estos ficheros al directorio de módulos de Apache (normalmente `C:\Archivos de programa\Apache Group\Apache2\modules`).

4. Copie el fichero `/bin/libdb*.dll` y `/bin/intl3_svn.dll` desde el directorio de instalación de Subversion al directorio bin de Apache.

5. Edite el fichero de configuración de Apache (normalmente `C:\Archivos de Programa\Apache Group\Apache2\conf\httpd.conf`) con un editor de texto como el Bloc de Notas y haga los siguientes cambios:

   Descomente (quitando la marca '#') las siguientes líneas:

   ```
   #LoadModule dav_fs_module modules/mod_dav_fs.so
   #LoadModule dav_module modules/mod_dav.so
   ```

   Añada las dos líneas siguientes al final de la sección `LoadModule`.

   ```
   LoadModule dav_svn_module modules/mod_dav_svn.so
   LoadModule authz_svn_module modules/mod_authz_svn.so
   ```

3.7.4. Configuración

Ahora ya ha preparado Apache y Subversion, pero Apache aún no sabe cómo manejar los clientes de Subversion como TortoiseSVN. Para que Apache sepa qué URL debe utilizarse para los repositorios de Subversion debe editar el fichero de configuración de Apache (normalmente está en `C:\Archivos`

de programa\Apache Group\Apache2\conf\httpd.conf) con cualquier editor de texto que desee (por ejemplo, el Bloc de notas):

1. Al final del fichero de configuración, añada las siguientes líneas:

```
<Location /svn>
  DAV svn
  SVNListParentPath on
  SVNParentPath D:\SVN
  #SVNIndexXSLT "/svnindex.xsl"
  AuthType Basic
  AuthName "Repositorios de Subversion"
  AuthUserFile passwd
  #AuthzSVNAccessFile svnaccessfile
  Require valid-user
</Location>
```

   Esto configura el Apache de forma que todos sus repositorios de Subversion están físicamente localizados bajo `D:\SVN`. Los respositorios se sirven al mundo exterior desde la URL: `http://MiServidor/svn/` . El acceso está restringido a los usuarios/contraseñas listados en el fichero `passwd`

2. Para crear el fichero `passwd`, abra el Símbolo del sistema o la línea de comandos (ventana DOS) de nuevo, cambie a la carpeta `Apache2` (normalmente `C:\Archivos de programa\Apache Group\Apache2`) y cree el fichero mediante

```
bin\htpasswd -c passwd <nombreusuario>
```

   Esto creará un nuevo fichero con el nombre `passwd` que se utilizará para la autentificación. Se pueden crear usuarios adicionales con

```
bin\htpasswd passwd <nombreusuario>
```

3. Reinice el servicio de Apache de nuevo.

4. Apunte su navegador a `http://MiServidor/svn/MiNuevoRepositorio` (donde `MiNuevoRepositorio` es el nombre del repositorio de Subversion que creó antes). Si todo ha ido bien debería ver una ventana preguntando por un usuario y una contraseña, y luego podrá ver los contenidos de su repositorio.

Una explicación breve sobre lo que acaba de introducir:

Configuración	Explicación
<Location /svn>	significa que los repositorios de Subversion están disponibles en la URL `http://MiServidor/svn/`
DAV svn	le dice a Apache qué módulo será responsable de servir esa URL - en este caso, el módulo de Subversion.
SVNListParentPath on	Para Subversion 1.3 y superiores, esta directiva habilita el listado de todos los repositorios disponibles bajo `SVNParentPath`.
SVNParentPath D:\SVN	le dice a Subversion que busque repositorios bajo `D:\SVN`
SVNIndexXSLT "/svnindex.xsl"	Utilizado para mejorar la visualización desde un navegador de web.

Configuración	Explicación
AuthType Basic	se utiliza para activar la autentificación básica, es decir, Usuario/contraseña
AuthName "Subversion repositories"	se utiliza cuando le aparezca un diálogo de autentificación al usuario como información para decirle para qué se necesita su autentificación
AuthUserFile passwd	especifica qué fichero de contraseñas se utiliza para la autentificación
AuthzSVNAccessFile	lugar del fichero de Acceso para las rutas dentro del repositorio de Subversion
Require valid-user	especifica que sólo los usuarios que hayan introducido un par usuario/contraseña válido podrán acceder a la URL

Tabla 3.1. Configuración de `httpd.conf` de Apache

Pero eso es sólo un ejemplo. Hay muchas, muchas más posibilidades de lo que puede hacer con el servidor web Apache.

- Si desea que su repositorio tenga acceso de lectura para todo el mundo pero el acceso de escritura sólo para usuarios específicos, puede cambiar la línea

  ```
  Require valid-user
  ```

  por

  ```
  <LimitExcept GET PROPFIND OPTIONS REPORT>
  Require valid-user
  </LimitExcept>
  ```

- Utilizando un fichero `passwd` se limita y se otorga acceso a todos sus repositorios como si fueran uno solo. Si desea más control sobre qué usuarios tienen acceso a cada carpeta dentro de un repositorio, puede descomentar la línea

  ```
  #AuthzSVNAccessFile svnaccessfile
  ```

  y crear un fichero de acceso de Subversion. Apache se asegurará que sólo los usuarios válidos pueden acceder su ruta `/svn`, y luego pasará el nombre de usuario al módulo de Subversion `AuthzSVNAccessFile` para que pueda forzar un acceso más controlado basado en las reglas que se especifican en el fichero de acceso de Subversion. Tenga en cuenta que las rutas se especifican o bien como `repos:ruta` o simplemente `ruta`. Si no especifica un repositorio en concreto, la regla de acceso se aplicará a todos los repositorios bajo `SVNParentPath`. El formato del fichero de política de autorización utilizado por `mod_authz_svn` se describe en Sección 3.7.6, “Autorización basada en rutas”

- Para hacer que la visualización del repositorio con un navegador web sea 'más bonita', descomente la línea

  ```
  #SVNIndexXSLT "/svnindex.xsl"
  ```

  y ponga los ficheros `svnindex.xsl`, `svnindex.css` y `menucheckout.ico` en su directorio raíz de documentos (normalmente `C:/Archivos de programa/Apache Group/Apache2/htdocs`). El directorio está establecido con la directiva `DocumentRoot` en su fichero de configuración de Apache.

Puede obtener estos tres ficheros directamente desde nuestro repositorio de código fuente en *http://tortoisesvn.googlecode.com/svn/trunk/contrib/svnindex*. (Sección 3, "¡TortoiseSVN es gratis!" le explica cómo acceder al repositorio de código de TortoiseSVN).

El fichero XSL del repositorio de TortoiseSVN tiene una característica destacada: si navega el repositorio con su navegador web, todas las carpetas de su repositorio mostrarán un icono a la derecha. Si pulsa sobre dicho icono, se iniciará el diálogo de obtener de TortoiseSVN para esa URL.

3.7.5. Múltiples repositorios

Si ha utilizado la directiva `SVNParentPath` no necesita cambiar el fichero de configuración de Apache cada vez que añada un nuevo repositorio de Subversion. Simplemente cree el nuevo repositorio bajo la misma ruta que el primer repositorio, ¡y ya está! En mi compañía, tengo acceso directo a esa carpeta en concreto utilizando SMB (el acceso normal de ficheros de Windows). Por lo que simplemente creo una nueva carpeta allí, ejecuto el comando de TortoiseSVN TortoiseSVN → Crear repositorio aquí... y ya hay un nuevo proyecto que tiene su hogar...

Si está utilizando Subversion 1.3 o posterior, puede utilizar la directiva `SVNListParentPath on` para permitir que Apache produzca un listado de todos los proyectos disponibles si apunta su navegador a la ruta raíz en vez de a un repositorio en concreto.

3.7.6. Autorización basada en rutas

El módulo `mod_authz_svn` le permite un control detallado de los permisos de acceso basado en nombres de usuarios y rutas de repositorio. Esto está disponible si utiliza el servidor de Apache, y desde Subversion 1.3 también está disponible si utiliza svnserve.

Esto sería un fichero de ejemplo:

```
[groups]
admin = john, kate
devteam1 = john, rachel, sally
devteam2 = kate, peter, mark
docs = bob, jane, miguel
training = zak
# Regla de acceso por defecto para TODOS los repositorios
# Todo el mundo puede leer, los administradores pueden escribir,
# Donpe Ligro está exluído.
[/]
* = r
@admin = rw
donpeligro =
# Permitir a los desarrolladores acceso completo
# al repositorio de su proyecto
[proj1:/]
@devteam1 = rw
[proj2:/]
@devteam2 = rw
[bigproj:/]
@devteam1 = rw
@devteam2 = rw
trevor = rw
# Dar a los documentadores acceso de escritura
# a todas las carpetas de documentación
[/trunk/doc]
```

```
@docs = rw
# Dar a los becarios acceso de escritura
# sólo al repositorio de pruebas
[TrainingRepos:/]
@training = rw
```

Tenga en cuenta que comprobar cada ruta puede ser una operación costosa, particularmente en el caso del registro de revisiones. El servidor toma cada ruta cambiada en cada revisión y comprueba si se puede leer, lo que puede ser lento en revisiones que afecten a un gran número de ficheros.

La autentificación y la autorización son procesos separados. Si un usuario desea tener acceso a una ruta de un repositorio, tiene que cumplir *ambos* requisitos, los requerimientos usuales de autentificación y los requerimientos de autorización del fichero de acceso.

3.7.7. Autentificación con un dominio de Windows

Como habrá notado necesita introducir una entrada usuario/contraseña en el fichero `passwd` para cada usuario de forma separada. Y si (por razones de seguridad) quiere que sus usuarios cambien periódicamente sus contraseñas tendrá que hacer el cambio de forma manual.

Pero hay una solución a este problema - al menos si accede al repositorio desde dentro de una LAN con un controlador de dominio de Windows: `mod_auth_sspi`!

El módulo original de SSPI fue ofrecido por Syneapps incluyendo el código fuente. Pero su desarrollo se detuvo. No se desespere, la comunidad lo ha retomado y mejorado. Tiene un nuevo hogar en *SourceForge* [http://sourceforge.net/projects/mod-auth-sspi/].

- Descargue el módulo que concuerde con su versión de Apache, y copie el fichero `mod_auth_sspi.so` en la carpeta de módulos de Apache.
- Edite el fichero de configuración de Apache: añada la línea

  ```
  LoadModule sspi_auth_module modules/mod_auth_sspi.so
  ```

  a la sección `LoadModule`. Asegúrese de insertar esta línea *antes* de la línea

  ```
  LoadModule auth_module modules/mod_auth.so
  ```

- Para que el localizador de Subversion utilice este tipo de autentificación tiene que cambiar la línea

  ```
  AuthType Basic
  ```

  por

  ```
  AuthType SSPI
  ```

  y también deberá añadir

  ```
  SSPIAuth On
  SSPIAuthoritative On
  SSPIDomain <domaincontroller>
  SSPIOmitDomain on
  SSPIUsernameCase lower
  SSPIPerRequestAuth on
  SSPIOfferBasic On
  ```

dentro del bloque `<Location /svn>` Si no tiene un controlador de dominio, deje el nombre del controlador de cominio como `<domaincontroller>`

Tenga en cuenta que si se autentifica utilizando SSPI, ya no necesitará la línea `AuthUserFile` para definir un fichero de contraseñas. En su lugar, Apache autentifica su usuario y contraseña contra su dominio de Windows. También necesitará actualizar la lista de usuarios en su `svnaccessfile` para referirse a `DOMINIO\usuario`.

Importante

La autenticación SSPI sólo está habilitada para conexiones seguras por SSL (https). Si únicamente está utilizando conexiones http normales a su servidor, no funcionará.

Para habilitar SSL en su servidor, vea el capítulo: Sección 3.7.9, “Asegurando el servidor con SSL”

Sugerencia

Los ficheros `AuthzSVNAccessFile` de Subversion distinguen mayúsculas y minúsculas respecto a los nombres de usuario (`JuanPerez` no es lo mismo que `juanperez`).

En el mundo de Microsoft, los dominios y nombres de usuarios de Windows no distinguen mayúsculas y minúsculas. Incluso así, a algunos administradores de redes les gusta crear las cuentas de usuario en CamelCase (por ejemplo, `JuanPerez`).

La diferencia puede morderle cuando utilice la autentificación SSPI ya que el dominio de Windows y los nombres de usuario se pasan a Subversion exactamente como los haya tecleado el usuario en la ventana. Internet Explorer a menudo pasa el nombre de usuario a Apache automáticamente utilizando el formato con el que se creó la cuenta.

El resultado final es que puede necesitar al menos dos entradas en su fichero `AuthzSVNAccessFile` para cada usuario -- una entrada en minúsculas y una entrada con el formato que Internet Explorer pasa a Apache. También tendrá que convencer a sus usuarios para que escriban sus credenciales en minúsculas cuando accedan a los repositorios utilizando TortoiseSVN.

Los registros de error y de acceso de Apache son su mejor aliado para descifrar problemas como estos, ya que le ayudarán a determinar la cadena de nombre de usuario pasada al módulo `AuthzSVNAccessFile` de Subversion. Puede que necesite experimentar con el formato exacto de la cadena del usuario en el `svnaccessfile` (por ejemplo, `DOMINIO\usuario` en vez de `DOMINIO//usuario`) para conseguir que todo funcione.

3.7.8. Múltiples orígenes de autentificación

También es posible tener más de un origen de autentificación para su repositorio de Subversion. Para conseguirlo, debe hacer que cada tipo de autentificación sea no-autoritario, para que Apache compruebe múltiples orígenes buscando un par usuario/contraseña que concuerden.

Un escenario común es utilizar tanto la autentificación de dominio de Windows como un fichero `passwd`, para que pueda dar acceso a SVN a usuarios que no tienen usuario en el dominio de Windows.

- Para habilitar tanto la autentificación de dominio de Windows como el fichero `passwd`, añada las siguientes entradas en el bloque `<Location>` de su fichero de configuración de Apache:

```
  AuthBasicAthoritative Off
  SSPIAuthoritative Off
```

Aquí tiene un ejemplo de la configuración completa de Apache para combinar la autentificación de dominio de Windows y el fichero `passwd`:

```
<Location /svn>
  DAV svn
  SVNListParentPath on
  SVNParentPath D:\SVN

  AuthName "Repositorios de Subversion"
  AuthzSVNAccessFile svnaccessfile.txt

# Usuarios de Dominio NT.
  AuthType SSPI
  SSPIAuth On
  SSPIAuthoritative Off
  SSPIDomain <domaincontroller>
  SSPIOfferBasic On

# Usuarios htpasswd.
  AuthType Basic
  AuthBasicAuthoritative Off
  AuthUserFile passwd

  Require valid-user
</Location>
```

3.7.9. Asegurando el servidor con SSL

Incluso aunque Apache 2.2.x tiene soporte para OpenSSL, no está activado por defecto. Necesitará activarlo manualmente.

1. En el fichero de configuración de Apache, descomente las líneas:

   ```
   #LoadModule ssl_module modules/mod_ssl.so
   ```

   y al final

   ```
   #Include conf/extra/httpd-ssl.conf
   ```

   y luego cambie la línea (en una única línea)

   ```
   SSLMutex "file:C:/Archivos de programa/Apache Software Foundation/\
   Apache2.2/logs/ssl_mutex"
   ```

   a

   ```
   SSLMutex default
   ```

2. Lo siguiente que necesita es crear un certificado SSL. Para hacer esto, abra un símbolo del sistema (ventana DOS) y vaya a la carpeta de Apache (por ejemplo, `C:\Archivos de programa \Apache Group\Apache2`) y escriba el siguiente comando:

```
bin\openssl req -config conf\openssl.cnf -new -out my-server.csr
```

Se le preguntará una contraseña. Por favor no utilice palabras sencillas, sino frases completas, por ejemplo, unos versos de un poema. Cuanto más larga sea la frase, mejor. También tendrá que introducir la URL de su servidor. Todas las demás preguntas son opcionales pero le recomendamos que las rellene también.

Normalmente el fichero `privkey.pem` se crea automáticamente, pero si no es así, necesitará teclear este comando para generarlo:

```
bin\openssl genrsa -out privkey.pem 2048
```

Ahora escriba los comandos

```
bin\openssl rsa -in conf\privkey.pem -out conf\server.key
```

y (en una única línea)

```
bin\openssl req -new -key conf\server.key -out conf\server.csr \
-config conf\openssl.cnf
```

y luego (en una única línea)

```
bin\openssl x509 -in conf\server.csr -out conf\server.crt
                 -req -signkey conf\server.key -days 4000
```

Esto creará un certificado que expirará en 4000 días. Y finalmente, introduzca (también en una única línea):

```
bin\openssl x509 -in conf\server.cert -out conf\server.der.crt
                 -outform DER
```

Estos comandos crearán algunos ficheros en la carpeta `conf` de Apache (`server.der.crt`, `server.csr`, `server.key`, `.rnd`, `privkey.pem`, `server.cert`).

3. Reinicie el servicio de Apache.
4. Apunte su navegador a `https://nombredelservidor/svn/project` ...

SSL e Internet Explorer

Si está asegurando su servidor con SSL y utiliza la autentificación contra un dominio de Windows se encontrará que la navegación de repositorios con el Internet Explorer ya no funcionará. No se preocupe - es sólo que Internet Explorer no se puede autentificar. Los demás navegadores no tienen ese problema y tanto TortoiseSVN como cualquier otro cliente de Subversion todavía podrán autentificarse.

Si todavía quiere utilizar IE para navegar en el repositorio, puede:

- Defina una directiva separada `<Location /ruta>` en el fichero de configuración de Apache, y añada `SSPIBasicPreferred On`. Esto permitirá que IE se autentifique de nuevo, pero los demás navegadores y Subversion no se podrán autentificar contra esa ruta.

- Ofrezca también la navegación sin autentificación encriptada (sin SSL). Extrañamente, IE no tiene ningún problema para autentificarse si la conexión no está asegurada con SSL.
- En la configuración "estándar" a menudo aparecen la siguiente sentencia en el host virtual SSL de Apache:

```
SetEnvIf User-Agent ".*MSIE.*" \
             nokeepalive ssl-unclean-shutdown \
             downgrade-1.0 force-response-1.0
```

  Hay (¿había?) buenas razones para esta configuración, lea *http://www.modssl.org/docs/2.8/ssl_faq.html#ToC49* Pero si desea usar autentificación NTLM deberá utilizar `keepalive`. Si descomenta toda la sentencia "SetEnvIf" debería ser capaz de autentificar IE con la autentificación de Windows sobre SSL contra un Apache sobre Win32 con el módulo incluído `mod_auth_sspi`.

Forzando el acceso SSL

Cuando haya preparado SSL para hacer su repositorio más seguro, puede que desee deshabilitar el acceso normal via no-SSL (http) y sólo permitir el acceso https. Para hacer esto, debería añadir otra directiva al bloque `<Location>` de Subversion: `SSLRequireSSL`.

Un bloque `<Location>` de ejemplo se parecería a éste:

```
<Location /svn>
  DAV svn
  SVNParentPath D:\SVN
  SSLRequireSSL
  AuthType Basic
  AuthName "Repositorios de Subversion"
  AuthUserFile passwd
  #AuthzSVNAccessFile svnaccessfile
  Require valid-user
</Location>
```

3.7.10. Utilizando certificados de cliente con hosts SSL virtuales

Enviado a la lista de correo de TortoiseSVN por Nigel Green. ¡Gracias!

En algunas configuraciones de servidor puede que necesite configurar un único servidor que contenga 2 hosts SSL virtuales: el primero para acceso web público, sin requerimientos de un certificado de cliente; el segundo, seguro requiriendo un certificado de cliente, y ejecutando un servidor Subversion.

Al añadir una directiva `SSLVerifyClient Optional` en la sección *por-servidor* de la configuración de Apache (es decir, fuera de cualquier bloque `VirtualHost` y `Directory`), se fuerza a Apache a pedir un certificado de cliente en el saludo inicial SSL. Debido a un bug en `mod_ssl`, es esencial que el certificado se pida en este punto, ya que no funciona si la conexión SSL se re-negocia.

La solución es añadir la siguiente directiva en el directorio del host virtual que quiere bloquear para Subversion:

```
SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS"
```

Esta directiva da acceso al directorio sólo si se recibió y verificó correctamente un certificado de cliente.

Para resumir, las líneas relevantes de la configuración de Apache son:

```
SSLVerifyClient Optional

### Configuración del virtual host para el host PÚBLICO
### (sin necesidad de un certificado de cliente)

<VirtualHost 127.0.0.1:443>
  <Directory "rutaalaraizdeficherospublicos">
  </Directory>
</VirtualHost>

### Configuración del virtual host para SUBVERSION
### (necesita un certificado de cliente)
<VirtualHost 127.0.0.1:443>
  <Directory "ruta a la raiz del host de subversion">
    SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS"
  </Directory>

  <Location /svn>
    DAV svn
    SVNParentPath /rutaalrepositorio
  </Location>
</VirtualHost>
```

Capítulo 4. Guía de uso diario

Este documento describe el uso diario del cliente TortoiseSVN. *No* es una introducción a los sistemas de control de versiones, y *no* es una introducción a Subversion (SVN). Es más como un lugar donde puede venir cuando sepa qué quiere hacer, pero no recuerde exactamente cómo hacerlo.

Si necesita una introducción al control de versiones con Subversion, le recomendamos que se lea el fantástico libro: *Control de versiones con Subversion* [http://svnbook.red-bean.com/].

Este documento es también un trabajo en progreso, igual que lo son TortoiseSVN y Subversion. Si encuentra algún error, por favor háganoslo saber en la lista de correo para que podamos actualizar la documentación. Algunas de las capturas de pantalla en la Guía de Uso Diario (GUD) puede que no reflejen el estado actual del software. Le rogamos nos disculpe. Trabajamos en TortoiseSVN en nuestro tiempo libre.

Para aprovechar al máximo la Guía de uso diario:

- Debe tener ya instalado TortoiseSVN.
- Debe estar familiarizado con los sistemas de control de versiones.
- Debe conocer las bases de Subversion.
- Debe haber preparado un servidor y/o tener acceso a un repositorio de Subversion.

4.1. Empezando

4.1.1. Iconos sobreimpresionados

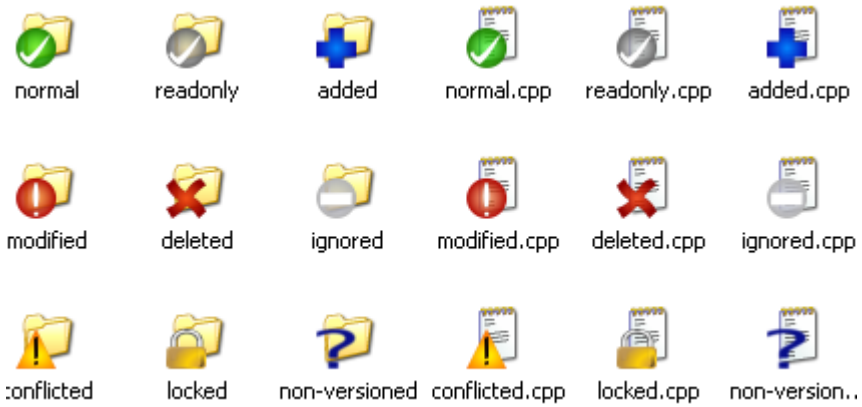


Figura 4.1. Explorador mostrando iconos sobreimpresionados

Una de las funciones más visibles de TortoiseSVN son los iconos sobreimpresionados que aparecen en los ficheros de su copia de trabajo. Estos le muestran de un vistazo qué ficheros han sido modificados. Lea Sección 4.7.1, “Iconos sobreimpresionados” para averiguar qué representa cada icono sobreimpresionado.

4.1.2. Menús contextuales

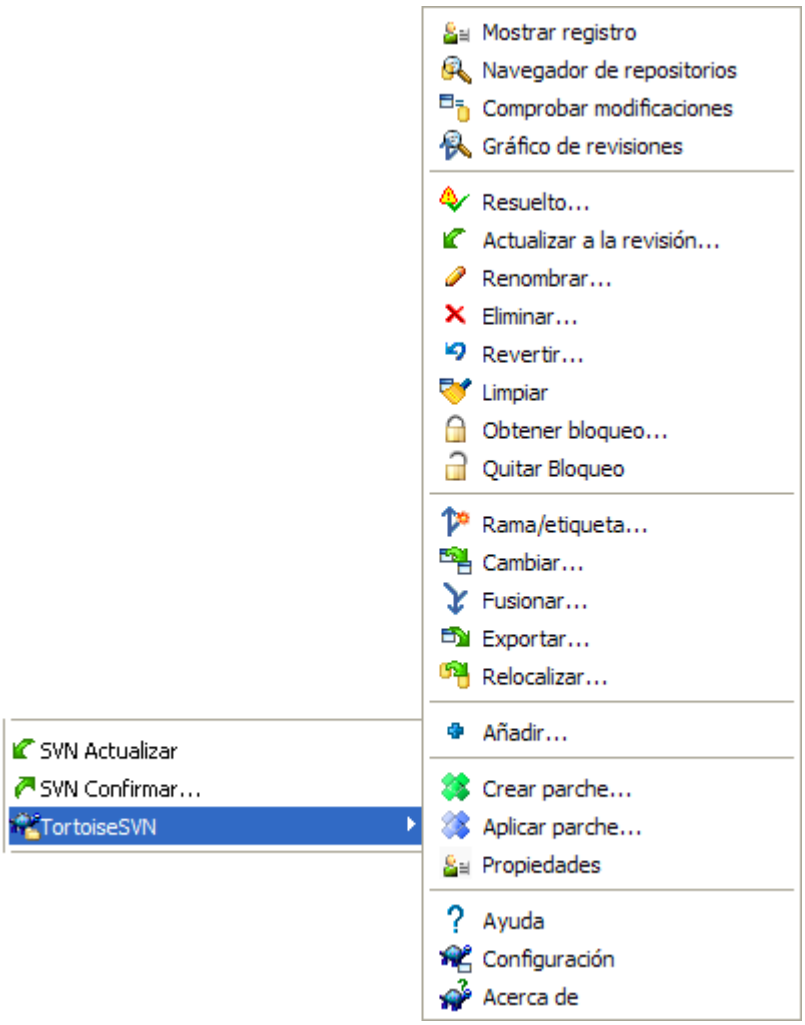


Figura 4.2. Menú contextual para un directorio bajo el control de versiones

Todos los comandos de TortoiseSVN se invocan desde el menú contextual del explorador de Windows. La mayoría se ven directamente, cuando hace click con el botón derecho en un fichero o una carpeta. Los comandos disponibles dependen de si el fichero o la carpeta o su carpeta padre está bajo el control de versiones o no. También puede ver el menú de TortoiseSVN como parte del menú archivo del explorador.

Sugerencia

Algunos comandos que se utilizan muy raramente sólo están disponibles en el menú contextual extendido. Para mostrar el menú contextual extendido, mantenga pulsada la tecla **Mayús** mientras hace click con el boton derecho.

En algunos casos puede ver varias entradas de TortoiseSVN. ¡Esto no es un error!

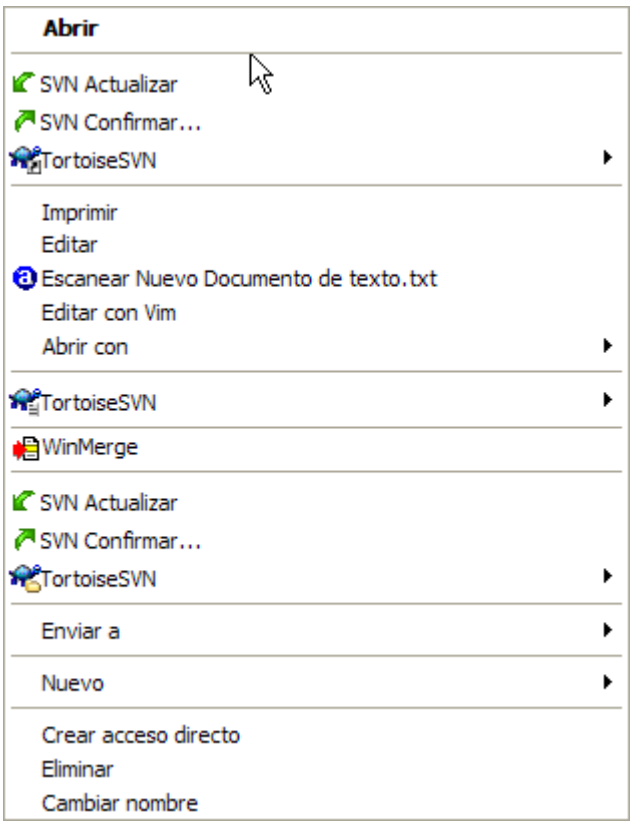


Figura 4.3. Menú archivo del explorador para un acceso directo en una carpeta versionada

Este ejemplo es para un acceso directo sin versionar dentro de una carpeta versionada, y en el menú de archivo del Explorador hay *tres* entradas para TortoiseSVN. Una es para la carpeta, otra para el acceso directo en sí mismo, y otra para el objeto al que apunta el acceso directo. Para ayudarle a distinguir entre ellos, los iconos tienen un indicador en la esquina inferior derecha para mostrarle que la entrada del menú es para un fichero, una carpeta, un acceso directo o para múltiples ítems seleccionados.

Si está utilizando Windows 2000 verá que los menús contextuales se muestran sólo como texto, sin los iconos de menú mostrados arriba. Sabemos que esto funcionaba en las versiones anteriores, pero Microsoft ha cambiado la forma en la que los iconos de menú funcionan en Vista, y eso nos ha forzado a utilizar una forma para mostrarlos que desafortunadamente no funciona en Windows 2000.

4.1.3. Arrastrar y soltar

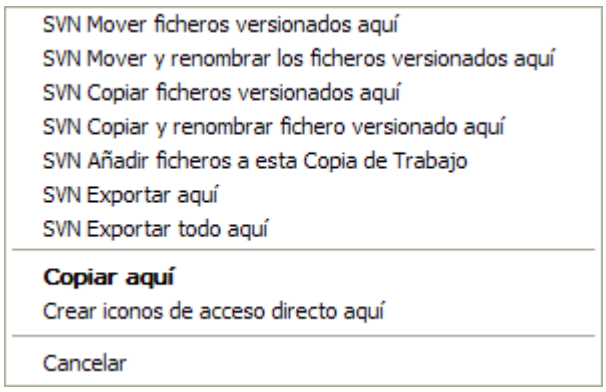


Figura 4.4. Menú de arrastre con el botón derecho para un directorio bajo el control de versiones

Otros comandos están disponibles como manejadores de arrastre, cuando arrastra con el botón derecho ficheros o carpetas a un nuevo destino dentro de copias de trabajo, o cuando arrastra con el botón derecho un fichero o una carpeta no versionados a un directorio que está bajo el control de versiones.

4.1.4. Atajos comunes

Algunas operaciones comunes tienen atajos de Windows bien conocidos, pero no aparecen en botones o en los menús. Si no puede averiguar cómo hacer algo obvio, como refrescar una vista, mire aquí.

F1
: La ayuda, por supuesto.

F5
: Refresca la vista actual. Este es quizás el comando de una tecla más útil. Por ejemplo... en el Explorador esto refresca los iconos sobreimpresionados en su copia de trabajo. En el diálogo de confirmación volverá a reescanear la copia de trabajo para ver qué puede necesitar ser confirmado. En el diálogo de Mostrar Registro contactará con el repositorio de nuevo buscando los cambios más recientes.

Ctrl-A
: Selecciona todo. Esto puede ser útil si obtiene un mensaje de error y quiere copiar y pegarlo en un email. Utilice Ctrl-A para seleccionar el mensaje de error y luego...

Ctrl-C
: ... copia el texto seleccionado.

4.1.5. Autentificación

Si el repositorio al que intenta acceder está protegido por contraseña, aparecerá un diálogo de autentificación.

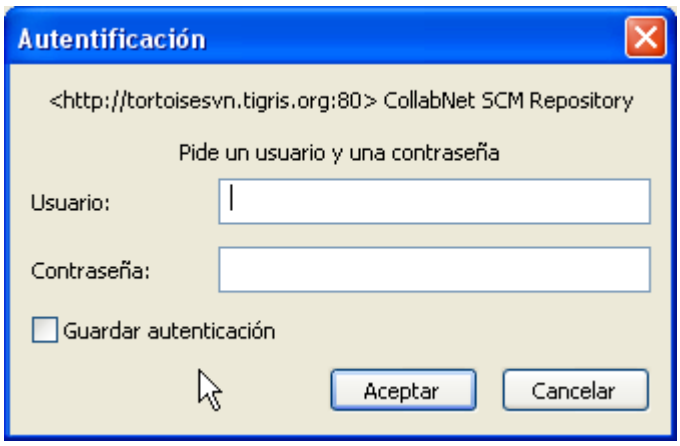


Figura 4.5. Diálogo de autentificación

Introduzca su usuario y contraseña. La casilla le permite que TortoiseSVN almacene las credenciales en el directorio por defecto de Subversion: `%APPDATA%\Subversion\auth`, en tres subdirectorios:

- `svn.simple` contiene las credenciales para la autentificación básica (usuario/contraseña).
- `svn.ssl.server` contiene los certificados SSL de servidor.
- `svn.username` contiene las credenciales para autentificación sólo por usuario (sin necesidad de contraseña).

Si desea eliminar la caché de autentificación para todos los servidores, puede hacerlo desde la página Datos Almacenados del diálogo de configuración de TortoiseSVN. Ese botón borrará todos los datos de autentificación cacheados de los directorios `auth` de Subversion, así como cualquier dato de autentificación almacenado en el registro por versiones anteriores de TortoiseSVN. Lea Sección 4.30.6, “Datos de configuración almacenados”.

Algunas personas quieren que se eliminen sus datos de autenticación cuando cierran su sesión de Windows, o cuando apagan el sistema. La forma de conseguir esto es utilizar un script de cierre para eliminar el directorio `%APPDATA%\Subversion\auth`, por ejemplo

```
@echo off
rmdir /s /q "%APPDATA%\Subversion\auth"
```

Puede encontrar una descripción sobre cómo instalar este tipo de scripts en *windows-help-central.com* [http://www.windows-help-central.com/windows-shutdown-script.html].

Para más información sobre cómo preparar su servidor para la autentificación y el control de acceso, vea Sección 3.5, “Accediendo al repositorio”

4.1.6. Maximizando ventanas

Muchos de los diálogos de TortoiseSVN tienen montones de información que mostrar, pero a menudo es más útil maximizar sólo la altura o sólo la anchura, mejor que maximizar para ocupar toda la pantalla. Como ayuda existen atajos para esto en el botón Maximizar. Utilice el botón central del ratón para maximizar verticalmente, y el botón derecho del ratón para maximizar horizontalmente.

4.2. Importando datos en un repositorio

4.2.1. Importar

Si está importando en un repositorio que ya tiene algunos proyectos, entonces la estructura del repositorio ya estará decidida. Si está importando datos a un nuevo repositorio entonces merece la pena tomar el tiempo para pensar en cómo debería organizarse. Lea Sección 3.1.5, “Organización del repositorio” para más información.

Esta sección describe el comando importar de Subversion, que fue diseñado para importar una jerarquía de directorios en el repositorio de una vez. Aunque funciona, tiene algunos inconvenientes:

- No hay forma de seleccionar los ficheros y carpetas a incluir, salvo si utiliza la configuración del patrón global de ignorar.
- La carpeta importada no se convierte en una copia de trabajo. Tiene que hacer una obtención para copiar los ficheros de nuevo desde el servidor.
- Es fácil importar en un nivel de carpetas erróneo en el repositorio.

Por estas razones le recomendamos que no utilice el comando importar en absoluto, y en cambio siga el método de dos pasos descrito en Sección 4.2.2, “Importar en el sitio”. Pero dado que ya está aquí, así es como funciona el comando importar básico ...

Antes de importar su proyecto en un repositorio debería:

1. Quitar todos los ficheros que no se necesitan para construir el proyecto (ficheros temporales, ficheros que se generan por un compilador como los *.obj, binarios compilados, ...)
2. Organizar los ficheros en carpetas y subcarpetas. Aunque es posible renombrar/mover los ficheros más tarde, ¡es muy recomendable que tenga la estructura del proyecto antes de importarlo!

Ahora seleccione la carpeta superior de la estructura de directorios del proyecto en el explorador de Windows, y haga click con el botón derecho para abrir el menú contextual. Seleccione el comando TortoiseSVN → Importar... y aparecerá un cuadro de diálogo:

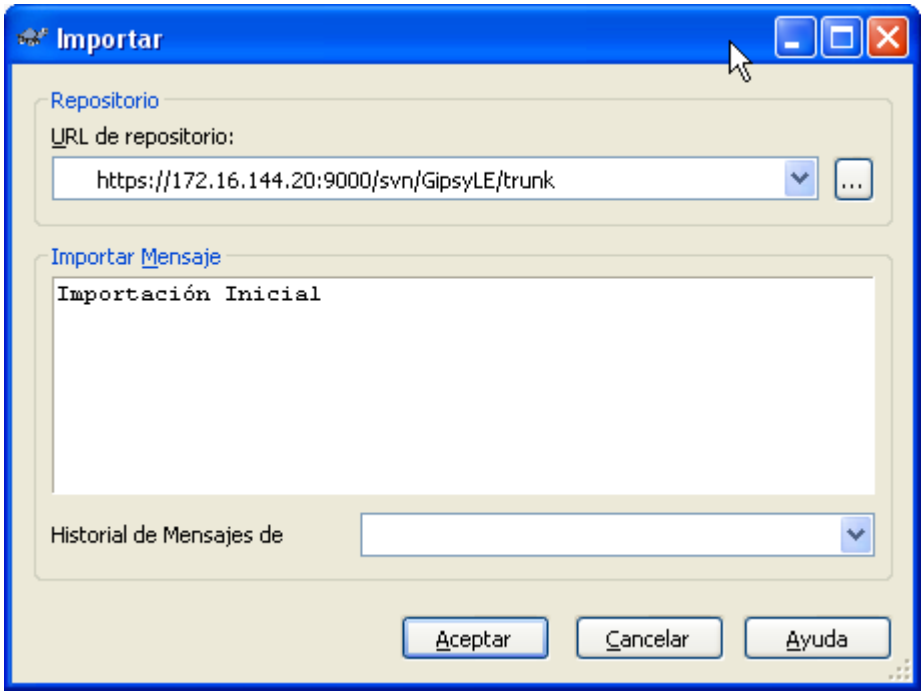


Figura 4.6. El diálogo Importar

En este diálogo tiene que introducir la URL del lugar del repositorio donde desea importar su proyecto. Es muy importante darse cuenta de que la carpeta local que está importando no aparece en sí misma en el repositorio, sólo su contenido. Por ejemplo, si tiene una estructura:

```
C:\Proyectos\Widget\source
C:\Proyectos\Widget\doc
C:\Proyectos\Widget\images
```

e importa `C:\Proyectos\Widget` en `http://mydomain.com/svn/trunk` entonces puede que se sorprenda al encontrar que sus subdirectorios van directos a `trunk` en vez de estar en un subdirectorio `Widget`. Necesita especificar el subdirectorio como parte de la URL, `http://mydomain.com/svn/trunk/Widget-X`. Tenga en cuenta que el comando importar automáticamente crea los subdirectorios en el repositorio si no existen.

El mensaje de importación se utiliza como un mensaje de registro.

Por defecto, los ficheros y carpetas que concuerden con los patrones globales de ignorar *no* se importan. Para cambiar este comportamiento, puede utilizar la casilla Incluir ficheros ignorados. Lea Sección 4.30.1, “Configuración general” para más información sobre cómo establecer un patrón global de ignorar.

Tan pronto como presione Aceptar, TortoiseSVN importa el árbol completo de directorios, incluyendo todos los ficheros, en el repositorio. El proyecto ahora está almacenado en el repositorio bajo el control de versiones. Por favor tenga en cuenta que la carpeta que ha importado ¡*NO* está bajo el control de versiones! Para obtener una *copia de trabajo* bajo el control de versiones necesita Obtener la versión que acaba de importar. O siga leyendo para averiguar cómo importar una carpeta en el sitio.

4.2.2. Importar en el sitio

Asumiendo que ya tiene un repositorio, y que quiere añadir una nueva estructura de carpetas e él, sólo tiene que seguir estos pasos:

1. Utilice el navegador de repositorios para crear nuevas carpetas de proyecto directamente en el repositorio.

2. Ejecute la operación obtener de la nueva carpeta sobre la carpeta de más alto nivel que desea importar. Obtendrá una advertencia porque la carpeta local no está vacía. Ahora tiene una carpeta de más alto nivel versionada con contenido no versionado.

3. Utilice TortoiseSVN → Añadir... en esta carpeta versionada para añadir parte o todo su contenido. Puede añadir y eliminar ficheros, establecer las propiedades `svn:ignore` en las carpetas y hacer cualquier otro cambio que necesite.

4. Confirme la carpeta de más alto nivel, y ya tiene un nuevo árbol versionado, y una copia de trabajo local, creada desde su carpeta existente.

4.2.3. Ficheros especiales

A veces necesitará tener un fichero bajo control de versiones que contenga datos específicos del usuario. Esto significa que tiene un fichero que cada desarrollador/usuario necesita modificar para que se ajuste a su configuración local. Pero versionar ese fichero es dificil, porque cada usuario haría confirmaciones de sus cambios cada vez en el repositorio.

En estos casos le sugerimos que utilice ficheros `plantilla`. Cree un fichero que contenga todos los datos que sus desarrolladores puedan necesitar, añádalo al control de versiones y haga que sus desarrolladores lo obtengan. Luego, cada desarrollador tendrá que *hacer una copia* de ese fichero y renombrar esa copia. Después de eso, modificar la copia no vuelve a ser un problema.

Por poner un ejemplo, puede mirar el script de compilación de TortoiseSVN. Se invoca a un fichero llamado `TortoiseVars.bat` que no existe en el repositorio. Sólo existe el fichero `TortoiseVars.tmpl`. `TortoiseVars.tmpl` es el fichero plantilla del que cada desarrollador tiene que hacer una copia y cambiarla de nombre a `TortoiseVars.bat`. Dentro de ese fichero, hemos añadido comentarios para que los usuarios vean qué lineas tienen que editar y cambiar de acuerdo a sus configuraciones locales para que funcione.

Para no molestar a los usuarios, también hemos añadido el fichero `TortoiseVars.bat` a la lista de ignorados de su carpeta padre, es decir, hemos cambiado la propiedad de Subversion `svn:ignore` para incluir ese nombre de fichero. De esta forma no se mostrará como no versionado en cada confirmación.

4.3. Obteniendo una copia de trabajo

Para tener una copia de trabajo necesita *obtener* una de un repositorio.

Seleccione un directorio en el explorador de Windows donde quiera poner su copia de trabajo. Haga click con el botón derecho para mostrar el menú contextual y seleccione el comando TortoiseSVN → Obtener..., que mostrará el siguiente cuadro de diálogo:

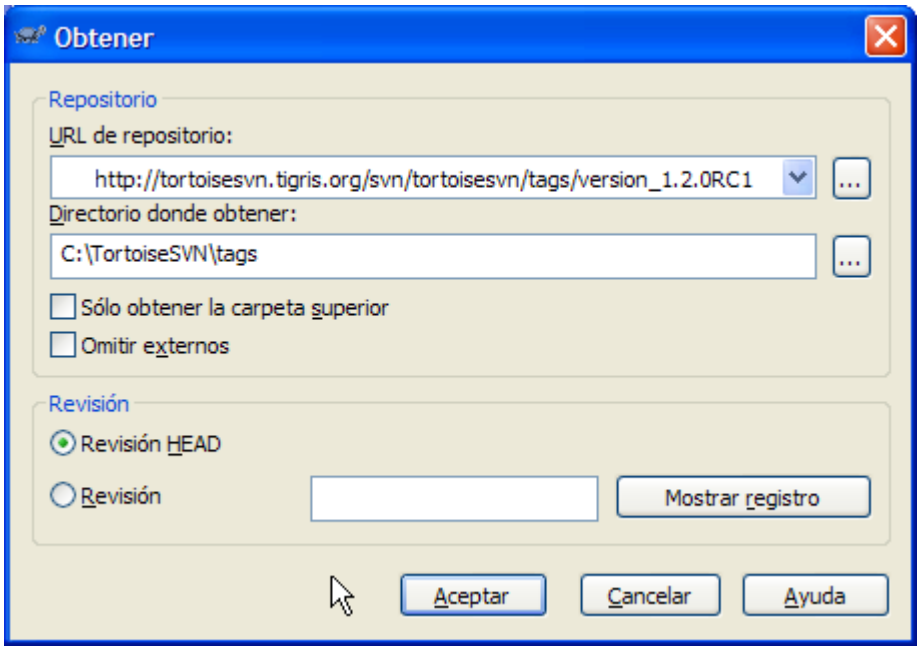


Figura 4.7. El diálogo Obtener

Si introduce un nombre de carpeta que no aún no exista, se creará un directorio con ese nombre.

4.3.1. Profundidad de obtención

Puede elegir la *profundidad* que desea para la obtención, lo que le permite especificar la profundidad de la recusión en las carpetas hijas. Si sólo desea unas pocas secciones de un árbol grande, puede obtener sólo la carpeta de más alto nivel, y luego actualizar las carpetas seleccionadas de forma recursiva.

Totalmente recursivo
: Obtener el árbol entero, incluyendo todas las carpetas hijas y subcarpetas.

Hijos inmediatos, incluyendo carpetas
: Obtener el directorio especificado, incluyendo todos los fichero sy carpetas hijas, pero no rellenar las carpetas hijas.

Sólo los ficheros hijos
: Obtener la carpeta especificada, incluyendo todos los ficheros pero no obtener ninguna carpeta hija.

Sólo este ítem
: Obtener sólo el directorio. No rellenarlo con ficheros ni carpetas hijas.

Copia de trabajo
: Retiene la profundidad especificada en la copia de trabajo. Esta opción no se utiliza en el diálogo obtener, pero es el valor por defecto para todos los demás diálogos que tengan opción de profundidad.

Excluír
: Utilizado para reducir la profundidad de una copia de trabajo después de que una carpeta haya sido rellenada. Esta opción sólo está disponible en el diálogo Acctualizar a la revisión.

Si obtiene una copia de trabajo dispersa (por ejemplo seleccionando cualquier otra opción distinta de `totalmente recursivo` para la profundidad de la obtención), puede conseguir sub-carpetas adicionales utilizando el Navegador de repositorios (Sección 4.24, “El navegador de repositorios”) o el diálogo Comprobar modificaciones (Sección 4.7.3, “Estado local y remoto”).

En el navegador de repositorios, haga click con el botón derecho en la carpeta donde ha ejecutado obtener, y luego utilice TortoiseSVN → Navegador de repositorios para lanzar el diálogo. Localice la subcarpeta que desea añadir a su copia de trabajo, y utilice Menú contextual → Actualizar ítem a la revisión... Este menú sólo será visible si el ítem seleccionado no existe aún en su copia de trabajo, mientras que el ítem padre sí existe.

En el diálogo Comprobar modificaciones, primero haga click en el botón Comprobar repositorio. El diálogo mostrará como `añadido remotamente` todos los ficheros y carpetas que están en el repositorio pero que aún no se han obtenido. Haga click con el botón derecho sobre la carpeta o carpetas que desea añadir a su copia de trabajo, y luego utilice Menú contextual → Actualizar.

Esta característica es muy útil si sólo desea obtener partes de un árbol más grande, pero desea la facilidad de poder actualizar una única copia de trabajo. Suponga que tiene un gran árbol que tiene subcarpetas desde `Proyecto01` a `Proyecto99`, y sólo desea obtener las carpetas `Proyecto03`, `Proyecto25` y `Proyecto76/SubProy`. Utilice estos pasos:

1. Obtenga la carpeta padre con profundidad “Sólo este ítem”. Ahora tiene una carpeta de nivel superior vacía.

2. Seleccione la nueva carpeta y utilice TortoiseSVN → Navegador de repositorios para ver el contenido del repositorio.

3. Haga click con el botón derecho sobre `Proyecto03` y Menú contextual → Actualizar ítem a la revisión.... Mantenga las configuraciones por defecto y haga click en Aceptar. Ahora tendrá una carpeta totalmente poblada.

   Repita el mismo proceso para `Proyecto25`.

4. Navegue a `Proyecto76/SubProy` y haga lo mismo. Esta vez notará que la carpeta `Proyecto76` no tiene otro contenido que `SubProy`, que a su vez está totalmente poblado. Subversion ha creado las carpetas intermedias pero sin rellenarlas.

Cambiando la profundidad de la copia de trabajo

Una vez que haya obtenido una copia de trabajo a una profundidad en concreto, puede cambiar dicha profundidad más tarde para obtener más o menos contenido utilizando Menú contextual → Actualizar ítem a la revisión....

Utilizando un servidor antiguo

Los servidores anteriores a la versión 1.5 no entienden la petición de profundidad de copia de trabajo, por lo que no siempre pueden manejar las peticiones de forma eficiente. El comando funcionará, pero un servidor antiguo puede que envíe todos los datos, dejando que sea el cliente quien filtre lo que no necesite, lo que puede significar un montón de tráfico de red. Si es posible, debería actualizar su servidor a la versión 1.5.

Si el proyecto contiene referencias a proyectos externos que *no* desea que se obtengan al mismo tiempo, utilice la casilla Omitir externos.

Importante

Si está marcado Omitir externos, o si desea incrementar el valor de la profundidad, deberá realizar actualizaciones a su copia de trabajo utilizando TortoiseSVN → Actualizar a la Revisión... en vez de TortoiseSVN → Actualizar. La actualización estándar incluirá todos los externos y mantendrá la profundidad actual.

Le recomendamos que obtenga únicamente la parte `trunk` del árbol de directorios, o desde más abajo. Si especifica la ruta padre del árbol de directorios en la URL, al final puede acabar con un disco duro lleno ¡porque obtendrá una copia del árbol completo del repositorio, incluyendo cada rama y etiqueta de su proyecto!

Exportando

A veces puede querer crear una copia local sin ninguno de esos directorios `.svn`, por ejemplo para crear un fichero comprimido de sus fuentes. Lea Sección 4.26, “Exportando una copia de trabajo de Subversion” para averiguar cómo hacerlo.

4.4. Confirmando sus cambios en el repositorio

Enviar los cambios que ha hecho al repositorio se conoce como *confirmar* los cambios. Pero antes de confirmar tiene que estar seguro de que su copia de trabajo está actualizada. Puede o bien ejecutar TortoiseSVN → Actualizar directamente, o bien ejecutar TortoiseSVN → Comprobar Modificaciones primero, para ver qué se ha cambiado localmente o en el servidor.

4.4.1. El diálogo de Confirmación

Si su copia de trabajo está actualizada y no hay conflictos, ya está preparado para confirmar sus cambios. Seleccione los ficheros y/o carpetas que desee confirmar y seleccione TortoiseSVN → Confirmar....

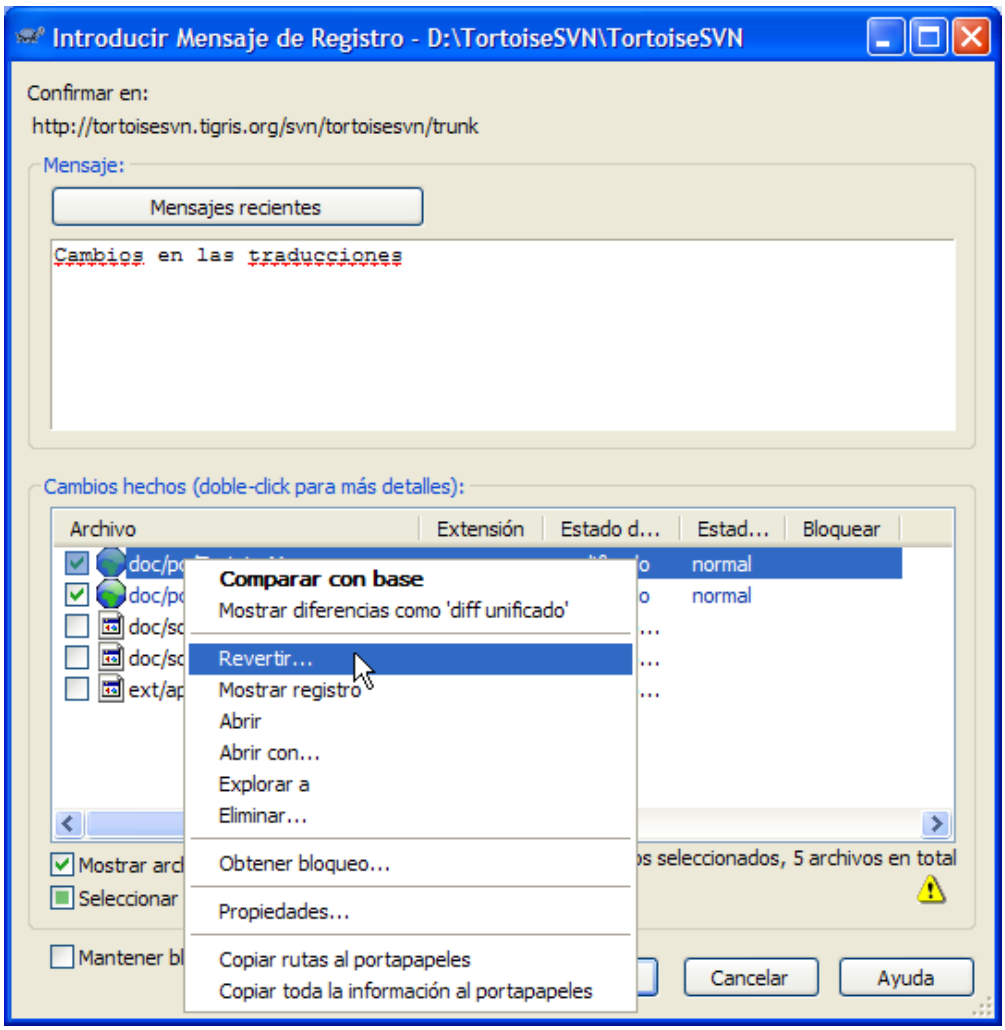


Figura 4.8. El diálogo de Confirmación

El diálogo de confirmación le mostrará todos los ficheros cambiados, incluso los ficheros añadidos, borrados o no versionados. Si no desea que un fichero cambiado se confirme, símplemente desmarque ese fichero. Si desea incluir un fichero no versionado, márquelo para añadirlo a la confirmación.

Los ítems que han sido cambiados a una ruta de repositorio diferente también se indican utilizando un marcador `(s)`. Puede haber cambiado algo mientras trabaja en una rama y habérsele olvidado volver a cambiarlo al tronco. ¡Este es su signo de advertencia!

¿Confirmar ficheros o carpetas?

Cuando confirma ficheros, el diálogo de confirmación sólo le enseña los ficheros que ha seleccionado. Cuando confirma una carpeta el diálogo de confirmación seleccionará los ficheros que han cambiado de forma automática. Si se olvidó un fichero nuevo que haya creado, al confirmar la carpeta lo encontrará. Confirmar una carpeta *no* significa que todos los ficheros se marquen como cambiados; sólo le hace la vida más fácil haciendo más trabajo por usted.

Si ha modificado ficheros que han sido incluidos desde un repositorio diferente utilizando `svn:externals`, esos cambios no pueden ser incluídos en la misma confirmación atómica. Aparecerá un símbolo de advertencia bajo la lista de ficheros que le indicará si esto ha ocurrido, y el texto de ayuda le explicará que esos ficheros externos deben confirmarse de forma separada.

Muchos ficheros sin versionar en el diálogo de confirmar

Si cree que el diálogo de confirmación de TSVN le muestra demasiados ficheros no versionados (por ejemplo, ficheros generados por el compilador o copias de seguridad hechas por su editor), hay varias formas de manejar esta situación. Puede:

- añadir el fichero (o una extensión con máscara) a la lista de ficheros a ignorar en la página de configuración. Esto afectará a todas las copias de trabajo que tenga.
- añadir el fichero a la lista de `svn:ignore` utilizando TortoiseSVN → Añadir a la lista de ignorados. Esto únicamente afectará al directorio en el que establezca la propiedad `svn:ignore`. Puede cambiar la propiedad `svn:ignore` de un directorio utilizando el Diálogo de Propiedades SVN.

Para más información, lea Sección 4.13, “Ignorando ficheros y directorios”.

Haciendo doble click en cualquier fichero modificado en el diálogo de confirmación, se lanzará la herramienta externa de diferenciar para mostrarle sus cambios. El menú contextual le proporciona más opciones, como se ve en la captura de pantalla. También puede arrastrar ficheros desde aquí a otra aplicación, como un editor de textos o un IDE.

Puede seleccionar o deseleccionar ítems haciendo click en la casilla a la izquierda del ítem. Para los directorios puede utilizar **Mayúsculas**-Seleccionar para ejecutar la acción de forma recursiva.

Las columnas que se muestran en el panel inferior son personalizables. Si hace click con el botón derecho en cualquier cabecera de columna verá un menú contextual que le permite seleccionar qué columnas se muestran. También puede cambiar el ancho de la columna utilizando el manejador de arrastre que aparece cuando mueve el cursor sobre el límite de una columna. Estas personalizaciones se mantienen, por lo que verá los mismos encabezados la siguiente vez.

Por defecto cuando confirma los cambios, cualquier bloqueo que tenga en los ficheros se libera automáticamente cuando la confirmación tiene éxito. Si desea mantener esos bloqueos, asegúrese de que la casilla Mantener bloqueos está marcada. El estado por defecto de esta casilla se toma de la opción `no_unlock` del fichero de configuración de Subversion. Lea Sección 4.30.1, “Configuración general” para más información sobre cómo editar el fichero de configuración de Subversion.

Arrastrar y soltar

Puede arrastrar ficheros hasta el diálogo de confirmación desde cualquier otro lugar, siempre y cuando las copias de trabajo sean del mismo repositorio. Por ejemplo, puede tener una copia de trabajo enorme con diferentes ventanas del explorador abiertas en carpetas distantes de la jerarquía. Si quiere evitar confirmar desde la carpeta más alta (lo que implica una lenta operación de búsqueda de cambios) puede abrir el diálogo de confirmar para una carpeta y arrastrar desde las otras ventanas para incluir ficheros dentro de la misma confirmación atómica.

Puede arrastrar ficheros no versionados que residan dentro de una copia de trabajo al diálogo de confirmación, y automáticamente serán SVN añadidos.

Reparando renombrados externos

A veces los ficheros se renombran fuera de Subversion, y se muestran en la lista de ficheros como un fichero faltante y un fichero no versionado. Para evitar perder la historia necesita notificar a Subversion su conexión. Simplemente seleccione tanto el nombre antiguo (faltante) como el nombre nuevo (sin versionar) y utilice Menú contextual → Reparar movimiento para emparejar los dos ficheros como un renombrado.

4.4.2. Listas de cambios

El diálogo de confirmación da soporte a las listas de cambios de Subversion para ayudar a agrupar ficheros relacionados. Averigue más sobre esta característica en Sección 4.8, “Listas de cambios”.

4.4.3. Excluyendo ítems de la lista de confirmación

A veces tiene ficheros versionados que cambian con frecuencia pero que realmente no desea confirmar. En ocasiones esto indica un fallo en su sistema de compilación - ¿por qué están esos ficheros versionados? ¿debería utilizar ficheros de plantilla? Pero ocasionalmente es inevitable. Una razón clásica es que su IDE cambie una fecha en el fichero de proyecto cada vez que lo compile. El fichero de proyecto debe estar versionado ya que contiene todas las configuraciones de la compilación, pero no necesita confirmarse sólo porque la fecha haya cambiado.

Para ayudarle en casos tan extraños como estos, hemos reservado una lista de cambios llamada `ignore-on-commit`. Cualquier fichero añadido a esta lista de cambios se desmarcará automáticamente en el diálogo de confirmación. Aún puede confirmar los cambios, pero tendrá que selecionarlo manualmente en el diálogo de confirmación.

4.4.4. Mensajes de registro de confirmación

Asegúrese de introducir un mensaje de registro que describa los cambios que está confirmando. Esto le ayudará a saber qué ocurrió y cuando según navegue por los mensajes de registro del proyecto en el futuro. El mensaje puede ser tan extenso o escueto como desee; muchos proyectos tienen directrices sobre qué debe incluirse en ellos, el idioma que debe utilizarse, y a veces incluso un formato estricto.

Puede aplicar formatos sencillos en sus mensajes de registro utilizando una convención similar a la usada en los emails. Para aplicar un estilo a un `texto`, utilice `*texto*` para la negrita, `_texto_` para el subrayado, y `^texto^` para la cursiva.

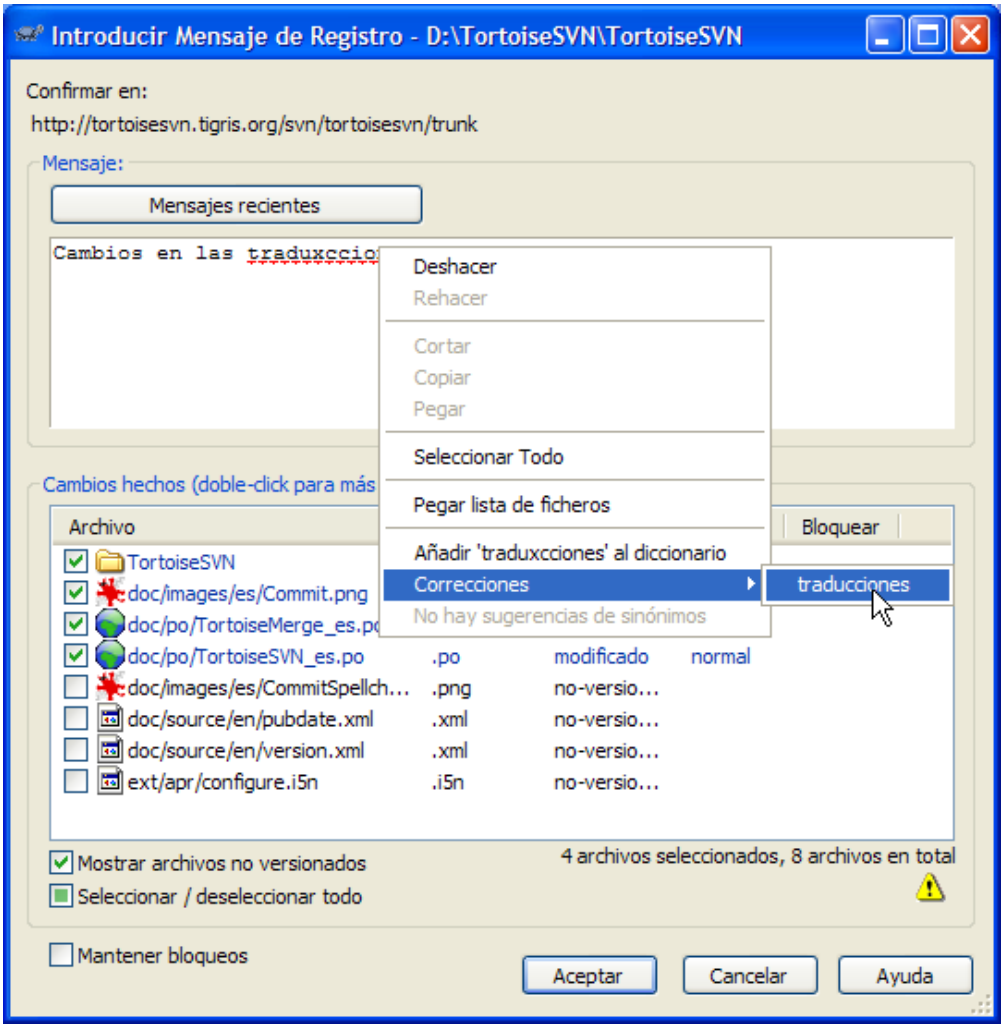


Figura 4.9. El corrector ortográfico del diálogo de Confirmación

TortoiseSVN incluye un corrector ortográfico para ayudarle a escribir sus mensajes de registro correctamente. Este corrector señalará cualquier palabra mal escriba. Utilice el menú contextual para acceder a las correcciones sugeridas. Por supuesto, el corrector no conoce *todos* los términos técnicos que utiliza, así que a veces palabras bien escritas aparecerán como errores. Pero no se preocupe. Puede simplemente añadirlas a su diccionario personal utilizando el menú contextual.

La ventana de mensajes de registro también incluye una facilidad de autocompletar nombres de ficheros y funciones. Esto utiliza expresiones regulares para extraer clases y nombres de funciones de los ficheros (de texto) que está confirmando, y también los propios nombres de ficheros. Si una palabra que está tecleando concuerda con algo en la lista (después de haber tecleado al menos 3 caracteres, o de pulsar **Ctrl+Espacio**), aparecerá un desplegable que le permitirá seleccionar el nombre completo. Las expresiones regulares suministradas con TortoiseSVN se mantienen en la carpeta `bin` de la instalación de TortoiseSVN. También puede definir sus propias expresiones regulares y almacenarlas en `%APPDATA%\TortoiseSVN\autolist.txt`. Por supuesto su lista privada no se sobreescribirá cuando actualice su instalación de TortoiseSVN. Si no está familiarizado con las expresiones regulares, eche un vistazo a la documentación en *http://en.wikipedia.org/wiki/Regular_expression*, y a la documentación en línea y al tutorial en *http://www.regular-expressions.info/*.

Puede reutilizar mensajes de registro que haya introducido anteriormente. Tan sólo debe pulsar en Mensajes recientes para ver una lista de los últimos mensajes que ha introducido para esta copia de trabajo. El número de mensajes almacenados se puede personalizar en el diálogo de configuración de TortoiseSVN.

Puede limpiar todos los mensajes de confirmación almacenados desde la página Datos Almacenados de la configuración de TortoiseSVN, o puede eliminar mensajes individuales dentro del diálogo Mensajes recientes utilizando la tecla **Supr**.

Si desea incluir las rutas marcadas en su mensaje de registro, puede utilizar el comando Menú contextual → Pegar lista de nombres de ficheros en el control de edición.

Otra forma de insertar rutas en el mensaje de registro es simplemente arrastrar los ficheros desde la lista de ficheros al control de edición.

Propiedades especiales de carpetas

Hay diversas propiedades especiales de carpeta que pueden usarse para darle mayor control sobre el formato de los mensajes de registro de las confirmaciones y el idioma que utiliza el módulo del corrector ortográfico. Para más información, lea Sección 4.17, “Configuración del proyecto”.

Integración con herramientas de control de errores

Si ha activado el sistema de control de errores, puede poner una o más incidencias en el cuatro de texto Bug-ID / Nº-Incid.:. Si quiere introducir múltiples incidencias, sepárelas por comas. Alternativamente, si está utilizando el soporte de control de errores basado en expresiones regulares, simplemente añada las referencias a sus incidencias como parte del mensaje de registro. Si desea saber más, lea Sección 4.28, “Integración con sistemas de control de errores / seguimiento de incidencias”.

4.4.5. Progreso de confirmación

Tras pulsar Aceptar aparece un diálogo mostrando el progreso de la confirmación.

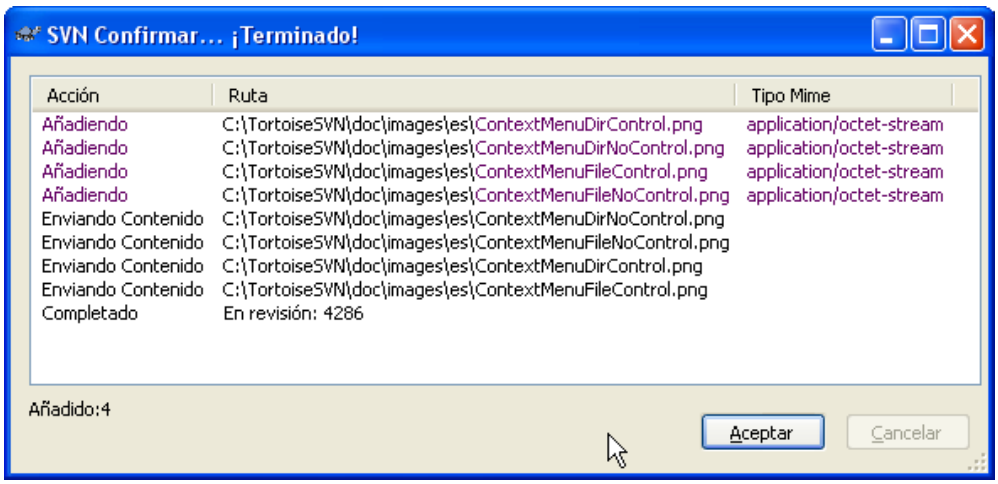


Figura 4.10. El diálogo Progreso mostrando el progreso de una confirmación

El diálogo de progreso utiliza una codificación de colores para resaltar las diferentes acciones de confirmación:

Azul
: Confirmando una modificación.

Púrpura
: Confirmando un ítem añadido.

Rojo oscuro
: Confirmando un borrado o un reemplazo.

Negro
: Todos los demás ítems.

Este es el esquema de colores por defecto, pero puede personalizar dichos colores utilizando el diálogo de configuración. Para más información, lea Sección 4.30.1.4, “Configuración de colores de TortoiseSVN”.

4.5. Actualice su copia de trabajo con los cambios de otros

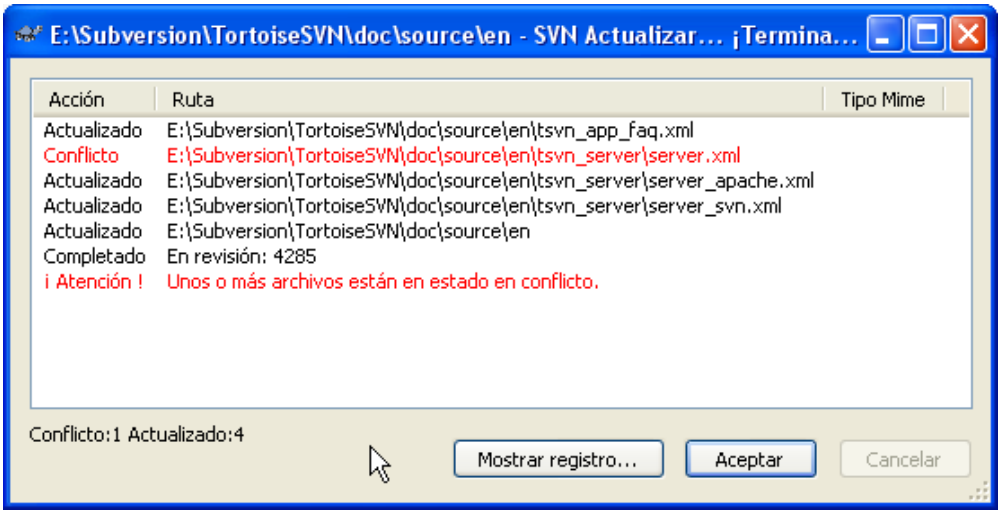


Figura 4.11. Diálogo de progreso mostrando una actualización terminada

Periódicamente, debería asegurarse de que los cambios que hacen los demás se incorporen en su copia de trabajo local. El proceso de incorporar los cambios desde el servidor a su copia de trabajo local se conoce como *actualización*. La actualización puede hacerse en ficheros sueltos, en un conjunto de ficheros, o recursivamente en jerarquías completas de directorios. Para actualizar, seleccione los ficheros y/o directorios que desee, haga click con el botón derecho y seleccione TortoiseSVN → Actualizar en el menú contextual del explorador. Aparecerá una ventana con el progreso de la actualización según se ejecuta. Los cambios que los demás hayan hecho se fusionarán con sus ficheros, manteniendo cualquier cambio que haya hecho en los mismos ficheros. El repositorio *no* se ve afectado por una actualización.

El diálogo de progreso utiliza un código de colores para resaltar diferentes acciones de actualización:

Púrpura
: Nuevo ítem añadido a su copia de trabajo

Rojo oscuro
: Ítem redundante borrado de su copia de trabajo, o ítem faltante reemplazado en su copia de trabajo.

Verde
: Cambios del repositorio que se han fusionado satisfactoriamente con sus cambios locales.

Rojo brillante
: Cambios del repositorio fusionados con sus cambios locales, pero que han dado lugar a conflictos que debe resolver.

Negro
: Items sin cambios en su copia de trabajo actualizados con una versión más nueva desde el repositorio.

Este es el esquema de colores por defecto, pero puede personalizar dichos colores utilizando el diálogo de configuración. Para más información, lea Sección 4.30.1.4, “Configuración de colores de TortoiseSVN”.

Si obtiene algún *conflicto* durante una actualización (esto puede suceder si los demás han cambiado las mismas líneas del mismo fichero a la vez que usted y esos cambios no concuerdan), el diálogo muestra esos conflictos en rojo. Puede hacer doble click en esas líneas para iniciar la herramienta de fusión externa para resolver los conflictos.

Cuando se completa la actualización, el diálogo de progreso le muestra un resumen con el número de ítems actualizados, añadidos, eliminados, en conflicto, etc. bajo la lista de ficheros. Esta información de resumen puede copiarse al portapapeles utilizando Ctrl+C.

El comando Actualizar estándar no tiene opciones y simplemente actualiza su copia de trabajo a la revisión HEAD del repositorio, lo que es el caso de uso más común. Si desea más control sobre el proceso de actualización, debería utilizar TortoiseSVN → Actualizar a la revisión.... Esto le permite actualizar su copia de trabajo a una revisión específica, no sólo a la más reciente. Suponga que su copia de trabajo está en la revisión 100, pero quiere que refleje el estado que tenía en la revisión 50 - entonces simplemente actualice a la revisión 50. En el mismo diálogo puede elegir la *profundidad* para la actualización de la carpeta actual. Estos términos se describen en Sección 4.3.1, “Profundidad de obtención”. La profundidad por defecto es Copia de trabajo, lo que respeta la configuración de profundidad ya existente. También puede decidir si desea ignorar cualquier proyecto externo en la actualización (esto es, los proyectos referenciados utilizando `svn:externals`).

Atención

Si actualiza un fichero o una carpeta a una revisión en concreto, no debería hacer cambios en esos ficheros. ¡Obtendrá mensajes `out of date` (desactualizado) cuando intente confirmarlos! Si desea deshacer los cambios de un fichero y empezar de nuevo desde una revisión anterior, puede revertir a una revisión previa desde el diálogo de registro de revisiones. Lea Sección B.4, “Deshacer revisiones en el repositorio” si desea más instrucciones, y métodos alternativos.

Actualizar a la revisión puede ser útil a veces para ver cómo estaba su proyecto en un momento anterior en su historia. Pero en general, actualizar ficheros individuales a una revisión anterior no es una buena idea, ya que deja su copia de trabajo en un estado inconsistente. Si el fichero que está actualizando ha cambiado de nombre, incluso puede encontrar que ese fichero ha desaparecido de su copia de trabajo porque en esa revisión no había ningún fichero con ese nombre. También debe tener en cuenta que el ítem mostrará una sobreimpresión normal verde, por lo que no se puede distinguir del resto de ficheros que están actualizados.

Si desea simplemente una copia local de una versión antigua de un fichero, es mejor utilizar el comando Menú contextual → Guardar revisión en... desde el diálogo de registro para dicho fichero.

Múltiples ficheros/carpetas

Si selecciona múltiples ficheros y carpetas en el explorador y luego selecciona Actualizar, todos esos ficheros/carpetas se actualizan uno a uno. ¡TortoiseSVN se asegura de que todos los ficheros/carpetas del mismo repositorio se actualicen exactamente a la misma revisión! Incluso si entre esas actualizaciones ocurrió alguna confirmación.

El fichero local ya existe

A veces cuando intente actualizar, la actualización falla con un mensaje para decir que ya existe un fichero local con el mismo nombre. Esto típicamente ocurre cuando Subversion intenta obtener un fichero recién versionado, y se encuentra un fichero no versionado del

mismo nombre en su copia de trabajo. Subversion nunca sobreescribirá un fichero no versionado - puede contener algo en lo que está trabajando, y que casualmente tiene el mismo nombre de fichero que otro desarrollador ha utilizado para su recién confirmado fichero.

Si obtiene este mensaje de error, la solución es simplemente renombrar el fichero local sin versionar. Tras completar la actualización, puede comprobar si el fichero renombrado sigue siendo necesario.

Si sigue obteniendo mensajes de error, utilize mejor el comando TortoiseSVN → Comprobar modificacioness para mostrar todos los ficheros con problemas. De esa forma puede lidiar con ellos de un golpe.

4.6. Resolviendo conflictos

De vez en cuando, obtendrá un *conflicto* cuando actualice/fusione sus ficheros desde el repositorio o cuando cambie su copia de trabajo a una URL diferente. Hay dos tipos de conflictos:

conflictos de fichero
: Un conflicto de fichero ocurre si dos (o más) desarrolladores han cambiado las mismas líneas de un fichero.

conflictos de árboles
: Un conflicto de árbol ocurre cuando un desarrollador mueve/renombra/elimina un fichero o una carpeta, que otro desarrollador también ha movido/renombrado/borrado, o quizás lo haya modificado.

4.6.1. Conflictos de ficheros

Un conflicto de fichero ocurre cuando uno o más desarolladores han hecho cambios en las mismas líneas de un fichero. Dado que Subversion no sabe nada de su proyecto, delega la resolución de los conflictos en los desarrolladores. Cuando se le informa de un conficto, debería abrir el fichero en cuestión, y buscar líneas que empiecen con el texto <<<<<<<. El área conflictiva se marca así:

```
<<<<<<< nombre-del-fichero
 sus cambios
=======
 código fusionado del repositorio
>>>>>>> revisión
```

Además, para cada fichero en conflicto Subversion deja tres ficheros adicionales en su directorio:

nombre-del-fichero.ext.mine
: Este es su fichero tal y como estaba en su copia de trabajo antes de que actualizara su copia de trabajo - esto es, sin marcadores de conflicto. Este fichero tiene sus últimos cambios en él y nada más.

nombre-del-fichero.ext.rREV-ANTIGUA
: Este es el fichero que era la revisión BASE antes de que actualizara su copia de trabajo. Esto es, el fichero que obtuvo antes de empezar a hacer sus últimos cambios.

nombre-del-fichero.ext.rREV-NUEVA
: Este es el fichero que su cliente de Subversion acaba de recibir desde el servidor del que actualizó su copia de trabajo. Este fichero corresponde a la revisión HEAD del repositorio.

Puede o bien lanzar una herramienta externa de fusiones / editor de conflictos con el menú contextual TortoiseSVN → Editar Conflictos o bien utilizar otro editor manualmente para resolver el conflicto. Debe decidir cómo tiene que quedar el código, hacer los cambios necesarios, y grabar el fichero.

Después, ejecute el comando TortoiseSVN → Resolver y confirme sus modificaciones al repositorio. Tome nota de que el comando Resolver realmente no resuelve el conflicto. Simplemente elimina los ficheros `filename.ext.mine` y `filename.ext.r*`, dejándole confirmar sus cambios.

Si tiene conflictos con ficheros binarios, Subversion no intentará mezclar dichos ficheros por si mismo. El fichero local se mantendrá sin cambios (exactamente tal y como lo había cambiado usted) y obtendrá unos ficheros `nombrefichero.ext.r*`. Si desea descartar sus cambios y quedarse con la versión del repositorio, utilice el comando Revertir. Si desea mantener su versión y sobreescribir la versión del repositorio, utilice el comando Resuelto y luego confirme su versión.

Puede utilizar el comando Resuelto para múltiples ficheros si pulsa con el botón derecho en la carpeta padre y selecciona TortoiseSVN → Resuelto... Esto mostrará un diálogo con todos los ficheros en conflicto dentro de esa carpeta, y le permitirá seleccionar cuáles marcar como resueltos.

4.6.2. Conflictos de árbol

Un conflicto de árbol ocurre cuando un desarrollador mueve/renombra/elimina un fichero o una carpeta, que otro desarrollador también ha movido/renombrado/borrado, o quizás lo haya modificado. Hay diferentes situaciones que puede resultar en un conflicto de árbol, y todas ellas requiere pasos diferentes para resolver el conflicto.

Cuando se elimina un fichero de forma local en Subversion, el fichero también se elimina del sistema local de ficheros, por lo que incluso si es parte de un conflicto de árbol no se puede mostrar una sobreimpresión de conflicto y no puede hacer clic con el botón derecho sobre él para resolver el conflicto. En este caso, utilice el diálogo Comprobar modificaciones para acceder a la opción Editar conflictos.

TortoiseSVN puede ayudarle a encontrar el lugar correcto para fusionar los cambios, pero puede que necesite realizar un trabajo adicional para arreglar los conflictos. Recuerde que tras una actualización la BASE de trabajo siempre contendrá la revisión de cada ítem tal y como estaba en el repositorio en el momento de la actualización. Si revierte un cambio tras la actualización, se revierte a su estado del repositorio, no a como estaba cuando empezó a hacer sus propios cambios locales.

4.6.2.1. Borrado local, llega un cambio en la actualización

1. El desarrollador A modifica `Foo.c` y lo confirma en el repositorio

2. El desarrollador B al mismo tiempo ha movido `Foo.c` a `Bar.c` en su propia copia de trabajo, o simplemente ha borrado `Foo.c` o su carpeta padre.

Una actualización de la copia de trabajo del desarrollador B acaba con un conflicto de árbol:

- `Foo.c` ha sido borrado de la copia de trabajo, pero está marcado como un conflicto de árbol.

- Si el conflicto aparece después de un renombrado en vez de un borrado, entonces `Bar.c` está marcado como añadido, pero no contiene las modificaciones del desarrollador A.

El desarrollador B ahora tiene que elegir si mantiene los cambios realizados por el desarrollador A. En el caso de un renombrado, puede fusionar los cambios de `Foo.c` dentro del fichero renombrado `Bar.c`. Para simples borrados de ficheros o directorios puede elegir quedarse con los cambios del ítem del desarrollador A y descartar el borrado. O, si marca el conflicto como resuelto sin hacer nada más, estará descartando los cambios del desarrollador A.

El diálogo de editar conflictos ofrece la posibilidad de fusionar cambios si puede encontrar el fichero original del renombrado `Bar.c`. Dependiendo de dónde se haya realizado la actualización, puede que no sea posible encontrar el fichero de origen.

4.6.2.2. Edición local, entra un borrado en la actualización

1. El desarrollador A mueve `Foo.c` a `Bar.c` y lo confirma en el repositorio.

2. El desarrollador B modifica `Foo.c` en su copia de trabajo.

O en el caso de mover una carpeta...

1. El desarrollador A mueve la carpeta padre `FooFolder` a `BarFolder` y lo confirma en el repositorio.

2. El desarrollador B modifica `Foo.c` en su copia de trabajo.

Una actualización de la copia de trabajo de B acaba con un conflicto de árbol. Para un conflicto simple de fichero:

- `Bar.c` se añade a la copia de trabajo como un fichero normal.
- `Foo.c` se marca como añadido (con historia) y tiene un conflicto de árbol.

Para un conflicto de carpeta:

- `BarFolder` se añade a la copia de trabajo como una carpeta normal.
- `FooFolder` se marca como añadida (con historia) y tiene un conflicto de árbol.

  `Foo.c` se marca como modificado.

El desarrollador B tiene ahora que decidir si desea continuar con la reorganización del desarrollador A y fusionar sus cambios en los ficheros correspondientes de la nueva estructura, o simplemente revertir los cambios de A y mantener el fichero local.

Para fusionar sus cambios locales con la reorganización, el desarrollador B tiene que encontrar primero qué nombre de fichero tiene ahora el fichero en conflicto `Foo.c` que fue renombrado/movido en el repositorio. Esto puede hacerse utilizando el diálogo de registro. Luego debe fusionar los cambios a mano ya que no hay actualmente forma de automatizar o simplificar este proceso. Una vez que se hayan portado los cambios, la ruta en conflicto es redundante y puede borrarse. En este ccaso utilice el botón Eliminar en el diálogo de editar conflictos para limpiar y marcar el conflicto como resuelto.

Si el desarrollador B decide que los cambios de A eran erróneos deberá elegir el botón Mantener en el diálogo de editar conflictos. Esto marca el fichero o carpeta en conflicto como resuelto, pero los cambios del desarrollador A tendrán que eliminarse a mano. De nuevo el diálogo de registro ayuda a controlar lo que se ha movido.

4.6.2.3. Eliminación local, entra una eliminación en la actualización

1. El desarrollador A mueve `Foo.c` a `Bar.c` y lo confirma en el repositorio

2. El desarrollador B mueve `Foo.c` a `Bix.c`

Una actualización de la copia de trabajo del desarrollador B acaba con un conflicto de árbol:

- `Bix.c` se marca como añadido con historia.
- `Bar.c` se añade a la copia de trabajo con el estado 'normal'.
- `Foo.c` se marca como borrado y tiene un conflicto de árbol.

Para resolver este conflicto, el desarrollador B tiene que averiguar qué nombre de fichero tiene ahora el fichero en conflicto `Foo.c` que fue renombrado/movido en el repositorio. Esto puede hacerse utilizando el diálogo de registro.

Luego el desarrollador B tiene que decidir qué nuevo nombre de fichero de `Foo.c` mantiene - el del desarrollador A o el renombrado que hizo él mismo.

Después de que el desarrollador B haya resuelto manualmente el conflicto, el conflicto de árbol debe marcarse como resuelto mediante el botón del diálogo de editar conflictos.

4.6.2.4. Falta en local, entra un cambio en la fusión

1. El desarrollador A, que está trabajando en el tronco, modifica `Foo.c` y lo confirma en el repositorio
2. El desarrollador B, que está trabajando en una rama, mueve `Foo.c` a `Bar.c` y lo confirma en el repositorio

Una fusión de los cambios en el tronco del desarrollador A con los cambios de la copia de trabajo de la rama del desarrollador B acaba con un conflicto de árbol:

- `Bar.c` ya está en la copia de trabajo con estado 'normal'.
- `Foo.c` se marca como faltante con un conflicto de árbol.

Para resolver este conflicto, el desarrollador B tiene que marcar el fichero como resuelto en el diálogo de edición de conflictos, lo que lo eliminará de la lista de conflictos. Luego tendrá que decidir si copia el fichero faltante `Foo.c` desde el repositorio a la copia de trabajo, si fusiona los cambios del desarrollador A hechos a `Foo.c` en el fichero renombrado `Bar.c`, o si desea ignorar los cambios marcando el conflicto como resuelto y no haciendo nada más.

Tenga en cuenta que si copia el fichero faltante desde el repositorio y luego marca como resuelto, su copia de eliminará de nuevo. Tiene que resolver el conflicto antes.

4.6.2.5. Edición local, entra un borrado en la fusión

1. El desarrollador A, que está trabajando en el tronco, mueve `Foo.c` a `Bar.c` y lo confirma en el repositorio
2. El desarrollador B, que está trabajando en una rama, modifica el fichero `Foo.c` y lo confirma en el repositorio.

Hay un caso equivalente cuando se mueven carpetas, pero todavía no se detecta en Subversion 1.6...

1. El desarrollador A, que está trabajando en el tronco, mueve la carpeta padre `FooFolder` a `BarFolder` y lo confirma en el repositorio.
2. El desarrollador B, que está trabajando en un rama, modifica `Foo.c` en su copia de trabajo.

Una fusión de los cambios en el tronco del desarrollador A con los cambios de la copia de trabajo de la rama del desarrollador B acaba con un conflicto de árbol:

- `Bar.c` se marca como añadido.
- `Foo.c` se marca como modificado con un conflicto de árbol.

El desarrollador B tiene ahora que decidir si desea continuar con la reorganización del desarrollador A y fusionar sus cambios en los ficheros correspondientes de la nueva estructura, o simplemente revertir los cambios de A y mantener el fichero local.

Para fusionar sus cambios locales con la reorganización, el desarrollador B debe primero averiguar qué nombre de fichero tiene ahora el fichero en conflicto `Foo.c` que fue renombrado/movido en el repositorio. Esto se puede hacer utilizando el diálogo mostrar registro sobre el origen de la fusión. El editor de conflictos sólo muestra el registro para la copia de trabajo ya que no sabe qué ruta fue utilizada en la fusión, así que lo tendrá que averiguar por su cuenta. Luego se deben fusionar los cambios a mano ya que no hay actualmente forma de automatizar o simplificar este proceso. Una vez que los cambios se hayan portado, la ruta en conflicto es redundante y puede eliminarse. En este caso utilice el botón Eliminar en el diálogo de edición de conflictos para limpiar y marcar el conflicto como resuelto.

Si el desarrollador B decide que los cambios de A eran erróneos, deberá elegir el botón Mantener en el diálogo de editar conflictos. Esto marca el fichero o carpeta en conflicto como resuelto, pero los cambios del desarrollador A deberán eliminarse a mano. De nuevo el diálogo de registro sobre el origen de la fusión ayuda a averiguar qué se movió.

4.6.2.6. Eliminación local, entra un borrado en la fusión

1. El desarrollador A, que está trabajando en el tronco, mueve `Foo.c` a `Bar.c` y lo confirma en el repositorio
2. El desarrollador B, que está trabajando en una rama, mueve `Foo.c` a `Bix.c` y lo confirma en el repositorio

Una fusión de los cambios en el tronco del desarrollador A con los cambios de la copia de trabajo de la rama del desarrollador B acaba con un conflicto de árbol:

- `Bix.c` se marca con el estado normal (no modificado).
- `Bar.c` se marca como añadido con historia.
- `Foo.c` se marca como faltante con un conflicto de árbol.

Para resolver este conflicto, el desarrollador B tiene que averiguar qué nombre de fichero tiene ahora el fichero en conflicto `Foo.c` que fue renombrado/movido en el repositorio. Esto puede hacerse utilizando el diálogo de registro sobre el origen de la fusión. El editor de conflictos sólo muestra el registro de la copia de trabajo, dado que no conoce qué ruta se utilizó para la fusión, por lo que tendrá que averiguarlo por si mismo.

Luego el desarrollador B tiene que decidir qué nuevo nombre de fichero de `Foo.c` mantiene - el del desarrollador A o el renombrado que hizo él mismo.

Después de que el desarrollador B haya resuelto manualmente el conflicto, el conflicto de árbol debe marcarse como resuelto mediante el botón del diálogo de editar conflictos.

4.7. Obteniendo información del estado

Mientras está trabajando en su copia de trabajo a menudo necesitará saber qué ficheros ha cambiado/añadido/borrado o renombrado, o incluso qué ficheros han sido cambiados y confirmados por los demás.

4.7.1. Iconos sobreimpresionados

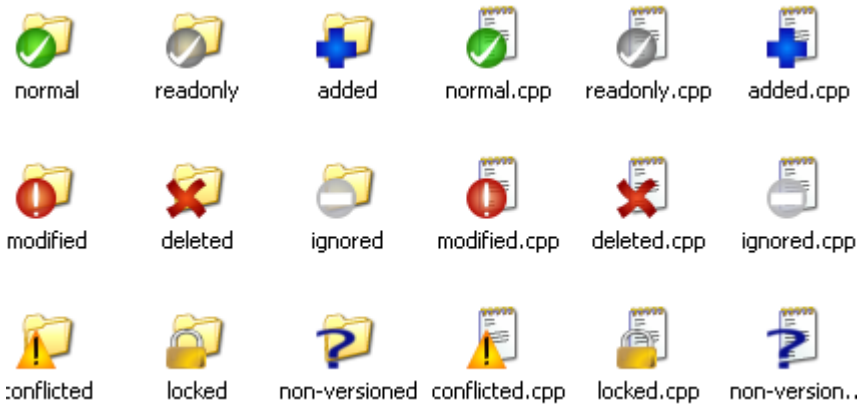


Figura 4.12. Explorador mostrando iconos sobreimpresionados

Ahora que ha obtenido una copia de trabajo desde un repositorio de Subversion, puede ver sus ficheros en el explorador de Windows con los iconos cambiados. Ésta es una de las razones por las que TortoiseSVN es tan popular. TortoiseSVN añade lo que se llama un icono sobreimpresionado al icono de cada fichero que se superpone al icono original del fichero. Dependiendo del estado en Subversion del fichero, el icono sobreimpresionado es diferente.

Una copia de trabajo recién obtenida tiene una marca verde como sobreimpresión. Esto significa que el estado de Subversion es `normal`.

En cuanto empiece a editar un fichero, el estado cambia a `modificado` y el icono sobreimpresionado cambia entonces a una marca de exclamación roja. De esta forma puede ver fácilmente qué ficheros se han cambiado desde la última vez que actualizó su copia de trabajo, y que necesitan ser confirmados.

Si durante una actualización ocurre un `conflicto`, el icono cambia a un signo de exclamación amarillo.

Si ha establecido la propiedad `svn:needs-lock` en un fichero, Subversion establece ese fichero como de sólo-lectura hasta que obtenga un bloqueo en él. Estos ficheros tienen esta sobreimpresión para indicarle que debe obtener un bloqueo antes de que pueda editarlo.

Si ha obtenido un bloqueo sobre un fichero, y el estado de Subversion es `normal`, este icono sobreimpresionado le recordará que debería liberar el bloqueo si no lo está utilizando para permitir a los demás que puedan confirmar sus cambios en el fichero.

Este icono le muestra que algunos ficheros o carpetas dentro de la carpeta actual se han marcado para ser `eliminados` del control de versiones, o bien que falta un fichero que está bajo el control de versiones dentro de una carpeta.

El signo más le indica que el fichero o carpeta está programado para ser `añadido` al control de versiones.

La barra le indica que el fichero o carpeta está *ignorado* para los asuntos de control de versiones. Esta sobreimpresión es opcional.

Este icono muestra los archivos y carpetas que no están bajo el control de versiones pero tampoco han sido ignorados. Esta sobreimpresión es opcional.

De hecho, puede que se encuentre con que no todos estos iconos se utilizan en su sistema. Esto se debe a que el número de sobreimpresiones permitidas por Windows está muy limitado y si está utilizando también una versión antigua de TrotoiseCVS, entonces no hay suficientes huecos de sobreimpresión disponibles. TortoiseSVN intenta ser un “Buen Ciudadano (TM)” y limita su uso de sobreimpresiones para darles una oportunidad al resto de aplicaciones.

Ahora que hay más clilentes Tortoise por ahí (TortoiseCVS, TortoiseHG, ...) el límite de iconos se ha convertido en un problema real. Para evitarlo, el proyecto TortoiseSVN ha introducido un conjunto compartido común de iconos, cargado como una DLL, que puede ser usado por todos los clientes Tortoise. Compruebe con el proveedor de su cliente para ver si esto ya se ha integrado :-)

Si desea una descripción de cómo se corresponden las sobreimpresiones con los estados de Subversion y otros detalles técnicos, lea Sección F.1, “Iconos sobreimpresionados”.

4.7.2. Columnas de TortoiseSVN en el Explorador de Windows

Se puede ver la misma información que está disponible en los iconos sobreimpresionados (y mucha más) como columnas adicionales en la Vista Detalles del Explorador de Windows.

Simplemente haga click con el botón derecho en la cabecera de una columna y seleccione Más... en el menú contextual que aparece. Se mostrará un diálogo donde puede especificar las columnas que se mostrarán en la “vista Detalles”, y su orden. Baje hasta que vea las entradas que empiezan por SVN. Marque aquellas que desee mostrar y cierre el diálogo pulsando Aceptar. Las columnas aparecerán a la derecha de las que ya se mostraban. Puede reorganizarlas utilizando arrastrar y soltar, o cambiarlas de tamaño, para que se ajusten a sus necesidades.

Importante

Las columnas adicionales en el Explorador de Windows no están disponibles en Vista, dado que Microsoft decidió no habilitar tales columnas para *todos* los ficheros, sino únicamente para tipos de ficheros específicos.

Sugerencia

Si desea que la organización actual se muestre en todas sus copias de trabajo, puede que desee convertirla en su vista por defecto.

4.7.3. Estado local y remoto

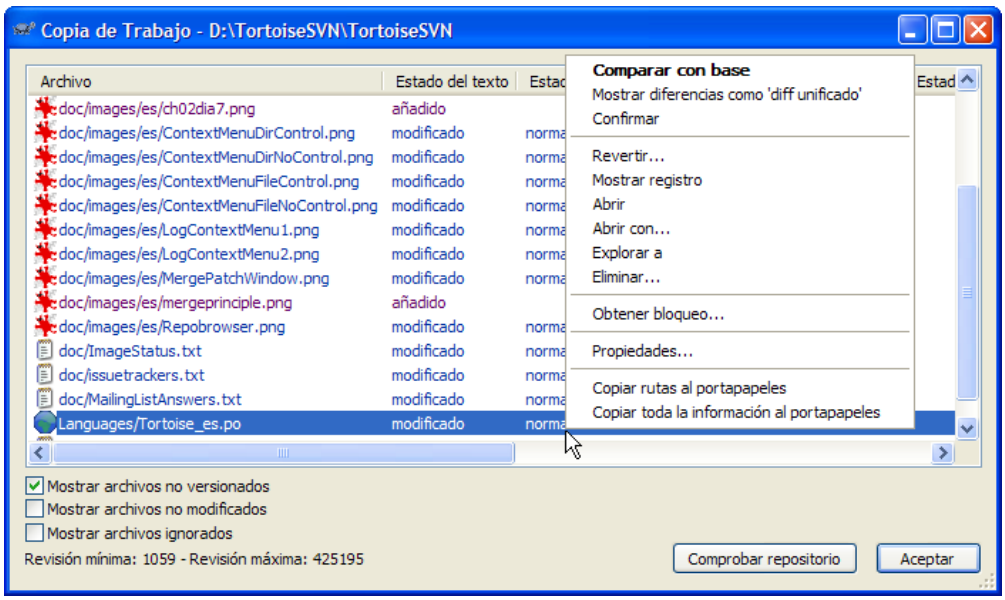


Figura 4.13. Comprobar modificaciones

A menudo es muy útil saber qué ficheros ha cambiado y también qué ficheros han cambiado y confirmado los demás. Ahí es donde viene bien el comando TortoiseSVN → Comprobar Modificaciones.... Este diálogo le muestra todos los ficheros que ha cambiado de alguna forma en su copia de trabajo, y además todos los ficheros no versionados que pueda tener.

Si pulsa en el botón Comprobar Repositorio también puede comprobar los cambios en el repositorio. De esa forma puede comprobar antes de hacer una actualización si es posible que haya un conflicto. También puede actualizar ficheros concretos desde el repositorio sin actualizar la carpeta completa. Por defecto, el botón Comprobar Repositorio sólo obtiene el estado remoto con la profundidad obtenida de la copia de trabajo. Si desea ver todos los ficheros y carpetas del repositorio, incluso aquellas que no ha obtenido, entonces tendrá que pulsar la tecla **Mays** mientras hace click en el botón Comprobar Repositorio.

El diálogo utiliza un código de colores para resaltar el estado.

Azul
: Ítems modificados localmente.

Púrpura
: Ítems añadidos. Los ítems que han sido añadidos con historia tienen un signo + en la columna Estado del texto, y un texto de ayuda que le muestra de dónde ha sido copiado.

Rojo oscuro
: Ítems faltantes o borrados.

Verde
: Ítems modificados localmente y en el repositorio. Los cambios se fusionarán al actualizar. Ésto *puede* producir conflictos al actualizar.

Rojo brillante
: Items modificados localmente y borrados en el repositorio, o modificados en el repositorio y borrados localmente. Esto *producirá* conflictos al actualizar.

Negro
: Ítems sin cambios y sin versionar.

Este es el esquema de colores por defecto, pero puede personalizar dichos colores utilizando el diálogo de configuración. Para más información, lea Sección 4.30.1.4, “Configuración de colores de TortoiseSVN”.

Los ítems que han sido cambiados a una ruta de repositorio diferente también se indican utilizando un marcador (s). Puede haber cambiado algo mientras trabaja en una rama y habérsele olvidado volver a cambiarlo al tronco. ¡Este es su signo de advertencia!

Desde el menú contextual del diálogo puede mostrar un resumen de los cambios. Compruebe los cambios locales que *usted* ha hecho utilizando Menú Contextual → Comparar con Base. Compruebe los cambios en el repositorio hechos por los demás utilizando Menú Contextual → Mostrar Diferencias como Diff Unificado.

También puede revertir cambios en ficheros individuales. Si ha borrado un fichero de forma accidental, se mostrará como *Falta* y puede utilizar

Los ficheros sin versionar y los ignorados se pueden enviar a la papelera de reciclaje desde aquí utilizando Menú Contextual → Eliminar. Si quiere eliminar los ficheros de forma definitiva (sin utilizar la papelera de reciclaje) pulse la tecla **Mayúsculas** mientras hace click en Eliminar.

Si desea examinar un fichero más en detalle, puede arrastrarlo desde aquí a otra aplicación, tal como un editor de textos o un IDE.

Las columnas son personalizables. Si hace click con el botón derecho en cualquier cabecera de columna verá un menú contextual que le permite seleccionar qué columnas se muestran. También puede cambiar el ancho de la columna utilizando el manejador de arrastre que aparece cuando mueve el cursor sobre el límite de una columna. Estas personalizaciones se mantienen, por lo que verá los mismos encabezados la próxima vez.

Si está trabajando al mismo tiempo en varias tareas sin relacion entre ellas, también puede agrupar los ficheros juntos en listas de cambios. Para más información, lea Sección 4.4.2, “Listas de cambios”.

En la parte inferior del diálogo puede ver un sumario del rango de revisiones del repositorio en uso en su copia de trabajo. Estas son revisiones *confirmadas*, no las revisiones *actualizadas*; representan el rango de revisiones donde estos ficheros fueron confirmados por última vez, no las revisiones a las que se han actualizado. Tenga en cuenta que el rango de revisiones mostrado se aplica sólo a los ítems mostrados, no a la copia de trabajo completa. Si quiere ver esa información para la copia de trabajo completa debe seleccionar la casilla Mostrar ficheros no modificados.

Sugerencia

Si quiere una vista llana de su copia de trabajo, por ejemplo mostrando todos los ficheros y carpetas en todos los niveles de su jerarquía de carpetas, entonces el diálogo Comprobar modificaciones es la forma más sencilla de conseguirlo. Símplemente seleccione la casilla Mostrar archivos no modificados para ver todos los ficheros de su copia de trabajo.

Reparando renombrados externos

A veces los ficheros se renombran fuera de Subversion, y se muestran en la lista de ficheros como un fichero faltante y un fichero no versionado. Para evitar perder la historia necesita notificar a Subversion su conexión. Simplemente seleccione tanto el nombre antiguo (faltante) como el nombre nuevo (sin versionar) y utilice Menú contextual → Reparar movimiento para emparejar los dos ficheros como un renombrado.

4.7.4. Viendo diferencias

A menudo querrá mirar dentro de sus ficheros, para echar un vistazo a lo que ha cambiado. Puede llevar esto a cabo seleccionando un fichero que haya cambiado, y seleccionando Diferenciar desde el menú contextual de TortoiseSVN. Esto inicia el visor externo de diferencias, que comparará el fichero actual con la copia prístina (revisión BASE), que se guardó tras su obtención o tras la última actualización.

Sugerencia

Puede mostrar diferencias incluso cuando no está dentro de una copia de trabajo, o cuando tiene múltiples versiones del fichero alrededor:

Seleecione los dos ficheros que desea comparar en el explorador (por ejemplo, utilizando la tecla **Ctrl** y el ratón) y seleccione Diferenciar del menú contextual de TortoiseSVN. El fichero que haya pulsado en último lugar (el que tiene el foco, es decir, el rectángulo con puntos) será tomado como más nuevo.

4.8. Listas de cambios

En un mundo ideal, sólo trabajará en una cosa cada vez, y su copia de trabajo sólo contendrá un conjunto de cambios lógicos. Vale, de vuelta al mundo real. A menudo ocurre que tiene que trabajar en varias tareas sin relación entre sí a la vez, y cuando mira en el diálogo de confirmar, todos los cambios están juntos y mezclados. La característica *lista de cambios* le ayuda a hacer agrupaciones de ficheros, facilitando ver qué se está haciendo. Por supuesto ésto sólo funciona si los cambios no se superponen. Si dos tareas diferentes afectan al mismo archivo, no hay forma de separar los cambios.

Importante

La característica de lista de cambios en TortoiseSVN sólo está disponible en Windows XP y posterior, puesto que depende de una capacidad del shell que no está presente en Windows

> 2000. Lo sentimos, pero Windows 2000 es ya demasiado antiguo, así que por favor no se queje.

Puede ver las listas de cambios en varios lugares, pero los más importantes son el diálogo de confirmación y el de comprobar modificaciones. Empecemos en el diálogo comprobar modificaciones después de que haya estado trabajando en varias características y varios ficheros. La primera vez que abra el diálogo, todos los ficheros modificados se muestran juntos. Supongamos que ahora quiere organizar las cosas y agrupar esos ficheros según la característica.

Seleccione uno o más ficheros y utilice Menú contextual → Mover a la lista de cambios para añadir un ítem a una lista de cambios. Inicialmente no habrá listas de cambios, por lo que la primera vez que ejecute esto tendrá que crear una nueva lista de cambios. Déle un nombre que describa para qué la está utilizando, y pulse Aceptar. El diálogo de confirmación cambiará para mostrar agrupaciones de ítems.

Una vez que haya creado una lista de cambios puede arrastrar y soltar ítems en ella, tanto desde otra lista de cambios como desde el Explorador de Windows. Arrastrar desde el Explorador puede ser útil ya que le permite añadir ítems a una lista de cambios antes de que el fichero sea modificado. Puede hacer eso desde el diálogo comprobar modificaciones, pero sólo si muestra todos los ficheros no modificados.

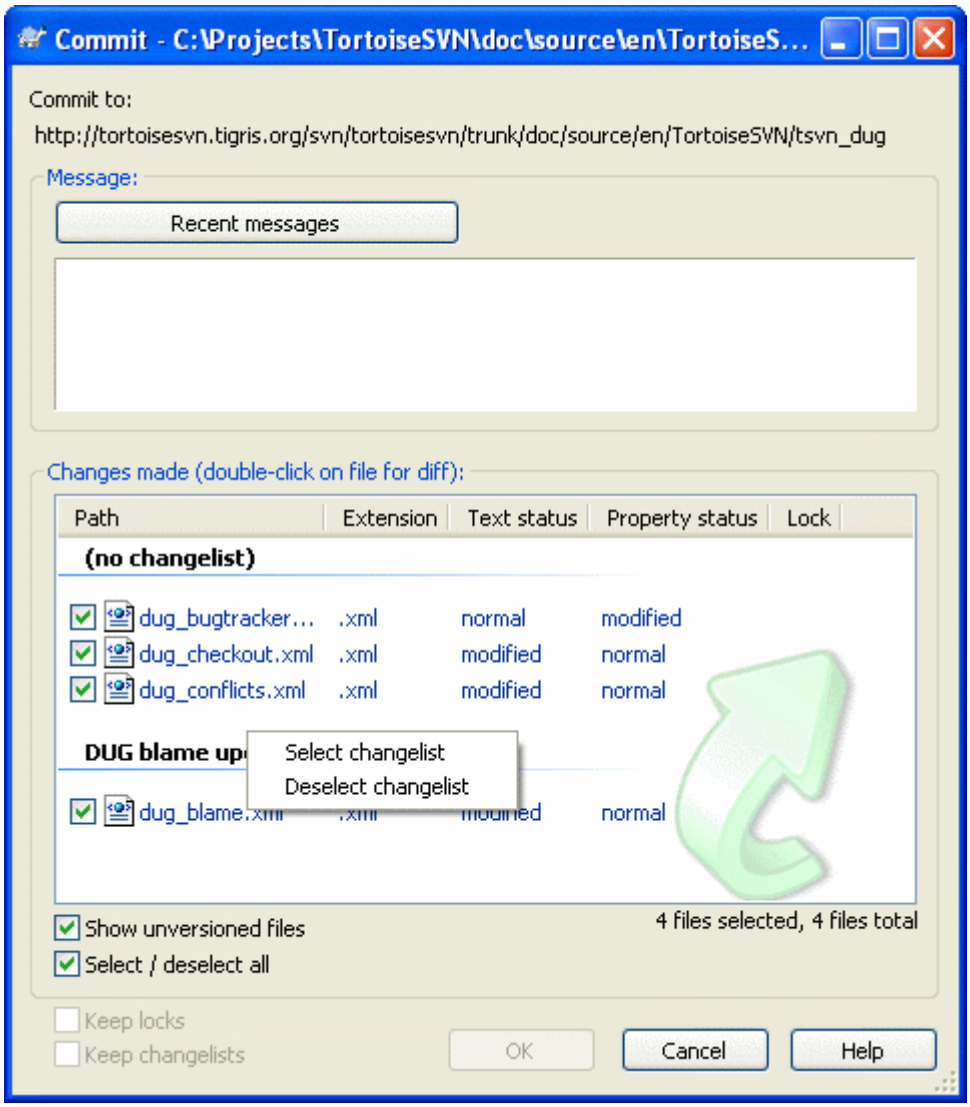


Figura 4.14. Diálogo de confirmación con listas de cambios

En el diálogo de confirmación puede ver esos mismos ficheros, agrupados por listas de cambios. Además de dar una indicación visual inmediata de las agrupaciones, también puede utilizar los encabezados de grupo para seleccionar qué ficheros confirmar.

En XP hay un menú contextual que aparece cuando hace click con el botón derecho en una cabecera de grupo, y que le ofrece la posibilidad de marcar o desmarcar todas las entradas de grupo. En Vista sin embargo el menú contextual no es necesario. Haga click en la cabecera de grupo para seleccionar todas las entradas, y luego marque la casilla de una de las entradas seleccionadas para marcarlas todas.

TortoiseSVN reserva un nombre de lista de cambios para su propio uso, llamada `ignore-on-commit`. Se utiliza para marcar los ficheros versionados que casi nunca querrá confirmar, incluso aunque tengan cambios locales. Esta característica se describe en Sección 4.4.3, “Excluyendo ítems de la lista de confirmación”.

Cuando confirme ficheros que pertenezcan a una lista de cambios, normalmente no es necesario que dichos ficheros sigan perteneciendo a la lista de cambios. Por este motivo, y por defecto, los ficheros se eliminan de las listas de cambios automáticamente al ser confirmados. Si desea mantener el fichero en su lista de cambios, utilice la casilla Mantener listas de cambios en la parte inferior del diálogo de confirmación.

Sugerencia

Las listas de cambios son una característica únicamente del cliente local. La creación y eliminación de listas de cambios no afectan ni al repositorio ni a la copia de trabajo de ningún otro cliente. Simplemente son una forma conveniente para organizar sus ficheros.

4.9. Diálogo de Registro de revisiones

Para cada cambio que haga y confirme, debería proporcionar un mensaje de registro de ese cambio. Así podrá averiguar después qué cambios hizo y por qué, y tendrá un registro detallado para su proceso de desarrollo.

El diálogo de Registro de revisiones recopila todos esos mensajes de registro y se los enseña. La pantalla se divide en tres paneles.

- El panel superior le muestra una lista de revisiones donde se confirmaron cambios a los ficheros/carpetas. Este sumario incluye la fecha y la hora, la persona que confirmó la revisión y el inicio del mensaje de registro.

  Las líneas azules indican que algo se ha copiado a esta línea de desarrollo (quizás desde una rama).

- El panel medio le muestra el mensaje de registro completo para la revisión seleccionada.

- El panel inferior le muestra una lista de todos los ficheros y carpetas que se cambiaron como parte de la revisión seleccionada.

Pero hace mucho más que eso - le proporciona comandos del menú contextual que puede utilizar para obtener aún más información de la historia del proyecto.

4.9.1. Invocando el diálogo de Registro de revisiones

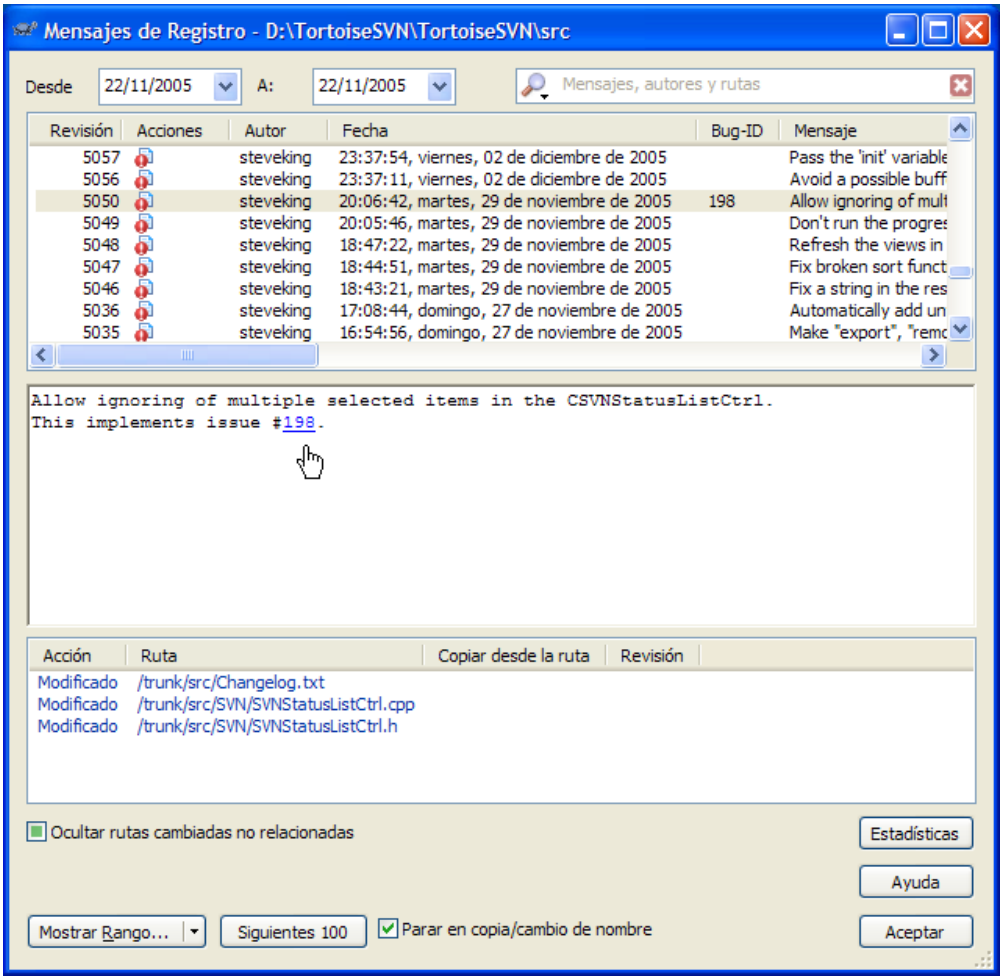


Figura 4.15. El diálogo de Registro de revisiones

Hay varios lugares desde los que puede mostrar el diálogo de Registro:

- Desde el submenú contextual de TortoiseSVN
- Desde la página de propiedades
- Desde el diálogo de Progreso después de que termine una actualización. En ese caso el diálogo de Registro sólo le mostrará aquellas revisiones que cambiaron desde su última actualización

Si el repositorio no está disponible verá el diálogo ¿Desea trabajar sin conexión?, descrito en Sección 4.9.10, “Modo sin conexión”.

4.9.2. Acciones del registro de revisiones

El panel superior tiene una columna Acciones que contiene iconos que resumen qué se ha hecho en esa revisión. Hay cuatro iconos diferentes, cada uno mostrado en su propia columna.

Si una revisión modificó un fichero o un directorio, se muestra el icono *modificado* en la primera columna.

Si en una revisión se añadió un fichero o directorio, se muestra el icono *añadido* en la segunda columna.

Si en una revisión se eliminó un fichero o directorio, se muestra el icono *eliminado* en la tercera columna.

Si una revisión reemplazó un fichero o un directorio, se muestra el icono *reemplazado* en la cuarta columna.

4.9.3. Obteniendo información adicional

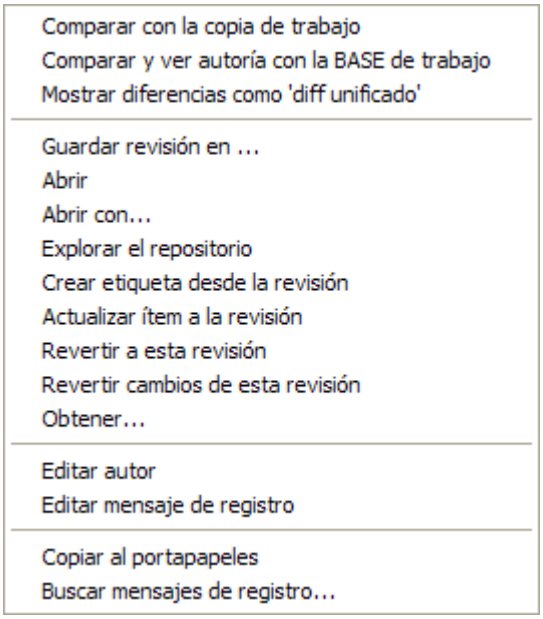


Figura 4.16. El panel superior del diálogo de Registro de revisiones con el menú contextual

El panel superior del diálogo de Registro tiene un menú contextual que le permite acceder a mucha más información. Algunas de las entradas de este menú aparecen sólo cuando se muestra el registro de un fichero, y algunas sólo cuando se muestra el registro de una carpeta.

Comparar con la copia de trabajo
: Comparar las revisiones seleccionadas con su copia de trabajo. La Herramienta de Diferencias por defecto es TortoiseMerge que se proporciona con TortoiseSVN. Si el diálogo de registro es de una carpeta, esto mostrará una lista de sus ficheros cambiados, y le permitirá revisar los cambios hechos a cada fichero individualmente.

Comparar y mostrar autoría con la BASE de trabajo
: Obtener la información de autoría de la revisión seleccionada, y del fichero en su BASE de trabajo, y comparar los resultados utilizando una herramienta visual de diferencias. Lea Sección 4.23.2, “Autoría de las diferencias” para más detalles. (Sólo para ficheros).

Mostrar cambios como diff unificado
: Ver los cambios hechos en la revisión seleccionada como fichero de diff unificado (formato de parche GNU). Esto le muestra sólo las diferencias con unas pocas líneas de contexto. Es más dificil de leer que una comparación visual de ficheros, pero le mostrará todos los cambios juntos en un formato compacto.

Comparar con la revisión anterior
: Compara la revisión seleccionada con la revisión anterior. Esto funciona de forma similar a comparar con su copia de trabajo. Para carpetas esta opción mostrará primero el diálogo de ficheros cambiados permitiéndole seleccionar los ficheros a comparar.

Comparar y mostrar autoría con la revisión anterior
: Muestra el diálogo de ficheros cambios permitiéndole seleccionar ficheros. Obtiene la información de autoría de la revisión seleccionada, y de la revisión anterior, y compara los resultados utilizando una herramienta visual de diferencias (sólo para carpetas).

Guardar revisión en...
: Almacenar la revisión seleccionada en un fichero, para que pueda tener una versión antigua de ese fichero. (Sólo para ficheros).

Abrir / Abrir con...
: Abrir el fichero seleccionado, bien con el visor por defecto para ese tipo de fichero, o bien con el programa que elija. (Sólo para ficheros).

Autoría...
: Muestra la autoría del fichero hasta la revisión seleccionada (sólo para ficheros).

Explorar el repositorio
: Abrir el navegador de repositorios para examinar el fichero o la carpeta seleccionados en el repositorio tal y como estaban en la revisión seleccionada.

Crear rama/etiqueta desde la revisión
: Crea una rama o una etiqueta desde la revisión seleccionada. Esto es útil por ejemlpo si se le olvidó crear una etiqueta y ya ha confirmado algunos cambios que se supone que no deben ir en esa versión.

Actualizar ítem a la revisión
: Actualizar su copia de trabajo a la revisión seleccionada. Útil si quiere hacer que su copia de trabajo refleje un momento en el pasado, o si ha realizado confirmaciones posteriores en el repositorio y quiere actualizar su copia de trabajo un paso cada vez. Es mejor actualizar un directorio completo en su copia de trabajo, no sólo un fichero, ya que si no su copia de trabajo podría ser inconsistente.

  Si desea deshacer un cambio anterior de forma permanente, utilice Revertir a esta revisión

Revertir a esta revisión
: Revierte a una revisión anterior. Si ha hecho varios cambios, y luego decide que realmente desea volver a dejar las cosas como estaban en la revisión N, este es el comando que necesita. Los cambios se revierten en su copia de trabajo, por lo que esta operación *no* afecta al repositorio hasta que confirme los cambios. Tenga en cuenta que esto deshará *todos* los cambios que se hayan realizado tras la revisión seleccionada, reemplazando el fichero o la carpeta con la versión anterior.

  Si su copia de trabajo está en un estado no modificado, después de realizar esta acción su copia de trabajo se mostrará como modificada. Si ya tiene cambios locales, este comando fusionará los cambios *de deshacer* en su copia de trabajo.

  Lo que ocurre internamente es que Subversion realiza una fusión inversa de todos los cambios realizados después de la revisión seleccionada, deshaciendo el efecto de esas confirmaciones sobre las previas.

  Si después de realizar esta acción decide que desea *deshacer lo deshecho* y volver a obtener su copia de trabajo a su estado previo siin modificaciones, deberá utilizar TortoiseSVN → Revertir desde el Explorador de Windows, lo que descartará los cambios locales hechos por esta acción de fusión inversa.

  Si desea simplemente ver cómo estaba un fichero o una carpeta en una revisión anterior, utilice Actualizar a la revisión o Guardar revisión como....

Revertir los cambios hechos en esta revisión
: Deshace los cambios que se hicieron en la revisión seleccionada. Estos cambios se deshacen en su copia de trabajo, ¡por lo que esta operación *no* afecta al repositorio en absoluto! Tenga en cuenta que

esto deshará únicamente los cambios hechos en esa revisión; no reemplaza su copia de trabajo con el fichero completo tal y como estaba en la revisión anterior. Esto es muy útil para deshacer un cambio anterior cuando ya se han hecho otros cambios que no tienen que ver con él.

Si su copia de trabajo está en un estado no modificado, después de realizar esta acción su copia de trabajo se mostrará como modificada. Si ya tiene cambios locales, este comando fusionará los cambios *de deshacer* en su copia de trabajo.

Lo que ocurre internamente es que Subversion realiza una fusión inversa de esa única revisión, deshaciendo sus efectos sobre la confirmación anterior.

Puede *deshacer lo deshecho* tal y como se describe arriba en Revertir a esta revisión.

Fusionar revisión en...
: Fusiona la(s) revision(es) seleccionada(s) en una copia de trabajo diferente. Un diálogo de selección de carpeta le permitirá elegir la copia de trabajo donde desea fusionar, pero después de eso no hay diálogo de confirmación, ni oportunidad de probar la fusión sin ejecutarla realmente. Es una buena idea fusionar en una copia de trabajo sin cambios, ¡y así poder revertir los cambios si no funcionan! Esta es una funcionalidad útil si desea fusionar las revisiones seleccionadas de una rama en otra.

Obtener...
: Hace una obtención nueva de la carpeta seleccionada en la revisión seleccionada. Esto muestra un diálogo para que confirme la URL y la revisión, y seleccione un lugar para la obtención.

Exportar...
: Exporta la carpeta o el fichero seleccionado en la revisión seleccionada. Esto muestra un diálogo para que confirme la URL y la revisión, y para que seleccione un lugar para la exportación.

Editar autor / mensaje de registro
: Editar el mensaje de registro o el autor adjunto a una confirmación anterior. Lea Sección 4.9.7, “Cambiando el mensaje de registro y el autor” para averiguar cómo funciona esto.

Mostrar propiedades de la revisión
: Ver y editar cualquier propiedad de revisión, no sólo el mensaje de registro y el autor. Lea Sección 4.9.7, “Cambiando el mensaje de registro y el autor”.

Copiar al portapapeles
: Copia los detalles de registro de las revisiones seleccionadas al portapapeles. Esto copia el número de revisión, el autor, la fecha, el mensaje de registro y la lista de ítems cambiados para cada revisión.

Buscar mensaje de registro...
: Buscar en los mensajes de registro el texto que desee. Esto busca en los mensajes de registro que ha introducido, y también en los sumarios de acción creados por Subversion (mostrados en el panel inferior). La búsqueda no distingue mayúsculas y minúsculas.

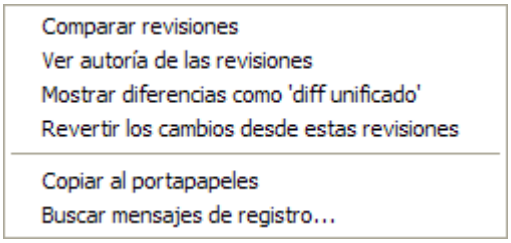


Figura 4.17. Menú contextual del panel superior para 2 revisiones seleccionadas

Si selecciona dos revisiones a la vez (utilizando el modificador habitual **Ctrl**), el menú contextual cambia y le ofrece menos opciones:

Comparar revisiones
: Compara las dos revisiones seleccionadas utilizando una herramienta de diferencias visual. La herramienta de diferencias por defecto es TortoiseMerge que se proporciona con TortoiseSVN.

  Si selecciona esta opción para una carpeta, aparecerá un diálogo posterior mostrando los ficheros cambiados y ofreciéndole más opciones de diferenciación. Lea más sobre el diálogo Comparar Revisiones en Sección 4.10.3, “Comparando carpetas”.

Autoría de las revisiones
: Obtener la información de autoría de las dos revisiones y comparar los resultados utilizando una herramienta visual de diferencias. Lea Sección 4.23.2, “Autoría de las diferencias” para más detalles.

Mostrar diferencias como diff unificado
: Ver las diferencias entre las dos revisiones seleccionadas como un fichero diff unificado. Esto funciona para ficheros y carpetas.

Copiar al portapapeles
: Copie los mensajes de registro al portapapeles tal y como se describió anteriormente.

Buscar mensaje de registro...
: Buscar mensajes de registro como se describe más arriba.

Si selecciona dos o más revisiones (utilizando los modificadores habituales **Ctrl** o **Mayúsculas**), el menú contextual incluirá una entrada para revertir todos los cambios que se hicieron en ese rango de revisiones. Ésta es la forma más sencilla para deshacer un grupo de revisiones de golpe.

También puede elegir fusionar en otra copia de trabajo las revisiones seleccionadas, como se describió más arriba.

Si todas las revisiones seleccionadas tienen el mismo autor, puede editar el autor de todas esas revisiones de un golpe.

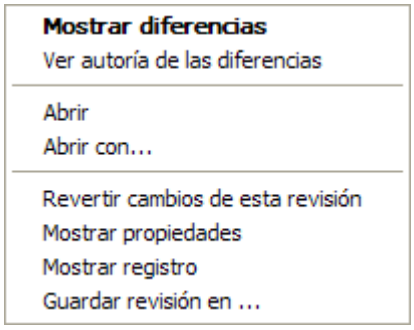


Figura 4.18. El panel inferior del diálogo de Registro con el menú contextual

El panel inferior del diálogo Registro también tiene un menú contextual que le permite

Mostrar cambios
: Mostrar los cambios hechos en la revisión seleccionada sobre el fichero seleccionado. Este menú contextual sólo está disponible para los ficheros que se muestran como *modificados*.

Autoría de los cambios
: Obtener la información de autoría de la revisión seleccionada y de la revisión anterior del fichero seleccionado, y comparar los resultados utilizando una herramienta visual de diferencias. Lea Sección 4.23.2, “Autoría de las diferencias” para más detalles.

Mostrar como diff unificado
: Mostrar los cambios del fichero en formato diff unificado. Este menú contextual sólo está disponible para los ficheros que se muestran como *modificados*.

Abrir / Abrir con...
: Abrir el fichero seleccionado, bien con el visor por defecto para ese tipo de fichero, o bien con el programa que elija.

Autoría...
: Abre el diálogo Autoría, permitiéndole ver la información de autoría hasta la revisión seleccionada.

Revertir los cambios hechos en esta revisión
: Revertir los cambios hechos al fichero seleccionado en esa revisión.

Mostrar propiedades
: Ver las propiedades de Subversion del ítem seleccionado.

Mostrar registro
: Mostrar el registro de revisiones para ese fichero seleccionado.

Obtener registros de fusiones
: Muestra el registro de revisiones para el único fichero seleccionado, incluyendo los cambios fusionados. Más información en Sección 4.9.6, “Características de registro de fusión”.

Guardar revisión en...
: Grabar la revisión seleccionada a un fichero, para que pueda tener una versión antigua de ese fichero.

Sugerencia

Puede haberse dado cuenta de que a veces nos referimos a cambios y otras veces a diferencias. ¿Cuál es la diferencia?

Subversion utiliza números de revisión para dos cosas diferentes. Una revisión generalmente representa el estado de un repositorio en un momento en el tiempo, pero también puede utilizarse para representar el conjunto de cambios que generaron esa revisión; por ejemplo, “Hecho en la r1234” significa que los cambios confirmados en la revisión 1234 implementan la característica X. Para dejar claro en qué sentido se utilizan, empleamos dos términos distintos.

Si selecciona dos revisiones N y M, el menú contextual le ofrecerá mostrar las *diferencias* entre estas dos revisiones. En términos de Subversion esto es `diff -r M:N`.

Si selecciona una única revisión N, el menú contextual le ofrecerá mostrar los *cambios* realizados en esa revisión. En términos de Subversion esto es `diff -r N-1:N` o `diff -c N`.

El panel inferior le muestra los ficheros cambiados en todas las revisiones seleccionadas, por lo que el menú contextual siempre ofrece mostrar *cambios*.

4.9.4. Obteniendo más mensajes de registro

El diálogo Registro no siempre le muestra todos los cambios que se hayan hecho alguna vez por unos cuantos motivos:

- En un repositorio grande, puede haber cientos o incluso miles de cambios, y obtenerlos todos puede llevar mucho tiempo. Normalmente estará interesado sólo en los cambios más recientes. Por defecto, el número de mensajes de registro obtenidos se limita a 100, pero puede cambiar este valor en TortoiseSVN → Configuración (Sección 4.30.1.2, “Configuración de diálogos de TortoiseSVN 1”),

- Cuando se marca la casilla Parar en copia/renombrado, Mostrar Registro se parará en el punto en el que el fichero o carpeta seleccionado se copió de algún otro lugar en el repositorio. Esto puede ser útil para buscar ramas (o etiquetas) porque se para en la raíz de esa rama, y le da una indicación rápida de los cambios hechos únicamente en esa rama.

Normalmente querrá dejar esta opción sin marcar. TortoiseSVN recuerda el estado de la casilla, por lo que respetará su preferencia.

Cuando se invoca el diálogo Mostrar Registro desde el diálogo Fusionar, la casilla siempre se marca por defecto. Esto es porque al fusionar lo más probable es buscar cambios en las ramas, y retroceder a la raíz de la rama no tiene sentido en ese caso.

Tenga en cuenta que Subversion actualmente implementa el renombrado como un par de copia/borrado, por lo que renombrar un fichero o carpeta también provocará que el diálogo de registro se pare si se marca esta opción.

Si desea ver más mensajes de registro, pulse Siguientes 100 para obtener los siguientes 100 mensajes de registro. Puede repetir ésto tantas veces como sea necesario.

Al lado de este botón hay un botón multifunción que recuerda la última opción que utilizó. Pulse en la flecha para ver las otras opciones ofrecidas.

Utilice Mostrar rango ... si desea ver un rango específico de revisiones. Aparecerá un diálogo que le preguntará por la revisión de inicio y de fin.

Utilice Mostrar Todos si desea ver *todos* los mensajes de registro desde HEAD hasta la revisión 1.

4.9.5. Revisión actual de la copia de trabajo

Dado que el diálogo de registro le muestra el registro desde HEAD, no desde la revisión actual de la copia de trabajo, a veces ocurre que se muestran mensajes de registro para contenidos que aún no han sido actualizados en su copia de trabajo. Para ayudar a mostrar esto más claro, el mensaje de confirmación que corresponde a la revisión que tiene en su copia de trabajo se muestra en negrita.

Cuando se muestra el registro de una carpeta, la revisión resaltada es la máxima revisión encontrada dentro de la carpeta, lo cual requiere una busqueda en la copia de trabajo. Esto puede ser una operación lenta cuando se cuenta con muchas copias de trabajo, y el registro no se muestra hasta que se haya completado la operación. Si desea deshabilitar o limitar esta característica debera asignar un valor a la clave de registro`HKCU\Software\TortoiseSVN\RecursiveLogRev` tal como se describe en Sección 4.30.10, “Configuraciones del registro”.

4.9.6. Características de registro de fusión

Subversion 1.5 y posteriores almacenan un registro de las fusiones utilizando propiedades. Esto nos permite obtener una historia más detallada de los cambios fusionados. Por ejemplo, si desarrolla una nueva característica en una rama y luego la fusiona en el tronco, la característica desarrollada aparecerá en el historial de registro del tronco como una única confirmación para la fusión, incluso aunque haya habido 1000 confirmaciones durante el desarrollo de la rama.

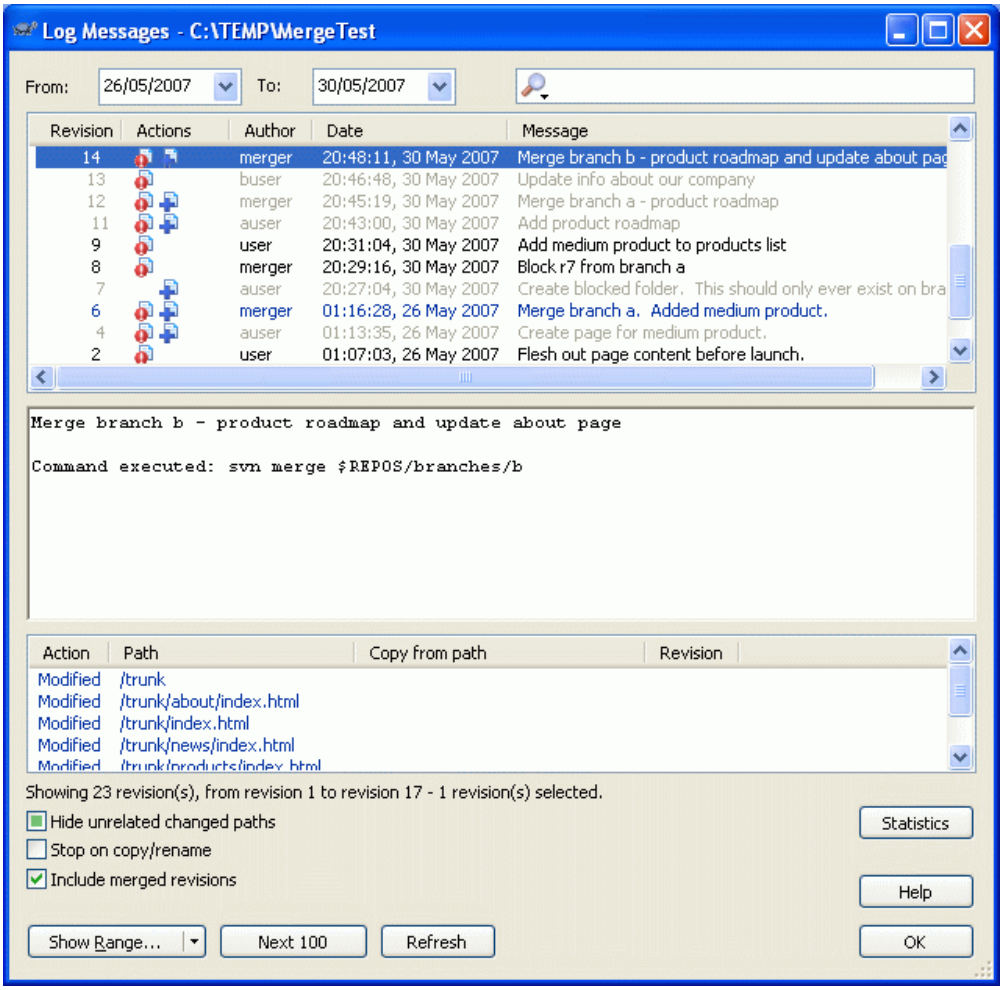


Figura 4.19. El diálogo de registro mostrando revisiones con registro de fusión

Si desea ver el detalle de qué revisiones se fusionaron como parte de esa confirmación, utilice la casilla Incluir revisiones fusionadas. Esto obtendrá los mensajes de registro de nuevo, pero también intercalará los mensajes de registro desde las revisiones que se fusionaron. Las revisiones fusionadas se muestran en gris porque representan cambios hechos en una parte diferente del árbol.

¡Por supuesto que fusionar nunca es sencillo! Durante el desarrollo de la rama habrá probablemente fusiones ocasionales desde el tronco para mantener la rama en sincronía con la línea principal del código. Por eso la historia de fusiones de la rama también incluirá otra capa de historia de fusiones. Estas capas se muestran en el diálogo de registro utilizando niveles de indentación.

4.9.7. Cambiando el mensaje de registro y el autor

Las propiedades de revisión son completamente diferentes de las propiedades Subversion de cada ítem. Las propiedades de revisión (revprops) son ítems descriptivos que están asociados con un número de revisión en concreto en el repositorio, como el mensaje de registro, la fecha de confirmación o el nombre del confirmador (autor).

A veces querrá cambiar un mensaje de registro que introdujo en su día, quizás porque hay un error ortográfico en él o porque quiere mejorar el mensaje o cambiarlo por otras razones. O quizás quiera cambiar el autor de una confirmación porque se le olvidó preparar la autentificación, o ...

Subversion le permite cambiar las propiedades de la revisión en cualquier momento. Pero como estos cambios no se pueden deshacer (estos cambios no se versionan), esta característica está deshabilitada por defecto. Para hacer que ésto funcione, debe preparar un gancho pre-revprop-

change. Por favor consulte el capítulo sobre *Scripts gancho* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.create.html#svn.reposadmin.create.hooks] en el Libro de Subversion para tener más detalles sobre cómo hacerlo. Lea Sección 3.3, “Scripts gancho en el lado del servidor” para encontrar más notas sobre cómo implementar ganchos en una máquina Windows.

Una vez que haya preparado su servidor con los ganchos necesarios, puede cambiar tanto el autor como el mensaje de registro (o cualquier otra propiedad de revisión) de cualquier revisión, utilizando el menú contextual del panel superior del diálogo Registro. También puede editar un mensaje de registro utilizando el menú contextual del panel del medio.

Aviso

Dado que las propiedades de revisión de Subversion no se versionan, al hacer modificaciones a estas propiedades (por ejemplo, la propiedad del mensaje `svn:log`) se sobreescribirá el valor anterior de esa propiedad *para siempre*.

4.9.8. Filtrando los mensajes de registro

Si desea restringir los mensajes de registro para mostrar sólo en los que está interesado en vez de tener que navegar en una lista de cientos, puede utilizar los controles de filtro en la parte superior del Diálogo de Registro. Los controles de fecha de inicio y de fin le permite restringir la salida a un rango de fechas conocido. La caja de texto de búsqueda le permite mostrar sólo los mensajes que contengan una frase en particular.

Pulse en el icono de búsqueda para seleccionar sobre qué información desea buscar, y para seleccionar el modo *regex*. Normalmente sólo necesitará una búsqueda de texto simple, pero si necesita utilizar términos de búsqueda más flexibles, puede utilizar expresiones regulares. Si mueve el ratón encima de la caja, aparecerá un texto de ayuda que le proporcionará pistas sobre cómo utilizar las funciones regex. También puede encontrar documentación en línea y un tutorial en *http://www.regular-expressions.info/*. El filtro funciona comprobando si su cadena de filtro concuerda con las entradas de registro, y de ellas sólo se muestran aquellas entradas que *concuerdan* con la cadena de filtro.

Para hacer que el filtro muestre todas las entradas de registro que *no* concuerdan con la cadena del filtro, comience la cadena con un signo de exclamación ('!'). Por ejemplo, una cadena de filtro `!nombredeusuario` mostrará sólo aquellas entradas que no fueron confirmadas por `nombredeusuario`.

Tenga en cuenta que estos filtros actúan sobre los mensajes ya obtenidos. Ellos no controlan la descarga de mensajes desde el repositorio.

También puede filtrar los nombres de las rutas en el panel inferior utilizando la casilla Ocultar rutas cambiadas no relacionadas. Las rutas relacionadas son aquellas que contienen la ruta utilizada para mostrar el registro. Si está obteniendo el registro de una carpeta, eso significa todo lo que esté en esa carpeta o debajo de ella. Para un fichero, significa sólo ese fichero. La casilla tiene tres estados: puede mostrar todas las rutas, poner en gris las rutas no relacionadas, u ocultar esas rutas completamente.

A veces sus prácticas de trabajo requerirán que los mensajes de registro sigan un formato particular, lo que significa que el texto que describe los cambios no es visible desde el sumario abreviado mostrado en el panel superior. La propiedad `tsvn:logsummary` puede utilizarse para extraer una porción del mensaje de registro que se mostrará en el panel superior. Lea Sección 4.17.2, “Propiedades de projecto TortoiseSVN” para saber cómo se utiliza esta propiedad.

Ningún formato de registro desde el navegador de repositorios

Dado que el formateo depende del acceso a propiedades de Subversion, sólo verá los resultados cuando utilice una copia de trabajo obtenida. Acceder a las propiedades de forma

> remota es una operación lenta, por lo que no verá esta característica desde el navegador de repositorios.

4.9.9. Información estadística

El botón Estadísticas lanza un cuadro de diálogo que muestra algunas informaciones interesantes sobre las revisiones que se muestran en el diálogo Registro. Le muestra cuántos autores han estado trabajando, cuántas confirmaciones han hecho, el progreso por semanas, y mucho más. Ahora puede ver de un vistazo quién ha trabajado duro y quién se ha tocado la barriga ;-)

4.9.9.1. Página de estadísticas

Esta página le proporciona todas las cifras que pueda necesitar, en particular el período y el número de revisiones cubiertas, y algunos valores mínimos/máximos/medios.

4.9.9.2. Página de confirmaciones por autor

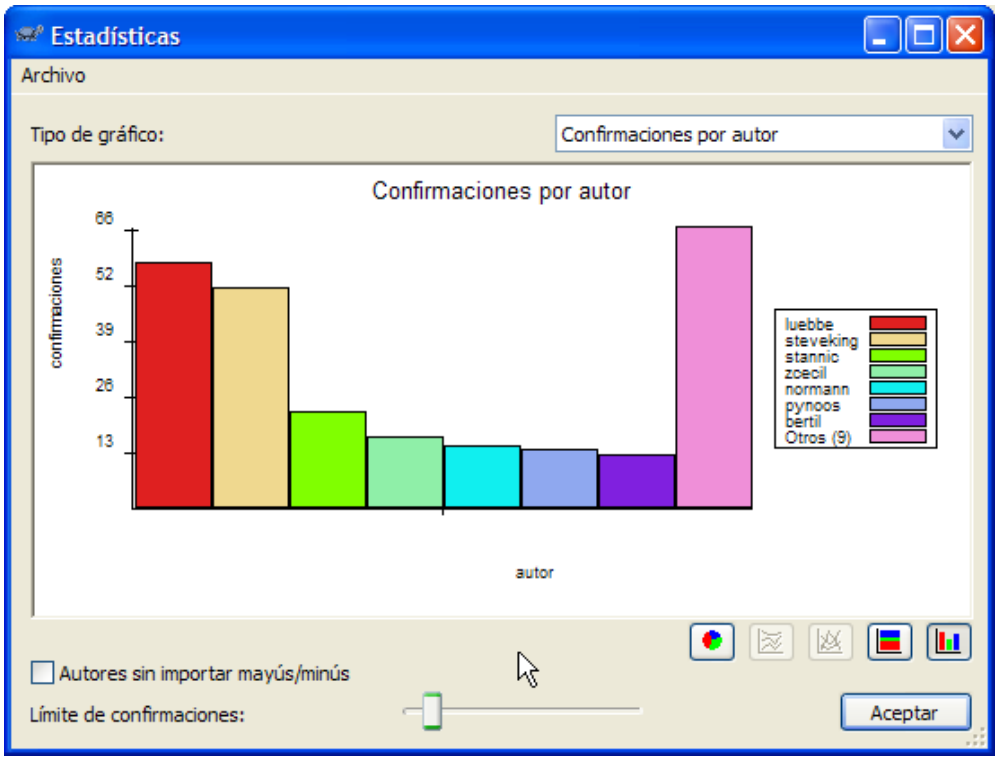


Figura 4.20. Histograma de confirmaciones por autor

Este gráfico le muestra qué autores han estado activos en el proyecto como un simple histograma, un histograma apilado o un gráfico de tarta.

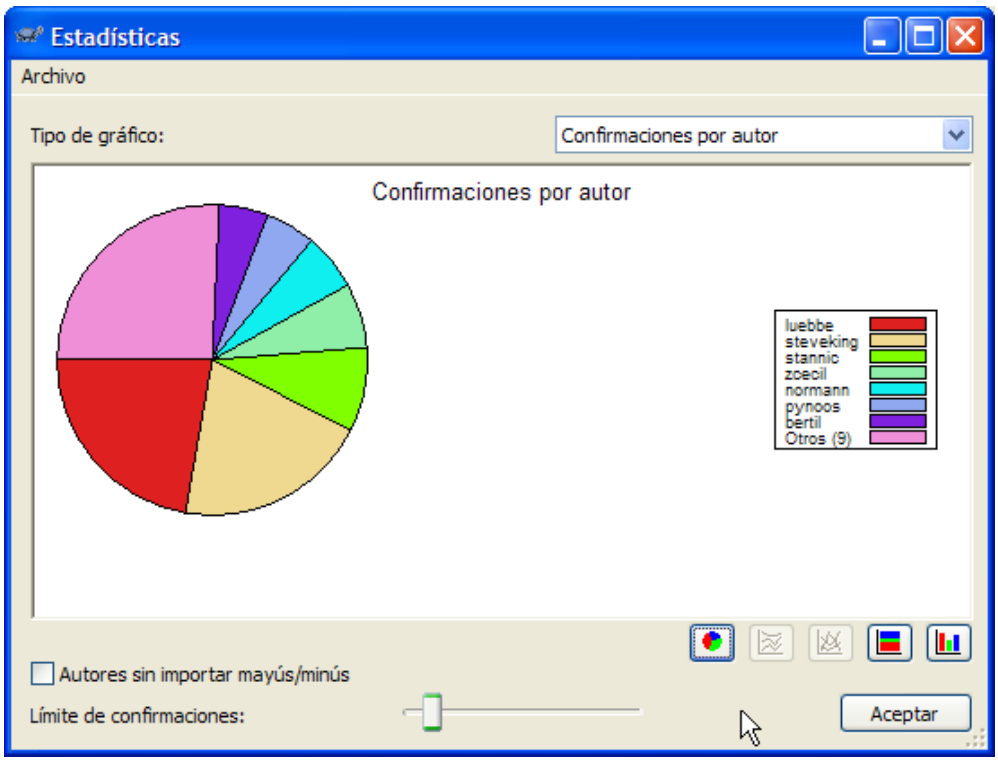


Figura 4.21. Gráfico de tarta de confirmaciones por autor

Cuando hay unos pocos autores muy activos y muchos pequeños contribuyentes, el número de segmentos pequeños puede hacer que el gráfico sea más dificil de leer. El selector deslizante en la parte inferior le permite establecer un límite (el porcentaje sobre el total de confirmaciones) bajo el cual cualquier actividad se agrupa en una categoría *Otros*.

4.9.9.3. Página de confirmaciones por fecha

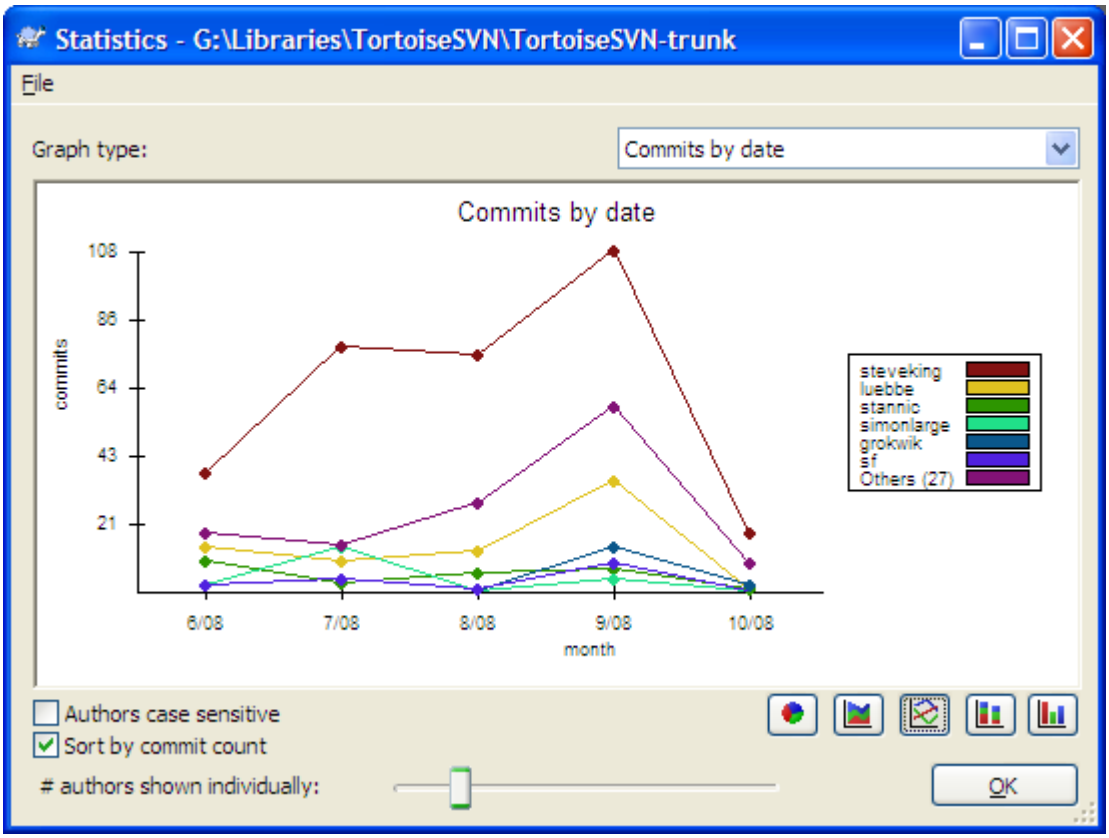


Figura 4.22. Gráfico de confirmaciones por fecha

Esta página proporciona una representación gráfica de la actividad del proyecto en términos del número de confirmaciones *y* autor. Esto da una idea de cuándo se ha trabajado en un proyecto, y quién estaba trabajando en cada momento.

Cuando hay varios autores, puede obtener muchas líneas en el gráfico. Hay dos vistas disponibles aquí: *normal*, en la que la actividad de cada autor se refiere a la línea base, y *apilado*, donde la actividad de cada autor se refiere a la línea subyacente. La última opción evita que se crucen las líneas, lo que puede permitir un gráfico más sencillo de leer, pero es menos fácil ver la salida de cada autor.

Por defecto el análisis distingue mayúsculas y minúsculas, por lo que los usuarios `PeterEgan` y `PeteRegan` se tratan como autores diferentes. Sin embargo, en muchos casos los nombres de usuario no distinguen mayúsculas y minúsculas, y a veces se introducen de forma inconsistente, por lo que puede querer que `DavidMorgan` y `davidmorgan` se traten como la misma persona. Utilice la casilla Autores sin importar mayús/minús para controlar este comportamiento.

Tenga en cuenta que las estadísticas cubren el mismo período que el diálogo Registro. Si sólo se está mostrando una revisión, las estadísticas no le dirán mucho.

4.9.10. Modo sin conexión

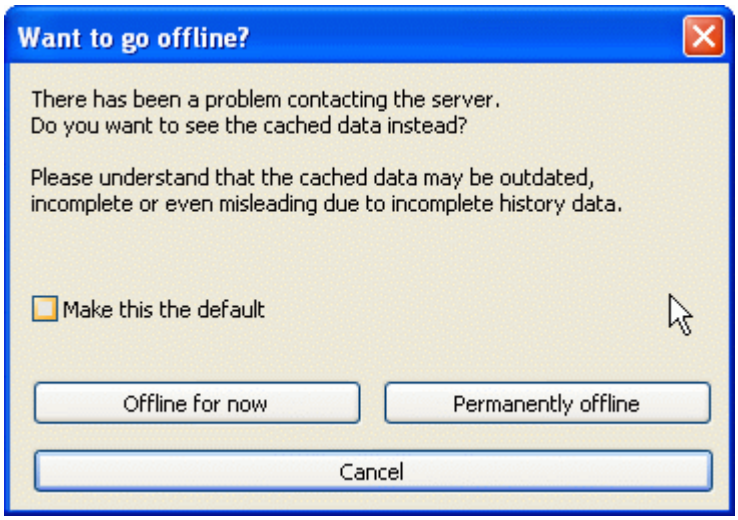


Figura 4.23. Modo sin conexión

Si no se puede alcanzar el servidor, y tiene habilitado el caché de registro, puede utilizar el diálogo de registro y el gráfico de revisiones en modo sin conexión. Esto utiliza la información de la caché, lo que le permite continuar trabajando aunque la información puede no estar actualizada o incluso completa.

Existen tres opciones:

Sin conexión por ahora
: Completar la operación actual en modo sin conexión, pero reintentar el repositorio la próxima vez que se pida la información de registro.

Permanentemente sin conexión
: Permanece en el modo sin conexión hasta que se pida específicamente una comprobación de repositorio. Vea Sección 4.9.11, “Refrescando la vista”.

Cancelar
: Si no desea continuar la operación con datos posiblemente antiguos, simplemente cancele.

La casilla Establecer como por defecto evita que este diálogo aparezca de nuevo y siempre utiliza la opción que seleccione a continuación. Siempre podrá cambiar (o eliminar) la opción por defecto tras esta operación desde TortoiseSVN → Configuración.

4.9.11. Refrescando la vista

Si desea comprobar el servidor de nuevo para obtener los mensajes de registro más recientes, puede simplemente refrescar la vista utilizando **F5**. Si está utilizando la caché de registro (habilitada por defecto), esto buscará en el repositorio los mensajes más recientes y traerá sólo los nuevos. Si la caché de registro estaba en modo desconectado, esto intentará volver a ponerla en línea.

Si está utilizando la caché de registros y cree que el contenido del mensaje o el autor han podido cambiar, puede utilizar **Mays-F5** o **Ctrl-F5** para re-obtener los mensajes mostrados desde el servidor y actualizar la caché de registro. Tenga en cuenta que esto sólo afecta a los mensajes que se están mostrando actualmente y que por tanto no invalida la caché completa de ese repositorio.

4.10. Viendo diferencias

Uno de los requisitos más comunes en el desarrollo de proyectos es ver qué ha cambiado. Puede querer ver las diferencias entre dos revisiones del mismo fichero, o las diferencias entre dos ficheros separados. TortoiseSVN provee una herramienta integrada llamada TortoiseMerge para ver las diferencias entre ficheros de texto. Para ver las diferencias entre ficheros de imagen, TortoiseSVN también tiene una herramienta llamada TortoiseIDiff. Por supuesto, puede utilizar su herramienta de diferencias favorita si lo desea.

4.10.1. Diferencias de ficheros

Cambios locales
: Si desea ver qué cambios ha hecho *usted* en su copia de trabajo, simplemente utilice el menú contextual del explorador y seleccione TortoiseSVN → Diferenciar.

Diferenciar con otra rama/etiqueta
: Si desea ver qué ha cambiado en el tronco (si está trabajando en una rama) o en una rama específica (si está trabajando en el tronco), puede utilizar el menú contextual del explorador. Simplemente sostegna la tecla **Mayúsculas** mientras hace click con el botón derecho en el fichero. Luego, seleccione TortoiseSVN → Diferenciar con URL. En el siguiente diálogo, especifique la URL del repositorio con la que quiere comparar su fichero local.

  También puede utilizar el navegador de repositorios y seleccionar dos árboles para diferenciar, quizás dos ramas, o una rama/etiqueta y el tronco. Ahí, el menú contextual le permite compararlos utilizando Comparar revisiones. Lea más en Sección 4.10.3, “Comparando carpetas”.

Diferenciar desde una revisión anterior
: Si desea ver las diferencias entre una revisión en concreto y su copia de trabajo, utilice el diálogo Registro de Revisiones, seleccione la revision de intereés, y luego seleccione Compar con la copia de trabajo desde el menú contextual.

  Si desea ver la diferencia entre la última revision confirmada y su copia de trabajo, asumiendo que la copia de trabajo no se haya modificado, simplemente haga click con el botón derecho sobre el fichero. Luego, seleccione TortoiseSVN → Comparar con la revisión anterior. Esto realizará una diferenciación entre la revisión anterior a la fecha-de-la-última-confirmación (tal y como se registró en su copia de trabajo) y la BASE de la copia de trabajo. Esto le muestra el último cambio hecho al fichero que lo llevó al estado que ahora está viendo en su copia de trabajo. No mostrará cambios más nuevos que su copia de trabajo.

Diferenciar entre dos revisiones antiguas
: Si desea ver las diferencias entre dos revisiones que ya se confirmaron, utilice el diálogo Registro de Revisiones y seleccione las dos revisiones que desea comparar (utilizando el modificador habitual **Ctrl**). Luego seleccione Comparar revisiones desde el menú contextual.

  Si hizo esto desde el historial de revisiones de una carpeta, aparece un diálogo Comparar Revisiones, mostrando una lista de ficheros cambiados en esa carpeta. Lea más en Sección 4.10.3, “Comparando carpetas”.

Todos los cambios hechos en una confirmación
: Si desea ver los cambios hechos a todos los ficheros en una revisión en particular de una vez, puede utilizar la salida diff unificado (formato de parche GNU). Esto le muestra sólo las diferencias con unas pocas líneas de contexto. Es más dificil de leer que una comparación visual de ficheros, pero le mostrará todos los cambios juntos. Desde el diálogo Registro de Revisiones, seleccione la revisión de interes, y luego seleccione Mostrar Diferencias como Diff Unificado desde el menú contextual.

Diferencias entre ficheros
: Si desea ver las diferencias entre dos ficheros diferentes, puede hacerlo directamente en el explorador seleccionando ambos ficheros (utilizando el modificador habitual **Ctrl**). Luego desde el menú contextual del explorador seleccione TortoiseSVN → Diferenciar.

Diferencias entre un fichero/carpeta en la copia de trabajo y una URL
: Si desea ver las diferencias entre un fichero en su copia de trabajo, y un fichero en cualquier repositorio de Subversion, puede hacerlo directamente en el explorador seleccionando el fichero y pulsando la tecla **Mayúsculas** mientras hace click con el botón derecho para obtener el menú contextual. Seleccione TortoiseSVN → Diferenciar con URL. Puede hacer lo mismo para una carpeta de copia de trabajo. TortoiseMerge muestra esas diferencias de la misma forma que muestra un fichero de parche - una lista de ficheros cambiados que puede ver de uno en uno.

Diferencias con información de autoría
: Si desea ver no sólo las diferencias sino también el autor, revisión y la fecha en la que se hicieron los cambios, puede combinar los informes de diferencias y de autoría desde dentro del diálogo del historial de revisiones. Lea Sección 4.23.2, “Autoría de las diferencias” para más detalles.

Diferencias entre carpetas
: Las herramientas proporcionadas con TortoiseSVN no le permiten ver las diferencias entre jerarquías de directorios. Pero si tiene una herramienta externa que soporte esta funcionalidad, puede utilizarla. En Sección 4.10.5, “Herramientas externas de diferencias/fusión” le contamos algunas de las herramientas que hemos utilizado.

Si ha configurado una herramienta de diferenciado de terceros, puede utilizar **Mayúsculas**cuando seleccione el comando Diferenciar para utilizar la herramienta alternativa. Lea Sección 4.30.5, “Configuración de programas externos” para saber más sobre configurar otras herramientas de diferenciación.

4.10.2. Opciones de fín de línea y espacios en blanco

A veces en la vida de un proyecto puede querer cambiar los fines de línea de CRLF a LF, o puede querer cambiar la indentación de una sección. Desafortunadamente esto marcará un gran número de líneas como modificadas, incluso aunque no haya cambios en el significado del código. Estas opciones le ayudarán a administrar estos cambios cuando se comparan y aplican diferencias. Verá estas opciones en los diálogos Fusionar y Autoría, además de en la configuración de TortoiseMerge.

Ignorar finales de línea excluye los cambios que sólo se deban a diferencias en el estilo de fin de línea.

Comparar espacios en blanco incluye todos los cambios en la indentación y en los espacios en blanco interiores como líneas añadidas/eliminadas.

Ignorar cambios en espacios en blanco excluye los cambios que sólo se deben a un cambio en la cantidad o tipo de espacios en blanco, como por ejemplo cambios en la indentación o cambio de tabuladores a espacios. Añadir espacios en blanco donde antes no había, o eliminar completamente espacios en blanco aún se mostrará como un cambio.

Ignorar todos los espacios en blanco excluye todos los cambios que sólo se deban a espacios en blanco.

Naturalmente, cualquier línea cuyo contenido haya cambiado se incluye siempre en la diferenciación.

4.10.3. Comparando carpetas

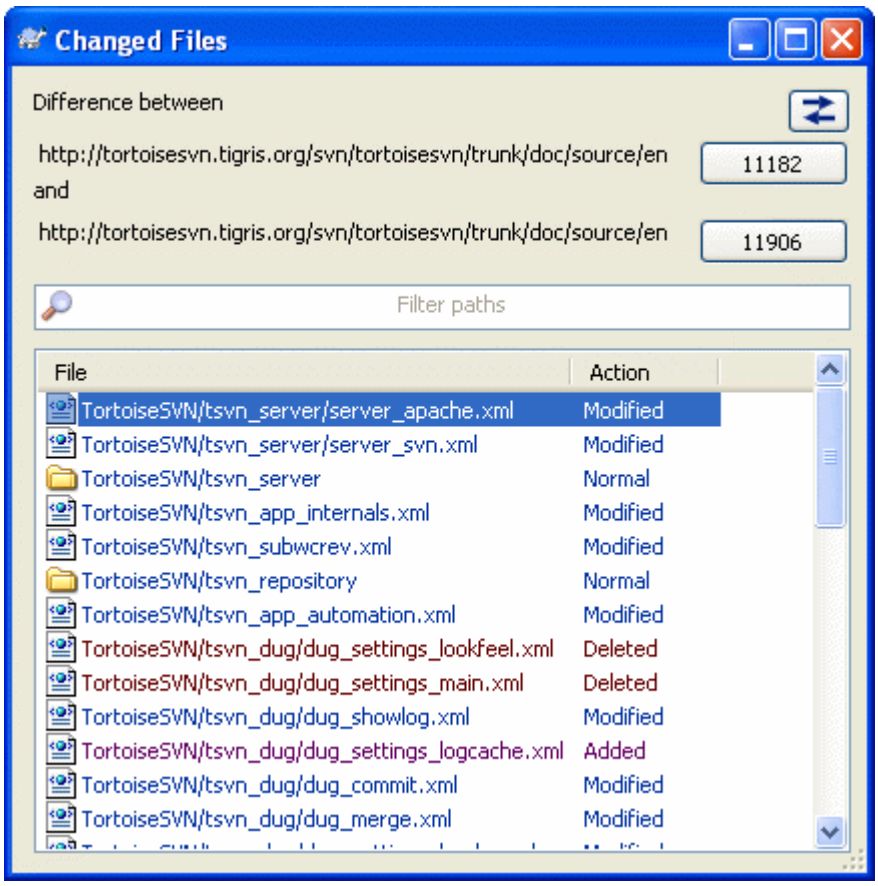


Figura 4.24. El diálogo Comparar Revisiones

Cuando seleccione dos árboles dentro del navegador de repositorios, o cuando seleccione dos revisiones de una carpeta en el diálogo de registro, puede Menú contextual → Comparar revisiones.

Este diálogo muestra una lista de todos los ficheros que han cambiado y le permite comparar o ver la autoría de cada uno individualmente utilizando el menú contextual.

Puede exportar un *árbol de cambios*, lo que es útil si necesita enviar a alguien la estructura de su árbol de proyecto, pero conteniendo sólo los ficheros que han cambiado. Esta operación trabaja sólo sobre los ficheros seleccionados, por lo que necesitará seleccionar los ficheros de interés - normalmente eso significa todos - y luego Menú contextual → Exportar selección a.... Se le preguntará por una ruta donde guardar el árbol de cambios.

También puede exportar la *lista* de ficheros cambiados a un fichero de texto utilizando Menú contextual → Guardar lista de ficheros seleccionados a....

Si desea exportar la lista de ficheros *y también* las acciones (modificado, añadido, borrado), puede hacerlo utilizando Menú contextual → Copiar la selección al portapapeles.

El botón en la parte superior le permite cambiar la dirección de la comparación. Puede mostrar los cambios necesarios para ir de A a B, o si lo prefiere, de B a A.

Los botones con los números de revisión pueden utilizarse para cambiar a un rango de revisiones diferente. Cuando cambia el rango, la lista de ítems que difieren entre las dos revisiones se actualizará automáticamente.

Si la lista de nombres de ficheros es muy larga, puede utilizar la caja de búsqueda para reducir la lista a los nombres de ficheros que contengan un texto específico. Tenga en cuenta que se utiliza una búsqueda

de texto simple, así que si desea restringir la lista a ficheros de código fuente C debería introducir `.c` en vez de `*.c`.

4.10.4. Diferenciando imágenes utilizando TortoiseIDiff

Hay muchas herramientas disponibles para diferenciar ficheros de texto, incluyendo nuestro propio TortoiseMerge, pero a veces nos encontramos en la situación de querer ver también cómo ha cambiado un fichero de imagen. Por eso es por lo que hemos creado TortoiseIDiff.

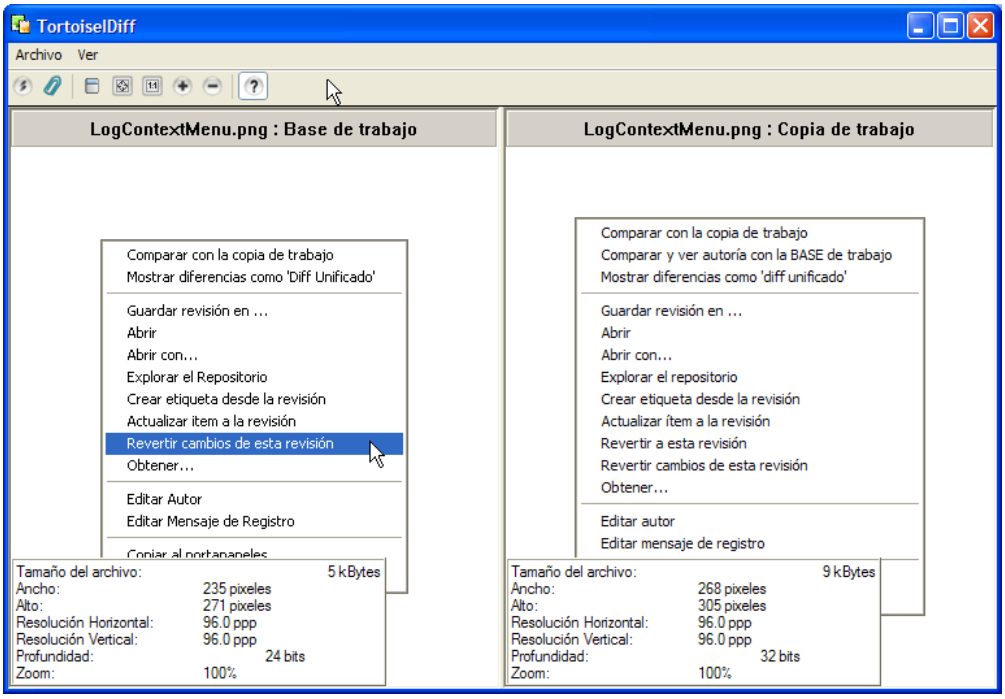


Figura 4.25. El visor de diferencias de imágenes

TortoiseSVN → Diff para cualquiera de los formatos de imágenes comunes iniciará TortoiseIDiff para mostrar las diferencias entre las imágenes. Por defecto las imágenes se muestran una al lado de la otra, pero puede utilizar el menú Ver o la barra de herramientas para cambiar a una vista arriba-abajo, o si lo prefiere, puede superponer las imágenes y simular que está usando una caja de luces.

Naturalmente, también puede acercar y alejar el zoom y moverse por la imagen. También puede desplazarse por la imagen símplemente arrastrando con el botón izquierdo. Si selecciona la opción Unir las imágenes, los controles de desplazamiento (barras de desplazamiento, rueda del ratón) de ambas imágenes quedan unidos.

Un cuadro de información de imagen le muestra detalles sobre el fichero de imagen, tales como el tamaño en pixeles, la resolución y la profundidad de color. Si la caja le molesta, utilice Ver → Información de imagen para ocultarla. Puede obtener la misma información en una etiqueta de ayuda si pasa el cursor por encima de la barra de título de la imagen.

Cuando las imágenes se superponen, la intensidad relativa de las imágenes (alpha blend) se controla mediante un control deslizante en el lado izquierdo. Puede pulsar en cualquier parte del deslizador para establecer el blend directamente, o puede arrastrar el deslizador para cambiar el blend de forma interactiva. **Ctrl+Mayúsculas**-Rueda para cambiar el blend.

El botón debajo del deslizador cambia entre fusión 0% y 100%, y si hace doble click en el botón, la fusión cambia automáticamente cada segundo hasta que pulse el botón de nuevo. Esto puede ser útil cuando busca múltiples cambios pequeños.

A veces quiere ver una diferencia en vez de un fundido. Puede que tenga las imágenes de dos revisiones de una placa de circuito impreso y desea ver qué pistas han cambiado. Si deshabilita el modo alpha blend, la diferencia se mostrará como un *XOR* de los valores de color del pixel. Las áreas sin cambios se mostrarán en blanco y los cambios se colorearán.

4.10.5. Herramientas externas de diferencias/fusión

Si las herramientas que le proporcionamos no son lo que necesita, pruebe alguno de los muchos programas de código abierto o comerciales disponibles. Cada uno tiene sus propios favoritos, y esta lista por supuesto no está completa, pero aquí hay algunas que debería considerar:

WinMerge
: *WinMerge* [http://winmerge.sourceforge.net/] es un muy buen visor de diferencias y herramienta de fusión de código abierto, que también puede manejar directorios.

Perforce Merge
: Perforce es un RCS comercial, pero puede descargar la herramienta de diferencias/fusiones gratuitamente. Obtenga más información en *Perforce* [http://www.perforce.com/perforce/products/merge.html].

KDiff3
: KDiff3 es una herramienta de diferencias gratuita que también maneja directorios. Puede descargarla desde *aquí* [http://kdiff3.sf.net/].

ExamDiff
: ExamDiff Standard es gratuito. Puede manejar ficheros pero no directorios. ExamDiff Pro es shareware y añade un número de ventajas que incluyen diferenciación de directorios y capacidades de edición. En ambos casos, la versión 3.2 y las siguientes pueden manejar también unicode. Puede descargarlas desde *PrestoSoft* [http://www.prestosoft.com/].

Beyond Compare
: Similar a ExamDiff Pro, esta es una herramienta shareware de diferencias que puede manejar diferenciación de directorios y unicode. Descárgela desde *Scooter Software* [http://www.scootersoftware.com/].

Araxis Merge
: Araxis Merge es una útil herramienta comercial para ver diferencias y fusionar tanto ficheros como carpetas. Hace comparaciones a tres bandas en las fusiones y tiene vínculos de sincronización para utilizar si ha cambiado el orden las funciones. Descárgelo desde *Araxis* [http://www.araxis.com/merge/index.html].

SciTE
: Este editor de texto incluye sintaxis coloreada para ficheros diff unificados, haciéndolos mucho más fáciles de leer. Descárgelo desde *Scintilla* [http://www.scintilla.org/SciTEDownload.html].

Notepad2
: Notepad2 está diseñado como un reemplazo del programa Bloc de Notas estándar de Windows, y se basa en el control de edición de código abierto Scintilla. Además de ser bueno para ver diffs unificados, es mucho mejor que el bloc de notas de Windows para la mayoría de trabajo. Descárgelo gratuitamente *aquí* [http://www.flos-freeware.ch/notepad2.html].

Lea Sección 4.30.5, “Configuración de programas externos” para más información sobre cómo preparar TortoiseSVN para utilizar estas herramientas.

4.11. Añadiendo nuevos ficheros y directorios

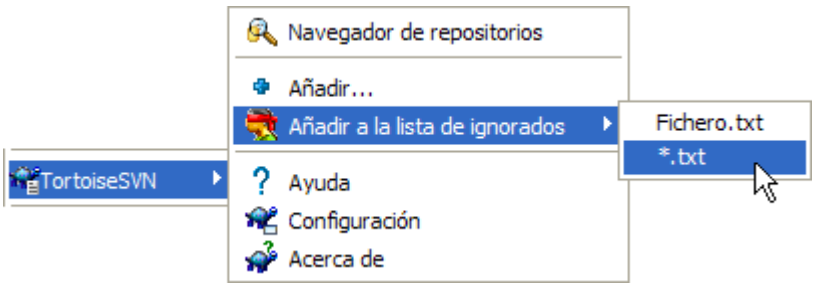


Figura 4.26. Menú contextual del explorador para ficheros no versionados

Si ha creado nuevos ficheros y/o directorios durante su proceso de desarrollo, necesitará añadirlos también al control de código. Seleccione los ficheros y/o directorios y utilice TortoiseSVN → Añadir.

Después de que añada los ficheros o directorios al control de código, el fichero aparece con una sobreimpresión de icono `añadido` que significa que primero debe confirmar su copia de trabajo para que esos ficheros y directorios estén disponibles para otros desarrolladores. ¡Añadir un fichero/directorio *no* afecta al repositorio!

Añadir muchos

También puede usar el comando Añadir en carpetas que ya estén versionadas. En ese caso, el diálogo de añadir le mostrará todos los ficheros sin versionar dentro de esa carpeta versionada. Esto ayuda si tiene muchos ficheros nuevos y necesita añadirlos todos de golpe.

Para añadir ficheros desde fuera de su copia de trabajo puede usar el manejador de arrastrar-y-soltar:

1. seleccione los ficheros que desea añadir
2. arrástrelos con el botón derecho a su nuevo destino dentro de la copia de trabajo
3. suelte el botón derecho del ratón
4. seleccione Menú contextual → SVN Añadir ficheros a esta Copia de Trabajo. En ese momento los ficheros se copiarán a la copia de trabajo y se añadirán al control de versiones.

También puede añadir ficheros dentro de una copia de trabajo simplemente arrastrándolos con el botón izquierdo y soltándolos en el diálogo de confirmación.

Si añade un fichero o carpeta por error, puede deshacer la adición antes de confirmar utilizando TortoiseSVN → Deshacer añadir....

4.12. Copiando/Moviendo/Renombrando ficheros y carpetas

A menudo ocurre que ya tiene los ficheros que necesita en otro proyecto en su repositorio, y que simplemente quiere copiarlos. Podría simplemente copiar los ficheros y añadirlos como se describe más arriba, pero eso no le dará ninguna historia. Y si posteriormente arregla un error en los ficheros originales, sólo podrá fusionar el arreglo automáticamente si la nueva copia está relacionada con la original en Subversion.

La manera más fácil de copiar ficheros y carpetas desde dentro de una copia de trabajo es utilizar el menú contextual que aparece al arrastrar con el botón derecho. Cuando arrastra con el botón derecho un fichero o una carpeta de una copia de trabajo a otra, o incluso dentro de la misma carpeta, aparece un menú contextual cuando suelta el botón del ratón.

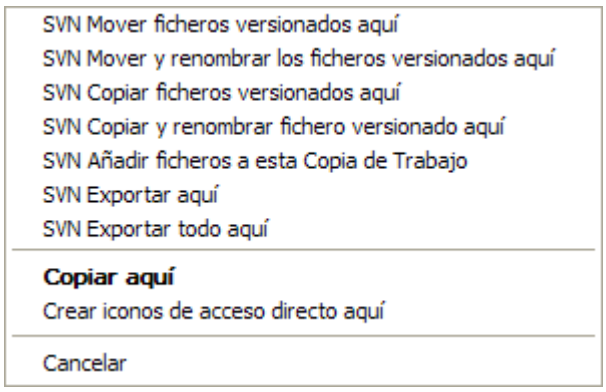


Figura 4.27. Menú de arrastre con el botón derecho para un directorio bajo el control de versiones

Ahora puede copiar contenido existente versionado a un nuevo lugar, posiblemente renombrándolo al mismo tiempo.

También puede copiar o mover ficheros versionados dentro de una copia de trabajo, o entre dos copias de trabajo, utilizando el método familiar corta-y-pega. Utilice el método estándar de Windows Copiar o Cortar para copiar uno o más ítems versionados al portapapeles. Si el portapapeles contiene ítems versionados, puede utilizar TortoiseSVN → Pegar (tenga en cuenta: no es el método estándar de Windows Pegar) para copiar o mover estos ítems al nuevo lugar de la copia de trabajo.

Puede copiar ficheros y carpetas desde su copia de trabajo a otro lugar del repositorio utilizando TortoiseSVN → Rama/Etiqueta. Para saber más, vea Sección 4.19.1, “Crando una rama o etiqueta”.

Puede localizar una versión anterior de un fichero o carpeta en el diálogo de registro y copiarlo a un nuevo lugar del repositorio directamente desde el diálogo de registro utilizando Menú contextual → Crear rama/etiqueta desde la revisión. Para saber más, vea Sección 4.9.3, “Obteniendo información adicional”.

También puede utilizar el navegador de repositorio para localizar el contenido que quiere, y copiarlo en su copia de trabajo directamente desde el repositorio, o copiar entre dos lugares dentro del repositorio. Para más información lea Sección 4.24, “El navegador de repositorios”.

No se puede copiar entre repositorios

Mientras que puede copiar y mover ficheros y carpetas *dentro* de un repositorio, *no puede* copiar o mover desde un repositorio a otro y preservar la historia utilizando TortoiseSVN. Ni siquiera si los repositorios residen en el mismo servidor. Todo lo que puede hacer el copiar el contenido en su estado actual y añadirlo como contenido nuevo en el segundo repositorio.

Si no está seguro de si dos URLs en el mismo servidor se refieren al mismo repositorio o a repositorios distintos, utilice el navegador de repositorios para abrir una de las URLs y averiguar dónde está la raíz del repositorio. Si puede ver ambos lugares en una única ventana del navegador de repositorios, entonces las dos rutas están en el mismo repositorio.

4.13. Ignorando ficheros y directorios

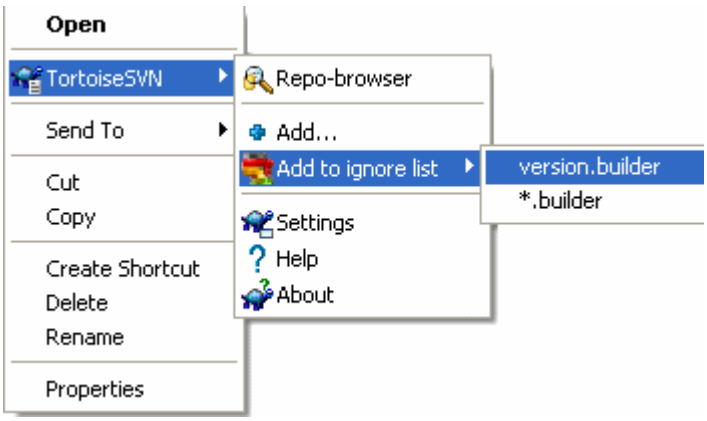


Figura 4.28. Menú contextual del explorador para ficheros no versionados

En la mayoría de los proyectos tendrá ficheros y carpetas que no deberán estar sujetos al control de versiones. Esto puede incluir ficheros creados por el compilador, `*.obj, *.lst`, o quizás una carpeta de salida donde se almacenan los ejecutables. Cuando confirma los cambios, TortoiseSVN le muestra los ficheros no versionados, que rellenan la lista de ficheros del diálogo de confirmar. Por supuesto que puede desactivarlos para que no se muestren, pero entonces quizás se olvide de añadir un nuevo fichero de código fuente.

La mejor forma de evitar estos problemas es añadir los ficheros derivados a la lista de ignorados del proyecto. De esta forma nunca se mostrarán en el diálogo de confirmar, pero se seguirán señalando los genuinos ficheros de código no versionados.

Si hace click con el botón derecho en un único fichero no versionado, y selecciona el comando TortoiseSVN → Ignorar del menú contextual, aparecerá un submenú que le permitirá seleccionar sólo ese fichero, o todos los ficheros con la misma extensión. Si selecciona múltiples ficheros, no hay submenú y sólo puede añadir esos ficheros o carpetas específicos.

Si desea eliminar uno o más ítems de la lista de ignorados, haga click con el botón derecho en dichos ítems y seleccione TortoiseSVN → Eliminar de la Lista de Ignorados También puede acceder a la propiedad `svn:ignore` de la carpeta directamente. Esto le permitirá especificar patrones más generales utilizando expansión de comodines en los nombres de los ficheros, descrito más adelante. Para más información sobre establecer propiedades directamente, lea Sección 4.17, “Configuración del proyecto”. Por favor, tenga en cuenta que cada patrón de ignorar debe escribirse en una nueva línea. No funcionará si los separa por espacios.

La lista global de ignorados

Otra forma de ignorar ficheros es añadirlos a la *lista global de ignorados*. La mayor diferencia aquí es que la lista global de ignorados es una propiedad del cliente. Se aplica a *todos* los proyectos de Subversion, pero sólo en el PC cliente. En general es mejor utilizar la propiedad `svn:ignore` donde sea posible, porque puede aplicarse a áreas del proyecto específicas, y funciona para todos los que obtengan el proyecto. Para más información, lea la Sección 4.30.1, “Configuración general”.

Ignorando ítems versionados

Los ficheros y las carpetas verisonadas nunca pueden ser ignoradas - esta es una característica de Subversion. Si ha versionado un fichero por error, lea Sección B.8, “Ignorar ficheros que ya están versionados” para ver las instrucciones sobre cómo “desversionarlo”.

4.13.1. Concordancia de patrones en las listas de ignorados

El patrón de ignorados de Subversion hace uso de la expansión de comodines en los nombres de los ficheros, una técnica que originariamente se utilizaba en Unix para especificar ficheros utilizando meta-caracteres como son los comodines. Los siguientes caracteres tienen un significado especial:

*
: Concuerda con cualquier cadena de caracteres, incluyendo la cadena vacía (sin caracteres).

?
: Concuerda con un único cáracter cualquiera.

[...]
: Concuerda con cualquiera de los caracteres incluídos entre los corchetes. Dentro de los corchetes, un par de caracteres separados por "-" concuerda con cualquier caracter existente entre los dos en el orden lexicográfico. Por ejemplo, `[AGm-p]` concuerda con cualquiera de estos: `A`, `G`, `m`, `n`, `o` o `p`.

La concordancia de patrones distingue entre mayúsculas y minúsculas, lo que puede causar problemas en Windows. Puede forzar que no se distingan a lo bruto, emparejando los caracteres; es decir, por ejemplo, para ignorar `*.tmp` sin tener en cuenta las mayúsculas y minúsculas, puede utilizar un patrón como `*.[Tt][Mm][Pp]`.

Si desea una definición oficial de la expansión de comodines en los nombres de ficheros, puede encontrarla en las especificaciones de la IEEE para el lenguaje de comandos del shell *Notación de Patrones de Concordancia* [http://www.opengroup.org/onlinepubs/009695399/utilities/xcu_chap02.html#tag_02_13].

Sin rutas en la lista global de ignorados

No debería incluir información de ruta en su patrón. La concordancia de patrones está pensanda para usarse contra nombres de fichero planos y nombres de carpeta. Si desea ignorar todas las carpetas `CVS`, simplemente añada `CVS` a la lista de ignorados. No hay necesidad de especificar `CVS */CVS` como se hacía en versiones anteriores. Si desea ignorar todas las carpetas `tmp` que existen dentro de una carpeta `prog` y no dentro de una carpeta `doc` debería utilizar la propiedad `svn:ignore`. No hay una forma fiable de conseguir esto utilizando patrones globales de ignorados.

4.14. Eliminando, moviendo y renombrando

Al contrario que CVS, Subversion le permite renombrar y mover ficheros y carpetas. Por tanto hay entradas de menú para borrar y renombrar en el submenú TortoiseSVN.

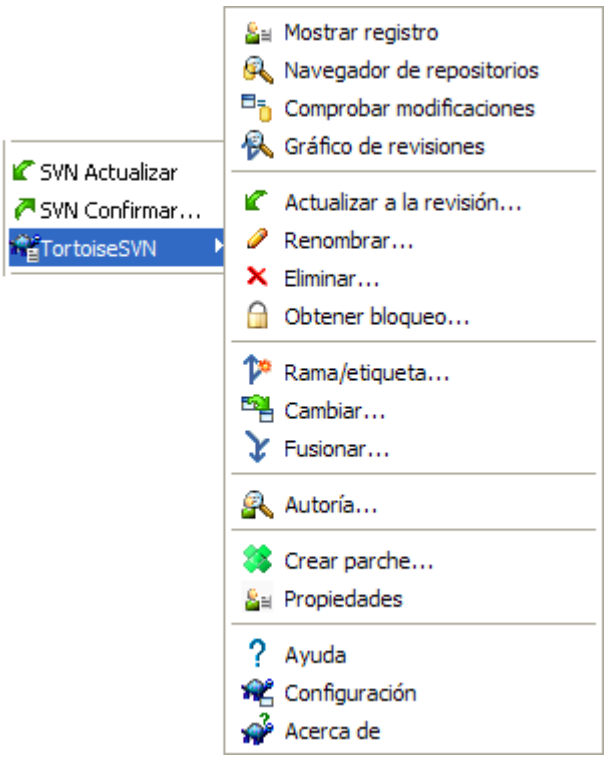


Figura 4.29. Menú contextual del explorador para ficheros versionados

4.14.1. Eliminando ficheros y carpetas

Utilice TortoiseSVN → Eliminar para eliminar ficheros o carpetas de Subversion.

Cuando TortoiseSVN → Elimina un fichero, se elimina de su copia de trabajo inmediatamente, a la vez que se marca para su borrado en el repositorio para la próxima confirmación. La carpeta padre del fichero muestra un icono sobreimpresionado de "eliminado". Hasta que confirme el cambio, puede recuperar el fichero utilizando TortoiseSVN → Revertir sobre la carpeta padre.

Cuando se utiliza TortoiseSVN → Eliminar sobre una carpeta, ésta continúa en su copia de trabajo, pero la sobreimpresión cambia para indicarle que está marcada para ser eliminada. Hasta que confirme el cambio, puede recuperar la carpeta utilizando TortoiseSVN → Revertir sobre la propia carpeta. Esta diferencia en el comportamiento entre ficheros y carpetas es parte de Subversion, no de TortoiseSVN.

Si desea eliminar un ítem del repositorio, pero mantenerlo localmente como un fichero o carpeta no versionados, utilice Menú contextual extendido → Eliminar (mantener local). Debe mantener pulsada la tecla **Mayús** mientras hace click con el botón derecho sobre el ítem en el panel de lista de exploración (panel derecho) para ver esto en el menú contextual extendido.

Si se borra un *fichero* utilizando el Explorador en vez de usar el menú contextual de TortoiseSVN, el diálogo de confirmación le muestra esos ficheros y le deja eliminarlos del control de versiones antes de confirmar. Sin embargo, si actualiza su copia de trabajo, Subversion se dará cuenta de que falta un fichero y lo reemplazará con la última versión del repositorio. Si necesita borrar un fichero bajo control de versiones, utilice siempre TortoiseSVN → Eliminar para que Subversion no tenga que averiguar qué es lo que realmente quiere hacer usted.

Si se borra una *carpeta* utilizando el Explorador en vez de utilizar el menú contextual de TortoiseSVN, su copia de trabajo se romperá y no podrá confirmar. Si actualiza su copia de trabajo, Subversion reemplazará

la carpeta que falta con la última versión del repositorio, y luego podrá eliminarla de la forma correcta, utilizando TortoiseSVN → Eliminar.

Recuperando un fichero o una carpeta borrados

Si ha borrado un fichero o una carpeta y ya ha confirmado esa operación de borrado en el repositorio, entonces un comando TortoiseSVN → Revertir normal no los podrá recuperar. Pero el fichero o la carpeta borrados no están perdidos para siempre. Si sabe la revisión en la que se borraron el fichero o la carpeta (si no lo sabe, utilice el diálogo de registro para averiguarlo), abra el navegador de repositorios y cambie a esa revisión. Luego seleccione el fichero o la carpeta que ha borrado, haga click con el botón derecho y seleccione Menú Contextual → Copiar a..., y como destino de esa operación de copia seleccione la ruta de su copia de trabajo.

4.14.2. Moviendo ficheros y carpetas

Si desea simplemente renombrar en el sitio un fichero o una carpeta, utillice Menú contextual → Renombrar... Introduzca el nuevo nombre para el ítem, y ya está.

Si quiere mover ficheros dentro de una copia de trabajo, quizás a una subcarpeta distinta, utilice el manejador de arrastrar-y-soltar con el botón derecho del ratón:

1. seleccione los ficheros o directorios que desea mover
2. arrástrelos con el botón derecho a su nuevo destino dentro de la copia de trabajo
3. suelte el botón derecho del ratón
4. en el menú contextual seleccione Menú contextual → SVN Mover ficheros versionados aquí

Confirmar la carpeta padre

Dado que las operaciones mover y renombrar se realizan como un borrado seguido de un añadir, debe confirmar la carpeta padre de los ficheros movidos/renombrados para que se muestre la parte de borrado de la operación en el diálogo de confirmación. Si no confirma la parte eliminada de los ficheros movidos/renombrados, se quedarán en el repositorio, y cuando sus compañeros se actualicen no se eliminará el fichero antiguo, es decir, que ellos tendrán *ambas* copias, la antigua y la nueva.

Usted *debe* confirmar un cambio de nombre de carpeta antes de cambiar cualquier fichero dentro de la carpeta; si no, su copia de trabajo realmente puede quedar estropeada.

También puede utilizar el navegador de repositorios para mover ítems. Lea Sección 4.24, “El navegador de repositorios” para saber más.

No SVN Mueva externos

NO debería utilizar los comandos TortoiseSVN Mover o Renombrar en una carpeta que ha sido creada utilizando `svn:externals`. Esta acción puede provocar que el ítem externo se elimine de su repositorio padre, probablemente molestando a muchas otras personas. Si necesita mover una carpeta externa, debería moverla como lo hace con el resto de ficheros sin versionar, y luego ajustar las propiedades `svn:externals` de las carpetas padres origen y destino.

4.14.3. Cambiando las mayúsculas/minúsculas en un nombre de fichero

Hacer cambios sólo en las mayúsculas/minúsculas de un nombre de fichero es complicado en Subversion sobre Windows, porque durante un corto periodo de tiempo durante el renombrado, ambos nombres de fichero deben existir. Dado que Windows tiene un sistema de ficheros que no distingue mayúsculas y minúsculas, esto no funciona utilizando el habitual comando Renombrar.

Afortunadamente hay (al menos) dos soluciones posibles para renombrar un fichero sin perder su historia de registro. Es importante renombrarlo dentro de Subversion. ¡Renombrarlo únicamente en el explorador corromperá su copia de trabajo!

Solución A) (recomendada)

1. Confirme los cambios en su copia de trabajo.
2. Renombrar el fichero de MAYusculas a mayUSCULAS directamente en el repositorio utilizando el Navegador de Repositorios.
3. Actualice su copia de trabajo.

Solución B)

1. Renombre de MAYusculas a MAYusculas_ con el comando Renombrar del submenú de TortoiseSVN.
2. Confirme los cambios.
3. Renombre de MAYusculas_ a mayUSCULAS.
4. Confirme los cambios.

4.14.4. Lidiando con conflictos en las mayúsculas/minúsculas de un nombre de fichero

En el caso de que tenga dos ficheros en el repositorio con el mismo nombre pero diferenciándose únicamente en las mayúsculas (por ejemplo, `TEST.TXT` y `test.txt`), ya no podrá actualizar u obtener el directorio padre en un cliente de Windows. Aunque Subversion soporta los nombres de ficheros que se distinguen en las mayúsculas, Windows no.

Esto ocurre a veces cuando dos personas confirman, desde copias de trabajo separadas, ficheros tales que tienen el mismo nombre pero con las mayúsculas/minúsculas diferentes. Esto también puede ocurrir cuando se confirman los ficheros desde un sistema con un sistema de ficheros que distingue mayúsculas/minúsculas, como Linux.

En ese caso, necesita decidir cuál desea mantener y borrar (o renombrar) el otro en el repositorio.

Evitando dos ficheros con el mismo nombre

Hay un script gancho de servidor disponible en: *http://svn.collab.net/repos/svn/trunk/contrib/hook-scripts/* que evitará confirmaciones cuyo resultado sea dos ficheros con conflicto en las mayúsculas.

4.14.5. Reparando renombrados de ficheros

A veces su amistoso IDE renombrará ficheros por usted como parte de un ejercicio de refactorización, y por supuesto no se lo dirá a Subversion. Si intenta confirmar sus cambios, Subversion verá el nombre de fichero antiguo como faltante y el nuevo nombre como fichero no versionado. Podría simplemente confirmar el nuevo nombre de fichero para añadirlo, pero entonces perdería la traza de la historia, ya que Subversion no sabe la relación entre los ficheros.

Una forma mejor es notificar a Subversion de que el cambio es realmente un renombrado, y puede hacer esto dentro de los diálogos Confirmar y Comprobar modificaciones. Simplemente seleccione tanto el

nombre de fichero antiguo (faltante) como el nuevo nombre (no versionado) y utilice Menú contextual → Reparar movimiento para emparejar los dos ficheros como un renombrado.

4.14.6. Eliminando ficheros no versionados

Normalmente especifica un la lista de ignorados tal que todos los ficheros generados se ignoran en Subversion. Pero ¿y si lo que quiere es limpiar todos esos ítems ignorados para producir una compilación limpia? Normalmente lo haría en su makefile (fichero de instrucciones de compilación), pero si está depurando el makefile, o cambiando el sistema de compilación, es útil tener una forma de barrer la casa.

TortoiseSVN proporciona exactamente esa opción utilizando Menú contextual extendido → Eliminar ítems no versionados.... Debe mantener pulsada la tecla **Mayús** mientras hace click con el botón derecho sobre el ítem en el panel de lista de exploración (panel derecho) para ver esto en el menú contextual extendido. Esto mostrará un diálogo que lista todos los ficheros no versionados en cualquier lugar de su copia de trabajo. Ahí podrá seleccionar o deseleccionar ítems para ser eliminados.

Cuando dichos ítems se eliminan, se utiliza la papelera de reciclaje, por lo que si comete un error aquí y borra un fichero que debería haber estado versionado, aún puede recuperarlo.

4.15. Deshacer cambios

Si desea deshacer todos los cambios que ha hecho en un fichero desde la última actualización, necesita seleccionar el fichero, hacer click con el botón derecho para sacar el menú contextual, y luego seleccionar el comando TortoiseSVN → Revertir Aparecerá un diálogo que le muestra los ficheros que ha cambiado y que puede revertir. Seleccione los que desee revertir y pulse Aceptar.

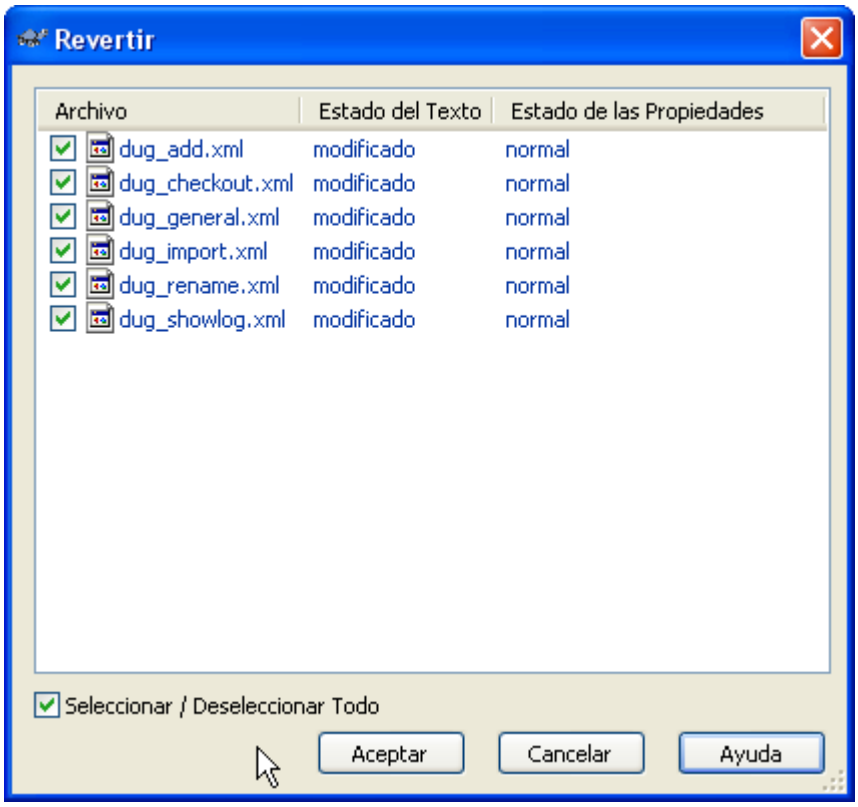


Figura 4.30. Diálogo de Revertir

Si desea deshacer un borrado o un renombrado, necesitará utilizar Revertir en la carpeta padre dado que el ítem borrado ya no existe y no podrá hacer click con el botón derecho sobre él.

Si quiere deshacer la adición de un ítem, esto aparece en el menú contextual como TortoiseSVN → Deshacer añadir.... Esto también es revertir, en realidad, pero su nombre se ha cambiado para hacerlo más obvio.

Las columnas en este diálogo pueden ser personalizadas de la misma forma que las columnas en el diálogo Comprobar modificaciones. Para más detalles, lea Sección 4.7.3, “Estado local y remoto”.

Deshaciendo cambios que han sido confirmados

Revertir sólo deshará sus cambios locales. *No* deshace ningún cambio que ya haya sido confirmado. Si desea deshacer todos los cambios que se confirmaron en una revisión en particular, lea Sección 4.9, “Diálogo de Registro de revisiones” para obtener más información.

Revertir es lento

Cuando revierte los cambios puede que note que la operación lleva bastante más tiempo del que sería de esperar. Esto es así porque la versión modificada del fichero se manda a la papelera de reciclaje, para que pueda recuperar sus cambios si ha revertido por error. Sin embargo, si su papelera de reciclaje está llena, Windows necesita bastante tiempo para encontrar dónde poner el fichero. La solución es simple: o bien vacía la papelera de reciclaje, o bien desactiva la casilla Utilizar la papelera de reciclaje al revertir en la configuración de TortoiseSVN.

4.16. Limpieza

Si un comando de Subversion no puede completarse de forma correcta, quizás por problemas en el servidor, su copia de trabajo puede quedarse en un estado inconsistente. En ese caso deberá utilizar TortoiseSVN → Limpiar en la carpeta. Es una buena idea hacerlo en la rama superior de la copia de trabajo.

El comando Limpiar tiene otro efecto secundario muy útil. Si ha cambiado la fecha de un fichero pero no su contenido, Subversion no puede determinar si el fichero ha cambiado realmente excepto haciendo una comparación byte-a-byte con la copia prístina. Si tiene muchos ficheros en este estado, comprobar su estado será muy lento, lo que hará que muchos diálogos ralenticen su respuesta. Ejecutando el comando Limpiar en su copia de trabajo reparará esas fechas “rotas” y restaurará las comprobaciones de estado a la máxima velocidad.

Utilizar fechas de confirmación

Algunas versiones anteriores de Subversion estaban afectadas con un error que causaba errores en la fecha cuando obtenía con la opción Utilizar fechas de confirmación activada. Utilice el comando Limpieza para acelerar esas copias de trabajo.

4.17. Configuración del proyecto

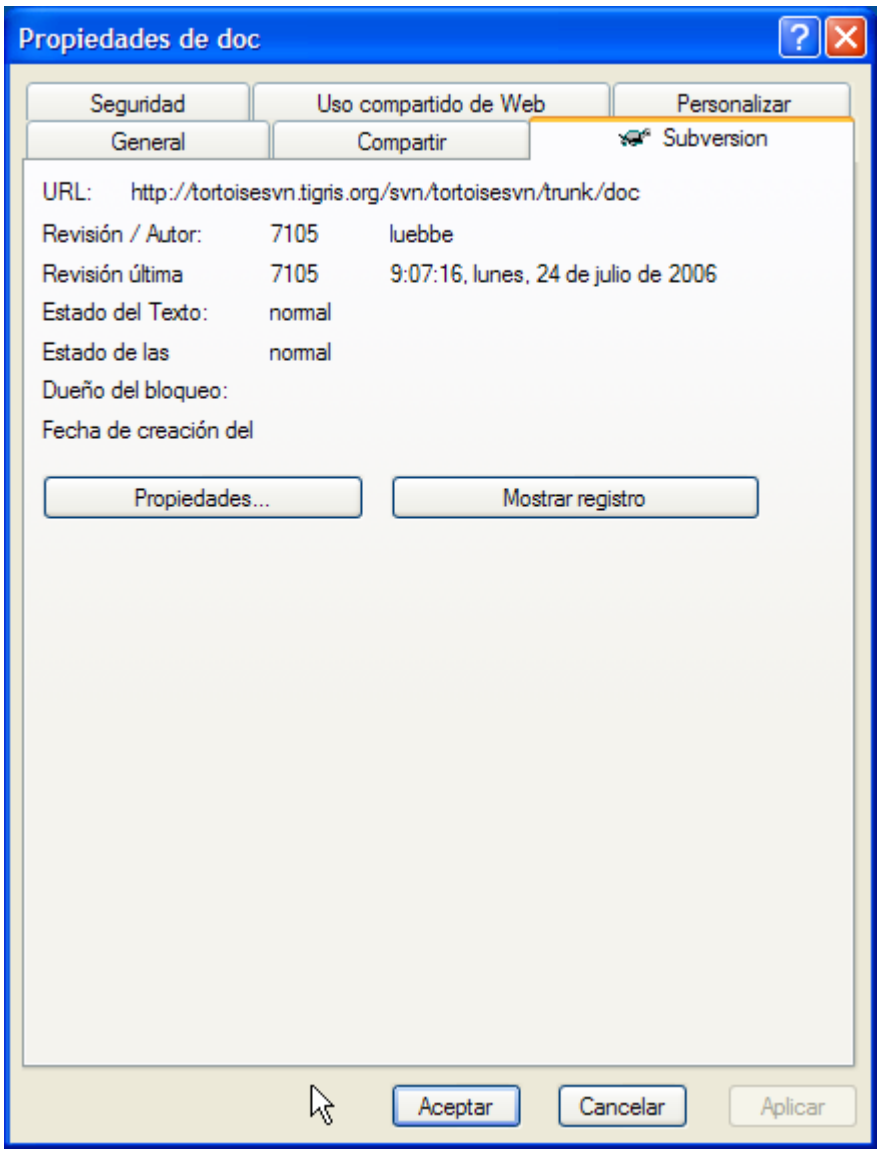


Figura 4.31. Página de propiedades del Explorador, pestaña Subversion

A veces desea obtener información más detallada sobre un fichero o directorio que la que proporcionan los iconos sobreimpresionados. Puede obtener toda la información que Subversion proporciona en el diálogo de propiedades del explorador. Simplemente seleccione el fichero o directorio y seleccione Menú de Windows → Propiedades en el menú contextual (atención: ésta es la entrada normal del menú de propiedades que el explorador proporciona, ¡no la que está en el submenú de TortoiseSVN!). En el cuadro de diálogo de propiedades, TortoiseSVN ha añadido una nueva página de propiedades para los ficheros y carpetas bajo control de Subversion, donde puede ver toda la información relevante sobre el fichero/directorio seleccionado.

4.17.1. Propiedades de Subversion

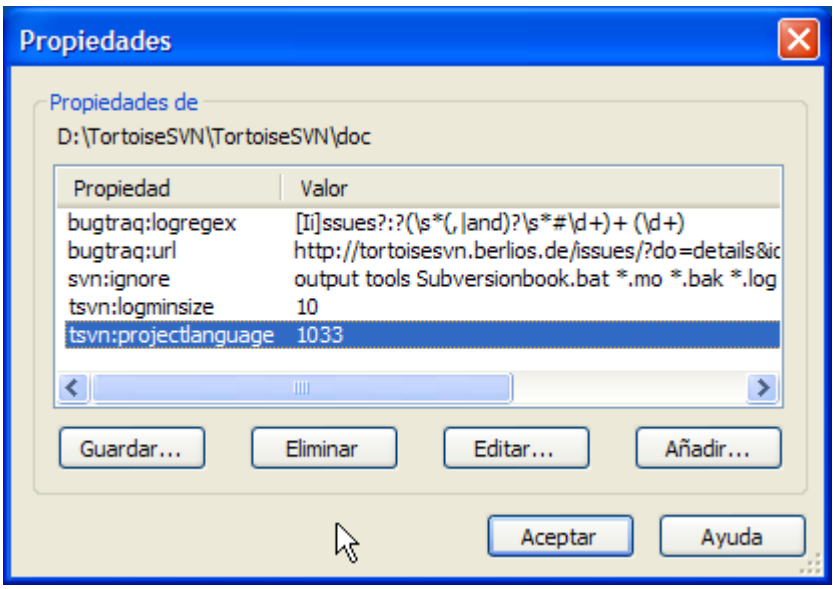


Figura 4.32. Página de propiedades de Subversion

Puede leer y establecer las propiedades de Subversion desde el diálogo de propiedades de Windows, pero también desde TortoiseSVN → Propiedades y dentro de las listas de estado de TortoiseSVN, desde Menú contextual → Propiedades.

Puede añadir sus propias propiedades, o algunas propiedades con un significado especial en Subversion. Éstas empiezan con `svn:`. `svn:externals` es una de esas propiedades; vea cómo manejar externos en Sección 4.18, “Ítems externos”.

4.17.1.1. svn:keywords

Subversion soporta expansión de palabras clave similar a CVS, lo que puede utilizarse para introducir el nombre del fichero y la información de la revisión dentro del propio fichero. Las palabras clave actualmente soportadas son:

$Date$
: Fecha de la última confirmación conocida. Esto se basa en la información obtenida cuando actualiza su copia de trabajo. *No* se contacta con el repositorio para buscar cambios más recientes.

$Revision$
: Revisión de la última confirmación conocida.

$Author$
: Autor que hizo la última confirmación conocida.

$HeadURL$
: La URL completa de este fichero en el repositorio.

Id
: Una combinación comprimida de las cuatro palabras clave anteriores.

Para averiguar cómo utilizar estas palabras clave, lea la *sección svn:keywords* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.special.keywords.htm] del Libro de Subversion, que porporciona información completa sobre estas palabras clave y cómo habilitarlas y utilizarlas.

Puede encontrar más información sobre las propiedades en Subversion en *Propiedades especiales* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.html].

4.17.1.2. Añadiendo y editando propiedades

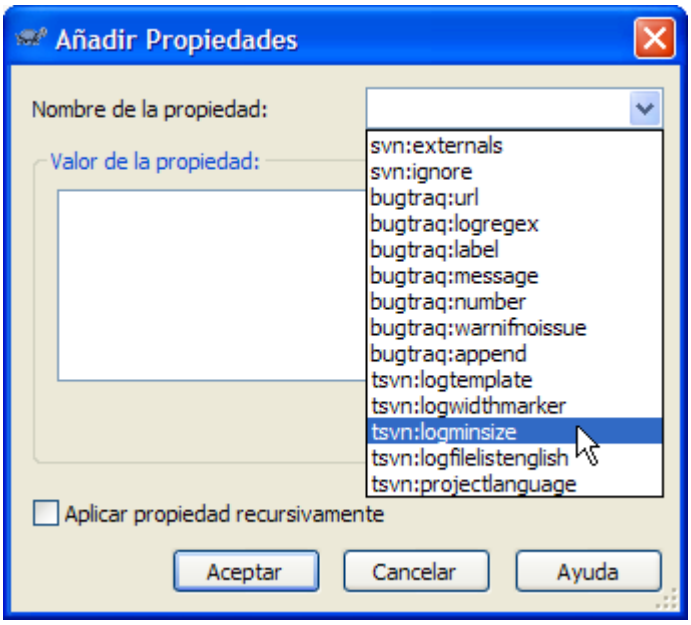


Figura 4.33. Añadiendo propiedades

Para añadir una nueva propiedad, primero pulse en Añadir.... Seleccione el nombre de la propiedad deseado del cuadro desplegable, o escriba un nombre de su elección, y luego introduzca un valor en la caja de texto que hay debajo. Para las propiedades que admiten múltiples valores, como una lista de ignorados, éstos se pueden introducir en múltiples líneas. Pulse en Aceptar para a

Si desea aplicar una propiedad a muchos ítems a la vez, seleccione los ficheros/carpetas en el explorador, y luego seleccione Menú contextual → propiedades

Si desea aplicar la propiedad a *cada* fichero y carpeta en la jerarquía debajo de la carpeta actual, seleccione la casilla Recursivo.

Algunas propiedades, por ejemplo `svn:needs-lock`, pueden aplicarse únicamente a ficheros, por lo que ese nombre de propiedad no aparece en la lista desplegable para las carpetas. Aún así puede aplicar dicha propiedad recursivamente a todos los ficheros en una jerarquía, pero tendrá que escribir el nombre de la propiedad usted mismo.

Si desea editar una propiedad existente, primero seleccione dicha propiedad de la lista de propiedades existentes, y luego pulse en Editar....

Si desea eliminar una propiedad existente, seleccione dicha propiedad de la lista de propiedades existentes, y luego pulse en Eliminar.

La propiedad `svn:externals` se puede utilizar para traer otros proyectos del mismo repositorio o de otro completamente distinto. Para más información, lea Sección 4.18, “Ítems externos”.

4.17.1.3. Exportando e importando propiedades

A menudo se encontrará aplicando el mismo conjunto de propiedades muchas veces, por ejemplo `bugtraq:logregex`. Para simplificar el proceso de copiar propiedades de un proyecto a otro, puede utilizar la característica Exportar/Importar.

Desde el fichero o carpeta en el que las propiedades ya están establecidas, utilice TortoiseSVN → Propiedades, seleccione las propiedades que desea exportar y haga click en Exportar.... Se le preguntará por un nombre de fichero donde se grabarán los nombres de las propiedades y sus valores.

Desde la(s) carpeta(s) donde desea aplicar estas propiedades, utilice TortoiseSVN → Propiedades y haga click en Importar.... Se le preguntará por un nombre de fichero desde el que importar, por lo que deberá navegar al lugar donde almacenó el fichero exportado anteriormente y seleccionarlo. Las propiedades se añadirán a las carpetas de forma no recursiva.

Si desea añadir las carpetas a un árbol recursivamente, siga los pasos anteriores, y luego en el diálogo de propiedades seleccione cada propiedad por turnos, haga click en Editar..., seleccione la casilla Aplicar propiedad recursivamentey haga click en Aceptar.

El formato del fichero de importación es binario y propietario de TortoiseSVN. Su único propósito es transferir las propiedades utilizando Importar y Exportar, por lo que no hay necesidad de editar estos ficheros.

4.17.1.4. Propiedades binarias

TortoiseSVN puede manejar valores de propiedades binarios utilizando ficheros. Para leer un valor de propiedad binario, utilice Grabar... para almacenarlo en un fichero. Para establecer un valor binario, utilice un editor hexadecimal u otra herramienta adecuada para crear un fichero con el contenido que necesite, y luego utilice Cargar... para leer ese fichero.

Aunque las propiedades binarias no se utilizan a menudo, pueden ser útiles en algunas aplicaciones. Por ejemplo, si está almacenando ficheros gráficos muy grandes, o si la aplicación que se utiliza para cargar el fichero es pesada, puede querer guardar una pequeña imagen guía como una propiedad para poder obtener una previsualización rápidamente.

4.17.1.5. Establecer propiedades automáticamente

Puede configurar Subversion y TortoiseSVN para que establezca propiedades automáticamente en ficheros y carpetas cuando se añaden al repositorio. Hay dos formas de hacerlo.

Puede editar el fichero de configuración de Subversion para habilitar esta característica en su cliente. La página General del diálogo de configuración de TortoiseSVN tiene un botón de edición para llevarle allí directamente. El fichero de configuración es un fichero de texto simple que controla algunas de las funciones de Subversion. Necesita cambiar dos cosas: primero, en la sección encabezada por `miscellany`, descomente la línea `enable-auto-props = yes`. En segundo lugar, necesita editar la sección a continuación para definir qué propiedades quiere que se añadan a qué tipos de ficheros. Este método es una función estándar de Subversion y funciona con cualquier cliente de Subversion. Sin embargo, debe definirse en cada cliente de forma individual - no hay forma de propagar estas configuraciones desde el repositorio.

Un método alternativo es establecer la propiedad `tsvn:autoprops` en las carpetas, como se describe en la siguiente sección. Este método funciona sólo con los clientes TortoiseSVN, pero se propaga a todas las copias de trabajo al actualizarlas.

Sea cual sea el método que elija, debe tener en cuenta que las auto-props se aplican sólo a los ficheros en el momento en que se añaden al repositorio. Las auto-props nunca cambiarán las propiedades de ficheros que ya estén versionados.

Si desea estar totalmente seguro de que a los nuevos ficheros se les aplica las propiedades correctas, debería establecer un script gancho pre-confirmación para rechazar las confirmaciones cuando estas propiedades necesarias no estén establecidas.

Confirmar propiedades

Las propiedades de Subversion están versionadas. Después de que cambie o añada una propiedad tiene que confirmar sus cambios.

Conflictos en las propiedades

Si hay un conflicto al confirmar los cambios, porque otro usuario ha cambiado la misma propiedad, Subversion genera un fichero `.prej`. Borre este fichero después de que haya resuelto del conflicto.

4.17.2. Propiedades de projecto TortoiseSVN

TortoiseSVN tiene unas pocas propiedades especiales para sí mismo, y estas empiezan con `tsvn:`.

- `tsvn:logminsize` establece la longitud mínima de un mensaje de registro para una confirmación. Si introduce un mensaje más corto de lo especificado aquí, la confirmación se deshabilita. Esta característica es muy útil para que se acuerde de proporcionar un mensaje descriptivo apropiado para cada confirmación. Si esta propiedad no se establece, o el valor es cero, se permiten mensajes de registro vacíos.

  `tsvn:lockmsgminsize` establece la longitud mínima para un mensaje de bloqueo. Si introduce un mensaje más corto de lo especificado aquí, el bloqueo se deshabilita. Esta característica es muy útil para que se acuerde de proporcionar un mensaje descriptivo apropiado para cada blqoueo que obtenga. Si esta propiedad no se establece, o el valor es cero, se permiten mensajes de bloqueo vacíos.

- `tsvn:logwidthmarker` se utiliza con proyectos que necesitan que los mensajes de registro se formateen con algún ancho máximo (típicamente, 80 caracteres) antes de un salto de línea. Al establecer esta propiedad a un valor distinto de cero ocurren dos cosas en el diálogo de entrada de mensajes de registro: pone un marcador para indicar el ancho máximo, y deshabilita el ajuste de línea en la pantalla, para que pueda ver si el texto que ha introducido es muy largo. Tenga en cuenta que esta característica sólo funcionará correctamente si ha seleccionado una fuente de ancho fijo para los mensajes de registro.

- `tsvn:logtemplate` se utiliza en proyectos que tienen reglas sobre el formateo de mensajes. Esta propiedad alberga una cadena de texto de múltiples líneas que se insertará en el cuadro del mensaje de confirmación cuando inice una confirmación. Puede editarlo para incluir la información necesaria. Tenga en cuenta que si estaba utilizando también `tsvn:logminsize`, debe asegurarse de establecer una longitud mayor que la de la plantilla o perderá el mecanismo de protección.

- Subversion le permite establecer “autoprops” (autopropiedades), que serán aplicadas a los ficheros recién añadidos o importados, basándose en la extensión del fichero. Esto necesita que todos los clientes hayan establecido las autoprops adecuadas en el fichero de configuración de Subversion. `tsvn:autoprops` puede establecerse en carpetas y éstas se fusionarán con las autoprops locales del usuario cuando se importen o añadan ficheros. El formato es el mismo que para las autopropiedades de Subversion, por ejemplo `*.sh = svn:eol-style=native;svn:executable` establece dos propiedades en los ficheros con extensión `.sh`.

  Si hay un conflicto entre las autoprops locales y `tsvn:autoprops`, la configuración del proyecto tiene preferencia ya que son específicas para ese proyecto.

- En el diálogo de Confirmar tiene la opción de pegar la lista de ficheros cambiados, incluyendo el estado de cada fichero (añadido, modificado, etc). `tsvn:logfilelistenglish` define si el estado del fichero se inserta en inglés o en el idioma traducido. Si la propiedad no se establece, el valor por defecto es `true`.

- TortoiseSVN puede utilizar los módulos de corrector ortográfico que se utilizan también por OpenOffice y Mozilla. Si tiene éstos instalados, esta propiedad determinará qué corrector ortográfico utilizar, es decir, en qué lenguaje deberían estar escritos los mensajes de registro para su proyecto. `tsvn:projectlanguage` establece el módulo de idioma que el motor del corrector ortográfico debería utilizar cuando introduzca un mensaje de registro. Puede encontrar los valores para su idioma en esta página: *MSDN: Language Identifiers* [http://msdn2.microsoft.com/en-us/library/ms776260.aspx].

Puede introducir este valor en decimal, o en hexadecimal si se precede por `0x`. Por ejemplo, Español (España, ordenación tradicional) puede introducirse como `0x040a` or `1034`.

- La propiedad `tsvn:logsummary` se utiliza para extraer una porción del mensaje de registro que luego se muestra en el diálogo de registro como el sumario del mensaje de registro.

  El valor de la propiedad `tsvn:logsummary` debe establecerse a una cadena de texto regex de una línea que contenga un grupo regex. Lo que satisfaga ese grupo se utilizará como sumario.

  Por ejemplo: `\[SUMARIO\]:\s+(.*)` capturará todo lo que haya tras "[SUMARIO]" en el mensaje de registro y lo utillizará como sumario.

- Cuando quiera añadir una nueva propiedad, puede o bien elegir una desde la lista en el cuadro desplegable, o puede introducir el nombre de propiedad que desee. Si su proyecto utiliza algunas propiedades personalizadas, y quiere que esas propiedades aparezcan en la lista del cuadro desplegable (para evitar errores ortográficos cuando introduzca un nombre de propiedad), puede crear una llista de sus propias propiedades personalizadas utilizando `tsvn:userfileproperties` y `tsvn:userdirproperties`. Aplique estas propiedades a una carpeta. Cuando vaya a editar las propiedades de cualquier ítem hijo, sus propiedades personalizadas aparecerán en la lista de nombres de propiedades predefinidas.

Algunas propiedades `tsvn:` necesitan un valor `true/false` (verdadero/falso). TortoiseSVN también entiende `yes` (sí) como un sinónimo de `true` y `no` como un sinónimo de `false`.

TortoiseSVN puede integrarse con algunas herramientas de control de errores. Esto utiliza propiedades de proyecto que comienzan con `bugtraq:`. Para más información, lea Sección 4.28, "Integración con sistemas de control de errores / seguimiento de incidencias".

También puede integrarse con algunos visores de repositorios basados en web, utilizando propiedades de proyecto que comienzan con `webviewer:`. Para más información, lea Sección 4.29, "Integración con visores de repositorios basados en web".

Establecer las propiedades de proyecto en carpetas

Estas propiedades especiales del proyecto deben estar establecidas en *carpetas* para que el sistema funcione. Cuando confirma un fichero o una carpeta, se leen las propiedades de esa carpeta. Si no se encuentran allí las propiedades, TortoiseSVN las buscará hacia arriba en el árbol de carpetas para encontrarlas, hasta que llega a una carpeta sin versionar, o se encuentra la raíz del árbol (por ejemplo, `C:\`). Si puede estar seguro de que cada usuario obtiene sólo desde por ejemplo `trunk/` y no desde alguna subcarpeta, entonces es suficiente establecer las propiedades en `trunk/`. Si no puede estar seguro, debería establecer las propiedades recursivamente en cada subcarpeta. Una propiedad establecida en una carpeta más profunda dentro de la jerarquía del proyecto tiene preferencia sobre las propiedades establecidas en niveles más altos (más cerca de `trunk/`).

Para las propiedades de proyecto *sólo* puede utilizar la casilla Recursivo para establecer la propiedad en todas las subcarpetas de la jerarquía, sin establecerla en todos los ficheros.

Cuando añada nuevas subcarpetas utilizando TortoiseSVN, cualquier propiedad de proyecto presente en la carpeta padre se añadirá también automáticamente a la nueva carpeta hija.

Atención

Aunque las propiedades de proyecto de TortoiseSVN son extremadamente útiles, sólo funcionan con TortoiseSVN, y algunas sólo funcionarán en las versiones más recientes de TortoiseSVN. Si la gente que trabaja en su proyecto utiliza un número de clientes

de Subversion diferentes, o es posible que tengan versiones antiguas de TortoiseSVN, puede que quiera utilizar ganchos de repositorio para forzar políticas de proyectos. Las propiedades de proyecto sólo ayudan a implementar una política, no pueden forzarlas.

4.18. Ítems externos

A veces es útil construir una copia de trabajo que esté hecha de un número de diferentes obtenciones. Por ejemplo, puede querer que diferentes ficheros o carpetas provengan de orígenes distintos de un repositorio, o quizás de repositorios diferentes. Si desea que todos los usuarios tengan la misma estructura, puede definir las propiedades `svn:externals` para traer el recurso especificado al lugar donde se le necesita.

4.18.1. Carpetas externas

Let's say you check out a working copy of `/project1` to `D:\dev\project1`. Select the folder `D:\dev\project1`, right click and choose Windows Menu → Properties from the context menu. The Properties Dialog comes up. Then go to the Subversion tab. There, you can set properties. Click Add.... Select the `svn:externals` property from the combobox and write in the edit box the repository URL in the format `url folder` or if you want to specify a particular revision, `-rREV url folder` You can add multiple external projects, 1 per line. Suppose that you have set these properties on `D:\dev\project1`:

```
http://sounds.red-bean.com/repos  sounds
http://graphics.red-bean.com/repos/fast%20graphics  "quick graphs"
-r21 http://svn.red-bean.com/repos/skin-maker  skins/toolkit
```

Now click Set and commit your changes. When you (or any other user) update your working copy, Subversion will create a sub-folder `D:\dev\project1\sounds` and checkout the sounds project, another sub-folder `D:\dev\project1\quick_graphs` containing the graphics project, and finally a nested sub-folder `D:\dev\project1\skins\toolkit` containing revision 21 of the skin-maker project.

Los URLs deberán de estar adecuadamente escapados o de lo contrario, no funcionarán, por ejemplo: usted debe de reemplazar cada espacio con `%20`, tal y como se muestra en el segundo ejemplo anterior.

Si usted desea que la ruta local incluya espacios u cualquier otro caracter especial, puede encerrarlo con comillas dobles, o bién, utilizar el `\` símbolo de barras hacia atrás como un caracter de escape al estilo del shell de Unix, precediendo a cualquier caracter especial. Por supuesto que esto significa también que deberá de usar el símbolo `/` (barra hacia delante) como delimitador de ruta. Este comportamiento es nuevo en Subversión 1.6 y no funcionará en clientes anteriores.

Utilice números globales de revisión

Debería considerar seriamente utilizar números de revisión explícitos en todas sus definiciones de externos, como se describe anteriormente. Hacerlo significa que tiene que decidir cuándo utilizar una instantánea diferente de información externa, y exáctamente qué instantánea utilizar. Además del aspecto de sentido común de no ser sorprendido por cambios en repositorios de terceras partes sobre los que puede que no tenga control, utilizar números de revisión explícitos también significa que cuando retroceda su copia de trabajo a una revisión anterior, sus definiciones externas también se revertiran a la forma a la que tenían en esa revisión, lo que a su vez significa que las copias de trabajo externas se actualizarán para concordar en la forma en la que *ellas* tenían cuando su repositorio estaba en esa revisión anterior. Para proyectos de software, esto puede ser la diferencia entre una compilación válida o fallida de una vieja intentánea de su complejo código.

SVN anterioes: definiciones externas

El formato mostrado aquí se introdujo en Subversion 1.5 Puede que vea el formato antiguo que tiene la misma información en distinto orden. El nuevo formato es mejor ya que soporta algunas características útiles que se describen a continuación, pero no funcionará con clientes antiguos. Las diferencias se muestran en el *Libro de Subversion* [http://svnbook.red-bean.com/en/1.5/svn.advanced.externals.html].

Si los proyectos externos están en el mismo repositorio, cualquier cambio que haga allí se incluirá en la lista de confirmación cuando confirme su proyecto principal.

Si los proyectos externos están en repositorios diferentes, cualquier cambio que haga en el proyecto externo se notificará cuando cnofirme el proyecto principal, pero tendrá que confirmar esos cambios externos de forma separada.

Si utiliza URLs absolutas en las definiciones `svn:externals` y tiene que relocalizar su copia de trabajo (por ejemplo si la URL de su repositorio cambia), entonces sus externos no cambiarán y puede que no funcionen más.

Para evitar dichos problemas, los clientes de Subversion versión 1.5 y superior soportan URLs externas relativas. Hay cuatro formas para especificar una URL relativa. En los siguientes ejemplos, asumimos que tenemos dos repositorios: uno en `http://example.com/svn/repos-1` y otro en `http://example.com/svn/repos-2`. Tenemos una copia de trabajo de `http://example.com/svn/repos-1/project/trunk` en `C:\Working` y la propiedad `svn:externals` está establecida en trunk.

Relativa a la carpeta padre
: Estas URLs siempre comienzan con la cadena `../`. Por ejemplo:

```
../../widgets/foo  common/foo-widget
```

Esto extraerá `http://example.com/svn/repos-1/widgets/foo` en `C:\Working\common\foo-widget`.

Tenga en cuenta que la URL es relativa a la URL del directorio con la propiedad `svn:externals`, no al directorio donde la carpeta externa se escribe en el disco.

Relativo a la raíz del repositorio
: Estas URLs siempre comienzan con la cadena `^/`. Por ejemplo:

```
^/widgets/foo  common/foo-widget
```

Esto extraerá `http://example.com/svn/repos-1/widgets/foo` en `C:\Working\common\foo-widget`.

Puede referirse fácilmente a otros repositorios con el mismo `SVNParentPath` (un directorio común que contiene varios repositorios). Por ejempo:

```
^/../repos-2/hammers/claw  common/claw-hammer
```

Esto extraerá `http://example.com/svn/repos-2/hammers/claw` en `C:\Working\common\claw-hammer`.

Relativa al esquema
: Las URLs que comienzan con la cadena `//` copian sólo la parte del esquema de la URL. Esto es útil cuando se debe acceder al mismo nombre de servidor con diferentes esquemas dependiendo de

la red; por ejemplo los clientes en la intranet utilizan `http://` mientras que los clientes externos utilizan `svn+ssh://`. Por ejemplo:

```
//example.com/svn/repos-1/widgets/foo  common/foo-widget
```

Esto extraerá `http://example.com/svn/repos-1/widgets/foo` o `svn+ssh://example.com/svn/repos-1/widgets/foo` dependiendo de qué método se utilice para obtener `C:\Working`.

Relativa al nombre del servidor
: Las URLs que comienzan con la cadena `/` copian el esquema y el nombre del servidor de la URL, por ejemplo:

```
/svn/repos-1/widgets/foo  common/foo-widget
```

Esto extraerá `http://example.com/svn/repos-1/widgets/foo` en `C:\Working\common\foo-widget`. Pero si obtiene su copia de trabajo desde otro servidor en `svn+ssh://another.mirror.net/svn/repos-1/project1/trunk` entonces la referencia externa extraerá `svn+ssh://another.mirror.net/svn/repos-1/widgets/foo`.

Si es necesario, también puede especificar una revisión concreta tras la URL, por ejemplo `http://sounds.red-bean.com/repos@19`.

Si necesita más información sobre cómo TortoiseSVN maneja las Propiedades, lea Sección 4.17, “Configuración del proyecto”.

Para averiguar los diferentes métodos para acceder a subproyectos comunes lea Sección B.6, “Incluir un sub-proyecto común”.

4.18.2. Ficheros externos

Desde Subversion 1.6 puede añadir un fichero suelo externo a su copia de trabajo utilizando la misma sintaxis que se usa para carpetas. Sin embargo, hay algunas restricciones.

- La ruta al fichero externo deberá posicionar a dicho fichero en una carpeta versionada existente. En general, lo más sensato es posicionar el fichero directamente dentro de la carpeta que tiene `svn:externos` activado, pero también puede estar localizado en una subcarpeta versionada, si lo considera necesario. Contrastando con lo anterior, los directorios externos crearán de manera automática carpetas intermedias no versionadas según se vaya requiriendo.
- La URL de un fichero externo debe estar en el mismo repositorio que la URL donde se insertará el fichero externo; los ficheros externos inter-repositorio no están soportados.

Un fichero externo se comporta como cualquier otro fichero versionado en muchos aspectos, pero no pueden ser movidos o eliminados utilizando los comandos normales; en vez de eso, debe modificarse la propiedad `svn:externals`.

Soporte de ficheros externos incompleto en Subversion 1.6

En Subversion 1.6 no es posible eliminar un fichero externo de una copia de trabajo una vez que lo ha añadido, incluso si elimina la propiedad `svn:externals`. Tendrá que obtener una copia de trabajo fresca para eliminar el fichero.

4.19. Haciendo ramas / etiquetas

Una de las características de los sistemas de control de versiones es la posibilidad de aislar cambios en una línea separada de desarrollo. Esto se conoce como una *rama*. Las ramas se utilizan a menudo para

probar nuevas características sin molestar la línea principal del desarrollo con errores de compilación y errores. Tan pronto como la nueva característica es lo suficiente estable, la rama de desarrollo se *fusiona* de nuevo en la rama principal (trunk, troncal/tronco).

Otra característica de los sistemas de control de versiones es la posibilidad de marcar revisiones particulares (por ejemplo, una versión lanzada a producción), para que pueda en cualquier momento recrear un cierto entorno o compilación. Este proceso se conoce como *etiquetar*.

Subversion no tiene comandos especiales para hacer ramas o etiquetas, pero en cambio utiliza lo que se denomina “copias baratas”. Las copias baratas son similares a los vínculos duros de Unix, que significa que en vez de hacer una copia completa en el repositorio, se crea un vínculo interno, apuntando a una revisión y árbol específicos. Como resultado, las ramas y las etiquetas son muy rápidas de crear, y casi no conllevan espacio extra en el repositorio.

4.19.1. Crando una rama o etiqueta

Si ha importado su proyecto con la estructura de directorios recomendados, crear una rama o una etiqueta es muy simple:

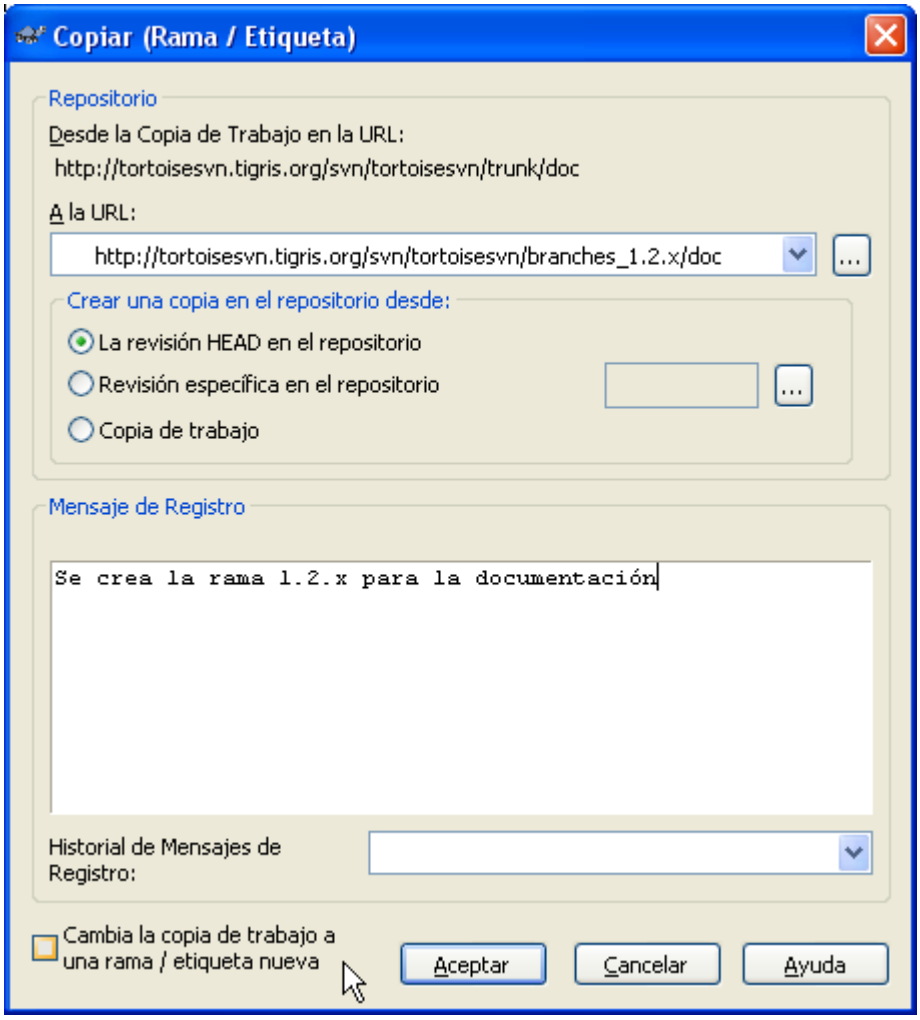


Figura 4.34. El diálogo Rama/Etiqueta

Seleccione la carpeta en su copia de trabajo de la que desea hacer una rama o una etiqueta, y luego seleccione el comando TortoiseSVN → Rama/Etiqueta....

La URL de destino por defecto para la nueva rama será la URL de origen en la que se basa su copia de trabajo. Necesitará editar esa URL con la nueva ruta para su rama/etiqueta. Así que en vez de

```
http://svn.collab.net/repos/NombreDelProyecto/trunk
```

querrá ahora usar algo como

```
http://svn.collab.net/repos/NombreDelProyecto/tags/Version_1.10
```

Si no puede recordar la convención de nombres que usó la última vez, pulse el botón a la derecha para abrir el navegador de repositorios, para que pueda ver la estructura actual del repositorio.

Ahora debe elegir el origen de la copia. Aquí tiene tres opciones:

Revisión HEAD en el repositorio
: La nueva rama se copia directamente en el repositorio desde la revisión HEAD. No se necesita transferir datos desde su copia de trabajo, y la rama se crea muy rápidamente.

Revisión específica en el repositorio
: La nueva rama se copia directamente en el repositorio, pero puede elegir una versión anterior. Esto es útil si se olvidó de crear una etiqueta cuando lanzó una versión de su proyecto la semana pasada. Si no puede acordarse del número de revisión, pulse el botón a la derecha para mostrar el registro de revisiones, y seleccione el número de revisión desde allí. De nuevo no se transfiere datos desde su copia de trabajo, y la rama se crea muy rápidamente.

Copia de trabajo
: La nueva rama es una copia idéntica de su copia de trabajo local. Si ha cambiado algunos ficheros a una revisión anterior en su copia de trabajo, o si ha hecho cambios locales, esto es exactamente lo que irá a la copia. Naturalmente, esta clase de etiquetado complejo conlleva transferir datos desde su copia de trabajo al repositorio si no existe ya allí.

Si desea que su copia de trabajo se cambie automáticamente a la rama recién creada, utilice la casilla Cambiar la copia de trabajo a la nueva rama/etiqueta. Pero si lo hace, asegúrese primero de que su copia de trabajo no contenga modificaciones. Si las tiene, esos cambios se mezclarán en la copia de trabajo de la rama cuando se haga el cambio.

Pulse Aceptar para confirmar la nueva copia al repositorio. No se olvide de proporcionar un mensaje de registro. Tenga en cuenta que la copia se crea *dentro del repositorio*.

Tenga en cuenta que, salvo que haya optado por cambiar su copia de trabajo a la rama recién creada, crear una etiqueta o una rama *no* afecta a su copia de trabajo. Incluso si creó la rama desde su copia de trabajo, estos cambios se confirmarán en la rama nueva, no en el tronco, así que su copia de trabajo todavía se marcará como modificada respecto al tronco.

4.19.2. Obtener o cambiar...

...esa (realmente no) es la cuestión. Mientras que obtener descarga todo de la rama elegida a su directorio de trabajo, TortoiseSVN → Cambiar... sólo transfiere los datos cambiados a su copia de trabajo. Bueno para la carga de la red, bueno para su paciencia. :-)

Para poder trabajar con su rama o etiqueta recién generada tiene varias opciones. Puede:

- TortoiseSVN → Obtener para obtener una copia nueva en una carpeta vacía. Puede obtener la copia de trabajo en cualquier parte de su disco duro, y puede crear tantas copias de trabajo de su repositorio como desee.
- Cambiar su copia de trabajo actual a la copia recién creada en el repositorio. De nuevo seleccione la carpeta superior de su proyecto y utilice TortoiseSVN → Cambiar... del menú contextual.

  En el siguiente diálogo, introduzca la URL de la rama que acaba de crear. Deje la opción Revisión HEAD activada y pulse Aceptar. Su copia de trabajo se cambia a la nueva rama/etiqueta.

Cambiar trabaja igual que Actualizar, en el sentido de que nunca pierde sus cambios locales. Cualquier cambio que haya hecho a su copia de trabajo que todavía no se hayan confirmado se fusionarán cuando haga el Cambio. Si no desea que esto ocurra, entonces debe o bien confirmar los cambios antes de cambiar, o revertir su copia de trabajo a una revisión ya-confirmada (típicamente HEAD).

- Si desea trabajar en el tronco (trunk) y en la rama, pero no desea el coste de una obtención nueva, puede utilizar el Explorador de Windows para hacer una copia de la copia de trabajo obtenida, y luego TortoiseSVN → Cambiar... esa copia a su nueva rama.

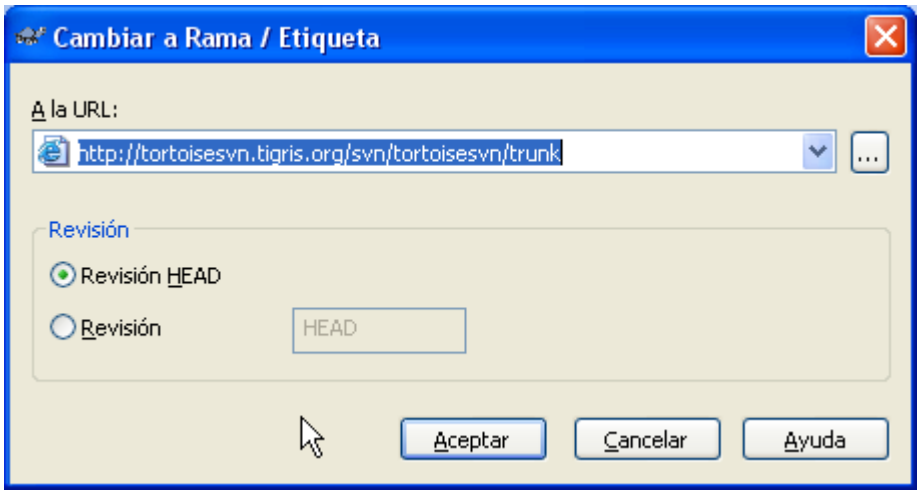


Figura 4.35. El diálogo Cambiar

Aunque Subversion por sí mismo no hace ninguna distinción entre etiquetas y ramas, la forma en la que normalmente se usan difiere un poco.

- Las etiquetas se usan típicamente para crear una copia estática de un proyecto en una etapa concreta. Como tales normalmente no se utilizan para el desarrollo - eso es para lo que se utilizan las ramas, y por esa razón recomendamos la estructura del repositorio `/trunk /branches /tags` en primer lugar. Trabajando en una revisión etiquetada *no es una buena idea*, pero dado que sus ficheros locales no están protegidos, no hay nada que le impida hacer esto por error. Sin embargo, si intenta confirmar a una ruta en el repositorio que contenga `/tags/` (en inglés), TortoiseSVN le avisará.

- Puede ser que necesite hacer más cambios a una versión que ya había etiquetado. La forma correcta de manejar esta situación es crear primero una nueva rama desde la etiqueta. Haga sus cambios en esta rama, y luego cree una nueva etiqueta para esta rama, por ejemplo `Version_1.0.1`.

- Si modifica una copia de trabajo creada desde una rama y confirma, entonces los cambios irán a la nueva rama y *no* en el tronco. Sólo se guardan las modificaciones. El resto continúa siendo una copia barata.

4.20. Fusionando

Mientras que las ramas se utilizan para mantener líneas de desarrollo separadas, en alguna etapa tendrá que fusionar los cambios hechos en una rama de vuelta en el tronco, o viceversa.

Es importante entender cómo funcionan las ramas y la fusión en Subversion antes de empezar a utilizarlos, porque puede convertirse en algo bastante complejo. Le recomendamos encarecidamente que lea el capítulo *Creando ramas y fusionando* [http://svnbook.red-bean.com/en/1.5/svn.branchmerge.html] en el Libro de Subversion, que le proporcionará una descripción completa, y muchos ejemplos de cómo se utiliza.

El siguiente punto a tener en cuenta es que la fusión *siempre* tiene lugar en una copia de trabajo. Si quiere fusionar cambios *en* una rama, deberá tener una copia de trabajo obtenida de dicha rama, e invocar el asistente de fusionado desde esa copia de trabajo utilizando TortoiseSVN → Fusionar....

En general es una buena idea realizar fusiones en una copia de trabajo sin modificar. Si ha hecho otros cambios en su copia de trabajo, confírmelos primero. Si la fusión no funciona como espera, puede querer revertir los cambios, y el comando Revertir descartará *todos* los cambios, incluidos cualquiera que haya hecho antes de la fusión.

Hay tres casos de uso comunes para fusionar que se manejan de formas ligeramente diferentes, como se describen a continuación. La primera página del asistente de fusión le pide seleccionar el método que necesita.

Fusionando un rango de revisiones.
: Este método cubre el caso en el que ha hecho una o más revisiones a una rama (o al tronco) y desea portar estos cambios a una rama diferente.

  Lo que le está pidiendo a Subversion que haga es: “Calcula los cambios necesarios para ir [DESDE] la revisión 1 de la rama A [HASTA] la revisión 7 de la rama A, y aplica esos cambios en mi copia de trabajo (del tronco o de la rama B).”

Reintegrando una rama.
: Este método cubre el caso cuando ha realizado una rama de funcionalidad como se discute en el libro de Subversion. Todos los cambios del tronco han sido portados a la rama de la funcionalidad, semana a semana, y ahora que la funcionalidad está completa desea fusionarla de nuevo en el tronco. Dado que ha mantenido la rama de la funcionalidad sincronizada con el tronco, las últimas versiones de la rama y del tronco serán absolutamente iguales excepto por sus cambios en la rama.

  Este es un caso especial de la fusión de árboles describa más abajo, y necesita sólo la URL desde la que fusionar (normalmente) su rama de desarrollo. Utiliza las características de registro de fusiones de Subversion para calcular el rango de revisiones correcto que se debe utilizar, y realiza comprobaciones adicionales para asegurarse de que la rama ha sido completamente actualizada con los cambios del tronco. Esto le asegura que no va a deshacer accidentalmente el trabajo que otros han confirmado en el tronco desde sus últimos cambios sincronizados.

  Tras la fusión, todo el desarrollo en las ramas ha sido completamente fusionado de vuelta a la línea principal. La rama ahora es redundante y puede eliminarse.

  Una vez que haya realizado una fusión de reintegración no debería continuar usándola para desarrollo. La razón para esto es que si intenta resincronizar su rama existente desde el tronco más adelante, el registro de fusión verá su reintegración como un cambio del tronco que aún no ha sido fusionado en la rama, ¡e intentará fusionar la fusión rama-a-tronco de nuevo en la rama! La solución a esto es simplenete crear una nueva rama desde el tronco para continuar con la siguiente fase de su desarrollo.

Fusionando dos árboles diferentes.
: Este es un caso más general del método de reintegración. Lo que le está pidiendo a Subversion que haga es: “Calcula los cambios necesarios para ir [DESDE] la revisión HEAD del tronco [HASTA] la revisión HEAD de la rama, y aplica esos cambios a mi copia de trabajo (del tronco).” El resultado neto es que el tronco ahora es exactamente igual a la rama.

  Si su servidor/repositorio no soporta el registro de fusiones, entonces esta es la única forma de fusionar una rama de nuevo al tronco (trunk). Otro caso de uso ocurre cuando utiliza ramas de vendedor y necesita fusionar los cambios después de incorporar una nueva versión de ese código en el tronco. Para más información lea el capítulo sobre *ramas de vendedor* [http://svnbook.red-bean.com/en/1.5/svn.advanced.vendorbr.html] en el Libro de Subversion.

4.20.1. Fusionando un rango de revisiones

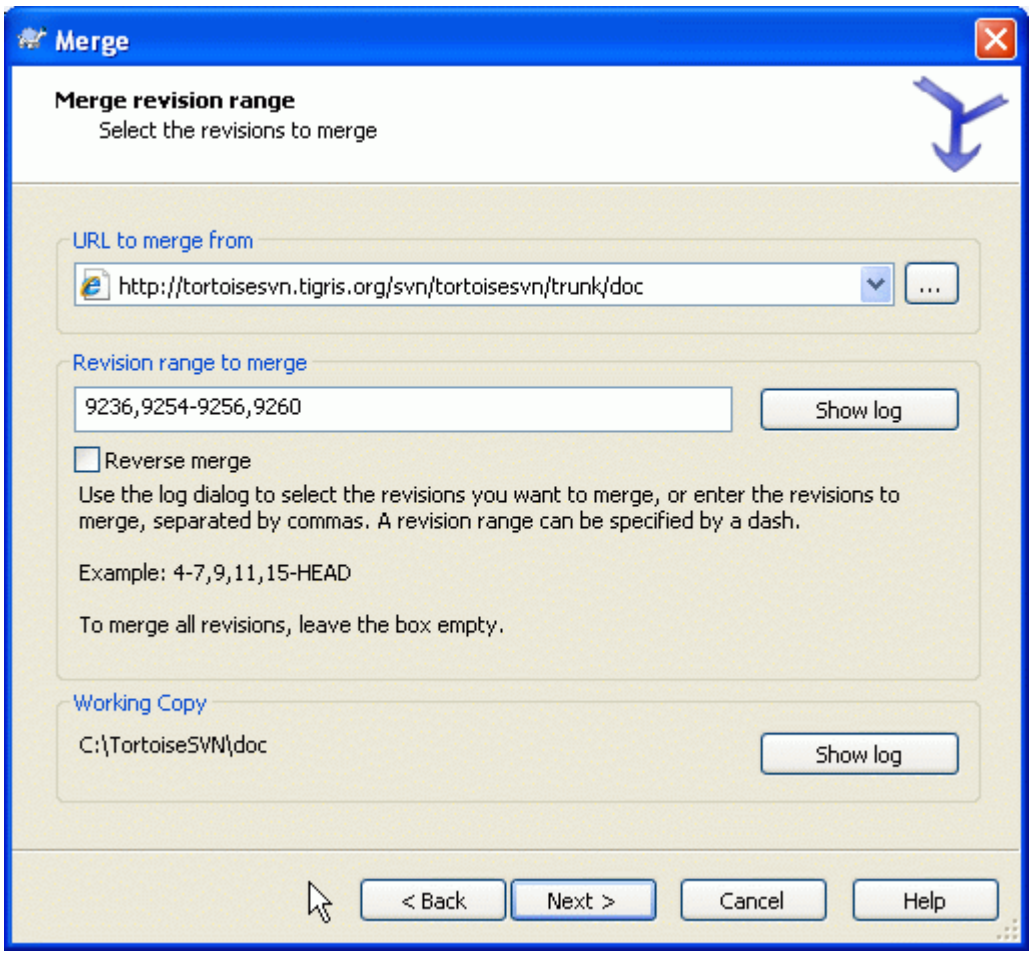


Figura 4.36. El asistente de fusionado - Seleccionar el rango de revisiones

En el campo Desde: introduzca la URL completa de la carpeta de la rama o la etiqueta que contiene los cambios que desea portar a su copia de trabajo. También puede pulsar ... para navegar el repositorio y encontrar la rama deseada. Si ha fusionado desde esta rama antes, entonces utilice la lista desplegable que le muestra una historia de las URLs utilizadas anteriormente.

En el campo Rango de revisiones a fusionar introduzca la lista de revisiones que desea fusionar. Puede ser una única revisión, una lista de revisiones específicas separadas por comas, un rango de revisiones separado por un guión, o cualquier combinación de éstas.

Importante

Hay una diferencia importante en la forma que se especifican los rangos de revisiones en TortoiseSVN comparado con el cliente de línea de comandos. La forma más fácil de visualizar esto es pensar en una valla con postes y paneles.

Con el cliente de línea de comandos se especifican los cambios a fusionar utilizando dos revisiones “postes” que especifican los puntos *anterior* y *posterior*.

Con TortoiseSVN se especifican los conjuntos de cambios a fusionar utilizando “paneles de vallas”. La razón para esto queda más clara cuando utiliza el diálogo de registro para seleccionar las revisiones a fusionar, donde cada revisión aparece como un conjunto de cambios.

> Si está fusionando revisiones en bloques, el método mostrado en el Libro de Subversion le hará fusionar 100-200 esta vez y 200-300 la próxima. Con TortoiseSVN fusionará 100-200 esta vez y 201-300 la próxima.
>
> Esta diferencia ha generado un montón de reacciones en las listas de trabajo. Nos damos cuenta de que es una diferencia respecto al cliente de línea de comandos, pero creemos que para la mayoría de usuarios GUI el método que hemos implementado es más fácil de entender.

La manera más sencilla de seleccionar el rango de revisiones que necesita es pulsar en Mostrar Registro, dado que esto mostrará los cambios recientes con sus comentarios de registro. Si desea fusionar los cambios de una única revisión, seleccione esa revisión. Si desea fusionar los cambios de varias revisiones, entonces seleccione ese rango (utilizando el modificador habitual **Mayúsculas**). Pulse Aceptar y la lista de números de revisión a fusionar se rellenará automáticamente.

Si desea fusionar los cambios de nuevo *fuera* de su copia de trabajo, para revertir un cambio que ya ha sido confirmado, seleccoine las revisiones a revertir y asegúrese de que la casilla Fusión inversa está marcada.

Si ya ha fusionado algunos cambios desde esta rama, esperemos que haya apuntado la última revisión fusionada en el mensaje de registro cuando confirmó ese cambio. En ese caso, puede utilizar Mostrar Registro en la Copia de Trabajo para buscar ese mensaje de registro. Recordando que estamos pensando en las revisiones como conjuntos de cambios, debería utilizar la revisión siguiente al punto final de la última fusión como punto de inicio para esta fusión. Por ejemplo, si la última vez fusionó las revisiones de la 37 a la 39, entonces el punto de inicio de esta fusión sería la revisión 40.

Si está utilizando las características de registro de fusiones de Subversion, no necesita recordar qué revisiones ya han sido fusionadas - Subversion lo apuntará por usted. Si deja el rango de revisiones en blanco, se incluirán todas las revisiones que no hayan sido aún fusionadas. Lea Sección 4.20.6, “Registro de fusión” para saber más.

Si hay otras personas que puedan estar confirmando cambios, entonces tenga cuidado al utilizar la revisión HEAD. Puede que no se refiera a la revisión que usted cree si alguien más ha hecho una confirmación después de su última actualización.

Haga click en Siguiente y vaya a Sección 4.20.4, “Opciones de fusión”

4.20.2. Reintegrando una rama.

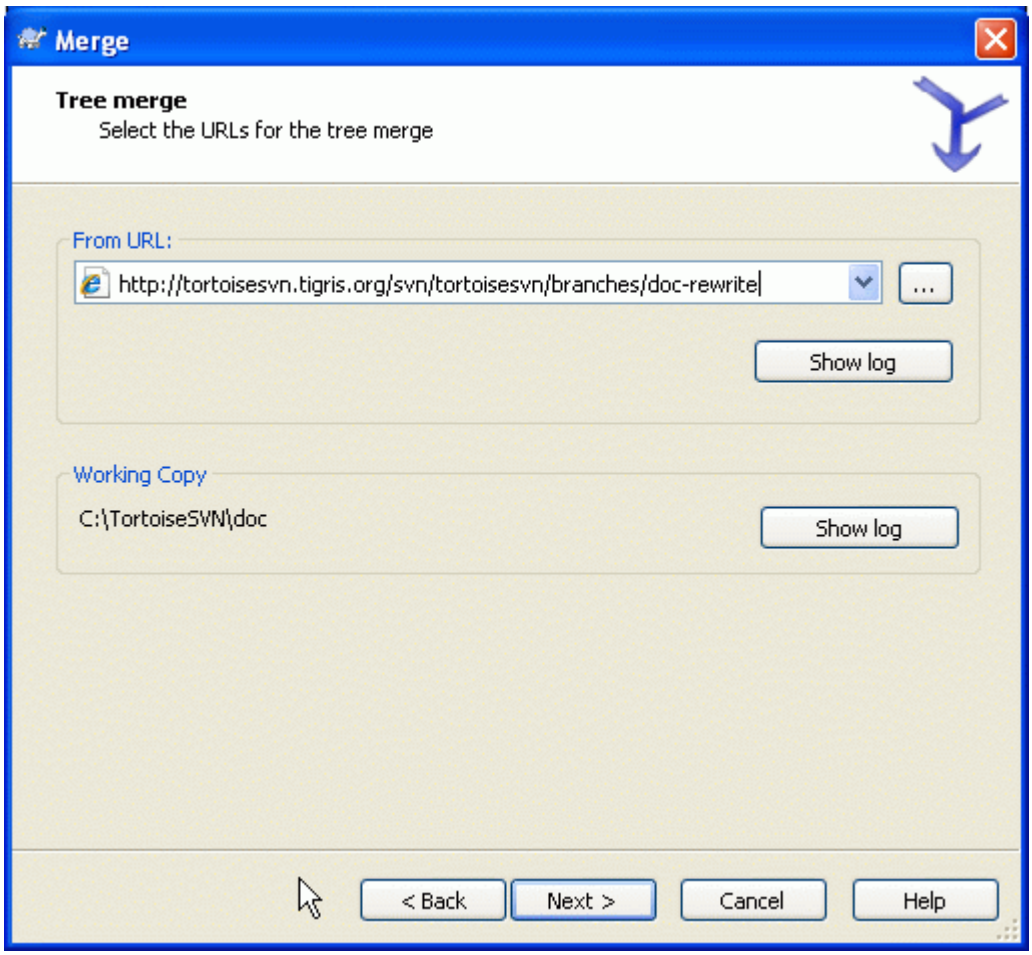


Figura 4.37. El asistente de fusión - Fusionar reintegración

Para fusionar una rama de la característica en el tronco necesita iniciar el asistente de fusión desde una copia de trabajo del tronco.

En el campo Desde la URL: introduzca la URL completa de la carpeta que quiere fusionar. También puede pulsar ... para navegar el repositorio.

Hay algunas condiciones que se aplican a la fusión de reintegración. En primer lugar, el servidor debe soportar el registro de fusiones. La copia de trabajo debe ser de profundidad infinita (sin obtenciones parciales), y no debe tener ninguna modificación local, ítems apuntando a otra ruta o ítems que hayan sido actualizados a otras revisiones distintas de HEAD. Todos los cambios del tronco hechos durante el desarrollo de la rama deben haberse fusionado en la rama (o marcados como que ya han sido fusionados). El rago de revisiones para la fusión se calculará automáticamente.

4.20.3. Fusionando dos árboles diferentes

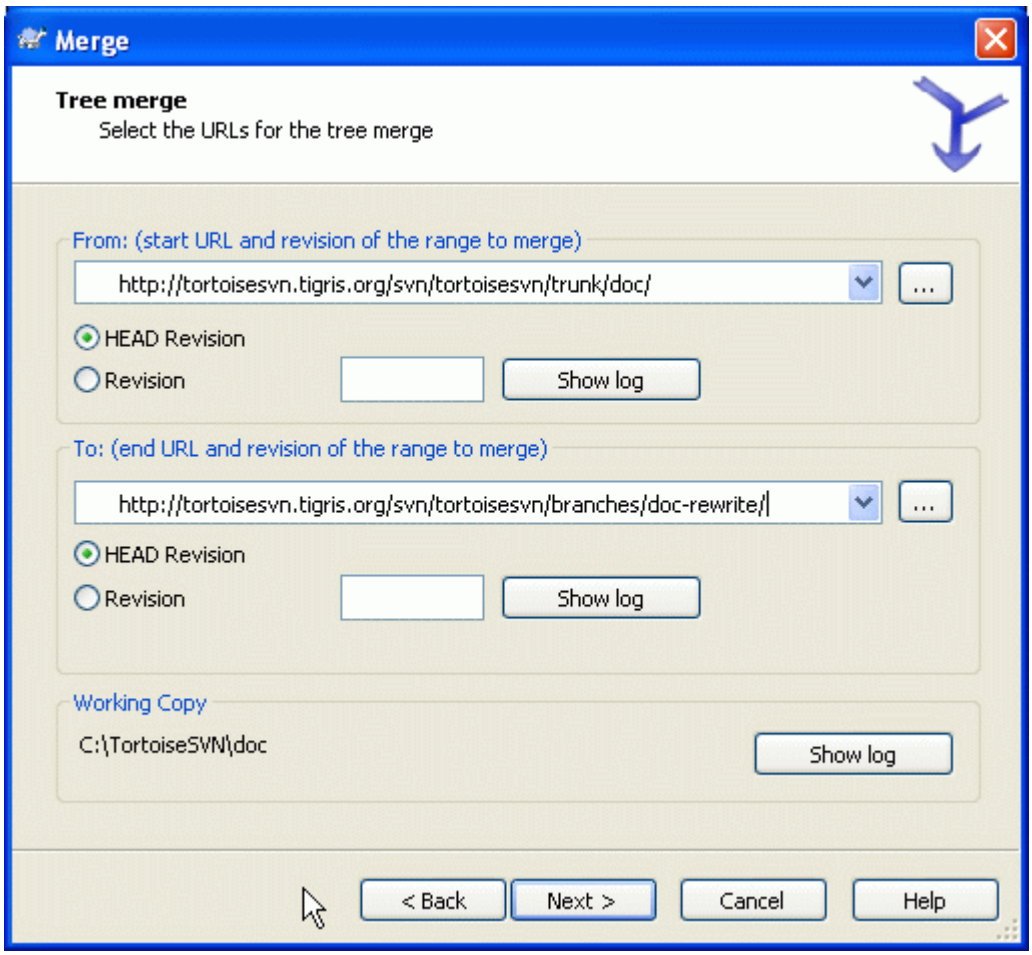


Figura 4.38. El asistente de fusión - Fusión de árboles

Si está utilizando este método para fusioanr una rama de característica de nuevo en el tronco, necesitará iniciar el asistente de fusión desde una copia de trabajo del tronco.

En el campo Desde: introduzca la URL completa del *tronco* (trunk). Esto puede sonar erróneo, pero recuerde que el tronco es el punto de inicio al que desea añadir los cambios de la rama. También puede pulsar ... para navegar el repositorio.

En el campo A: introduzca la URL completa de la rama de la característica.

En los dos campos Desde la Revisión y Hasta la Revisión, introduzca el último número de revisión en el que se sincronizaron los dos árboles. Si está seguro de que nadie más está haciendo confirmaciones puede utilizar HEAD en ambos casos. Si hay alguna posibilidad de que alguien haya hecho una confirmación desde esa sincronización, utilice el número de revisión específico para evitar perder cambios más recientes.

También puede usar Mostrar registro para seleccionar la revisión.

4.20.4. Opciones de fusión

Esta página del asistente le permite especificar opciones avanzadas antes de empezar el proceso de fusión. La mayoría de las veces puede simplemente utilizar las opciones por defecto.

Puede especificar la profundidad de la fusión, es decir, cuántos niveles debe bajar la fusión en su copia de trabajo. Los términos de profundidad se describen en Sección 4.3.1, “Profundidad de obtención”. La

profundidad por defecto es Copia de trabajo, que utiliza la configuración de profundidad existente, y que casi siempre es lo que quiere hacer.

La mayoría del tiempo deseará que la fusión tenga en cuenta el historial de un fichero, para que se fusionen los cambios relativos a un ancestro en común. A veces puede necesitar fusionar ficheros que quizás estén relacionados, pero no en su repositorio. Por ejemplo, puede haber importado las versiones 1 y 2 de una biblioteca de terceros en dos directorios separados. Aunque están relacionados de forma lógica, Subversion no tiene conocimiento de ello porque sólo ve los ficheros que ha importado. Si intenta fusionar las diferencias entre estos dos ficheros vería un borrado completo seguido de un adición completa. Para hacer que Subversion utilice únicamente las diferencias basadas-en-ruta en vez de diferencias basadas-en-historial, seleccione la casilla Ignorar ancestros. Puede leer más sobre este tema en el libro de Subversion; *Teniendo en cuenta o ignorando los ancestros* [http://svnbook.red-bean.com/en/1.5/svn.branchmerge.advanced.html#svn.branchmerge.advanced.ancestry]

Puede especificar la forma en la que se manejan los cambios en los finales de línea y en los espacios en blanco. Estas opciones se describen en Sección 4.10.2, “Opciones de fín de línea y espacios en blanco”. El comportamiento por defecto es tratar todos los espacios en blanco y las diferencias en los finales de líneas como cambios reales que deben fusionarse.

Si está utilizando el registro de fusiones y desea marcar una revisión como que ya se ha fusionado, sin hacer realmente efectiva la fusión aquí, marque la casilla Sólo registrar la fusión. Hay dos posibles razones por las que puede querer hacer esto. Puede ser que la fusión es demasiado complicada para los algoritmos de fusión, por lo que hace los cambios en el código a mano, y luego marca el cambio como fusionado para que el algoritmo de registro de fusión se de por enterado. O quizás quiera evitar que una revisión en concreto se fusione. Marcándola como ya fusionada evitará que la fusión se produzca con los clientes manejan la información de registro de fusión.

Ahora que todo está preparado, todo lo que debe hacer es pulsar en el botón Fusionar. Si quiere previsualizar los resultados, el botón Probar fusión realiza la operación de la fusión, pero *no* modifica la copia de trabajo en absoluto. Le muestra una lista de ficheros que serán cambiados por la fusión real, y le notifica las áreas donde ocurrirán conflictos.

El diálogo de progreso de fusión muestra cada etapa de la fusión, con los rangos de revisiones involucrados. Esto puede indicar un número de revisión más del que esperaba. Por ejemplo, si pidió fusionar la revisión 123, el diálogo de progreso mostrará “Fusionando de r122 a r123”. Para entenderlo debe recordar que Fusionar está íntimamente relacionado con Diferenciar. El proceso de fusión funciona generando una lista de diferencias entre dos puntos del repositorio, y posteriormente aplicando esas diferencias a su copia de trabajo. El diálogo de progreso está simplemente mostrando los puntos de inicio y fin de la diferenciación.

4.20.5. Revisando los resultados de la fusión

La fusión ya ha concluído. Es una buena idea observar la fusión y comprobar si el resultado es el esperado. Normalmente fusionar es un proceso complicado. A menudo aparecen conflictos cuando la rama se ha desviado mucho del tronco.

Los clientes y servidores de Subversion anteriores a la versión 1.5 no almacenan la información de fusión, y las revisiones fusionadas deben ser registradas de forma manual. Cuando haya comprobado los cambios y vaya a confirmar esta revisión, su mensaje de registro de confirmación debería incluir *siempre* los números de revisión que han sido portados en la fusión. Si desea aplicar otra fusión más tarde, necesitará saber lo que ya ha fusionado, porque no querrá portar un cambio más de una vez. Puede encontrar más información sobre esto en *Las mejores prácticas para fusionar* [http://svnbook.red-bean.com/en/1.4/svn.branchmerge.copychanges.html#svn.branchmerge.copychanges.bestprac] en el libro de Subversion.

Si su servidor y todos los clientes están ejecutando Subverison 1.5 o superior, las facilidades de registro de fusiones apuntarán las revisiones fusionadas y evitarán que una revisión se fusione más de una vez. Esto le facilita mucho la vida ya que puede simplemente fusionar el rango entero de revisiones cada vez y saber qué sólo las nuevas revisiones serán realmente fusionadas.

El manejo de las ramas es importante. Si desea mantener esta rama actualizada respecto al tronco, debería asegurarse de fusionar a menudo para que la rama y el tronco no se alejen mucho. Por supuesto, sigue debiendo evitar fusiones repetidas de cambios, como se explicó anteriormente.

Sugerencia

Si acaba de fusionar una rama de característica en el tronco, el tronco contiene ahora todo el código de la nueva característica, y la rama es obsoleta. Si lo necesita, puede borrar la rama del repositorio.

Importante

Subversion no puede fusionar un fichero con una carpeta, y viceversa - sólo carpetas con carpetas y ficheros con ficheros. Si pulsa en un fichero y abre el diálogo de fusionar, entonces deberá darle una ruta a un fichero en ese diálogo. Si selecciona una carpeta y llama al diálogo, debe especificar una URL de una carpeta para la fusión.

4.20.6. Registro de fusión

Subversion 1.5 introdujo facilidades para el registro de fusiones.. Cuando fusiona cambios de un árbol en otro, los números de revisión fusionados se almacenan, y esta información se puede utilizar para diferentes propósitos.

- Puede evitar el peligro de fusionar la misma revisión dos veces (problema de fusiones repetidas). Una vez que una revisión se ha marcado como que ha sido fusionada, las fusiones futuras que incluyan esa revisión en el rango la saltarán.
- Cuando fusiona una rama de nuevo al tronco, el diálogo de registro le puede mostrar las confirmaciones de la rama como parte del registro del tronco, proporcionando una mejor trazabilidad de los cambios.
- Cuando se meustra el diálogo de registro desde el diálogo de fusión, las revisiones ya fusionadas aparecen en gris.
- Cuando muestre información de autoría para un fichero, puede elegir mostrar el autor original de la revisión fusionada, en vez de la persona que hizo la fusión.
- Puede marcar revisiones como *no fusionar* si las incluye en la lista de revisiones fusionadas sin ejecutar relamente la fusión.

La información de registro de fusiones se almacena en la propiedad `svn:mergeinfo` por parte del cliente cuando realiza la fusión. Cuando la fusión se confirma, el servidor almacena dicha información en una base de datos, y cuando le pide información de la fusión, registro o autoría, el servidor puede responer apropiadamente. Para que el sistema funcione correctamente debe asegurarse de que el servidor y todos los clientes estén actualizados. Los clientes antiguos no almacenarán la propiedad `svn:mergeinfo` y los servidores anteriores no proporcionarán la información que piden los nuevos clientes.

Averigue más sobre el registro de fusión desde la *Documentación de registro de fusión* [http://subversion.tigris.org/merge-tracking/index.html] de Subversion.

4.20.7. Manejando conflictos durante la fusión

Las fusiones no siempre van bien. A menudo ocurre algún conflicto, y si está fusionando múltiples rangos, generalmente querrá resolver el conflicto antes de que comience la fusión del siguiente rango. TortoiseSVN le ayuda durante este proceso mostrándole el diálogo *información de conflicto de fusión*.

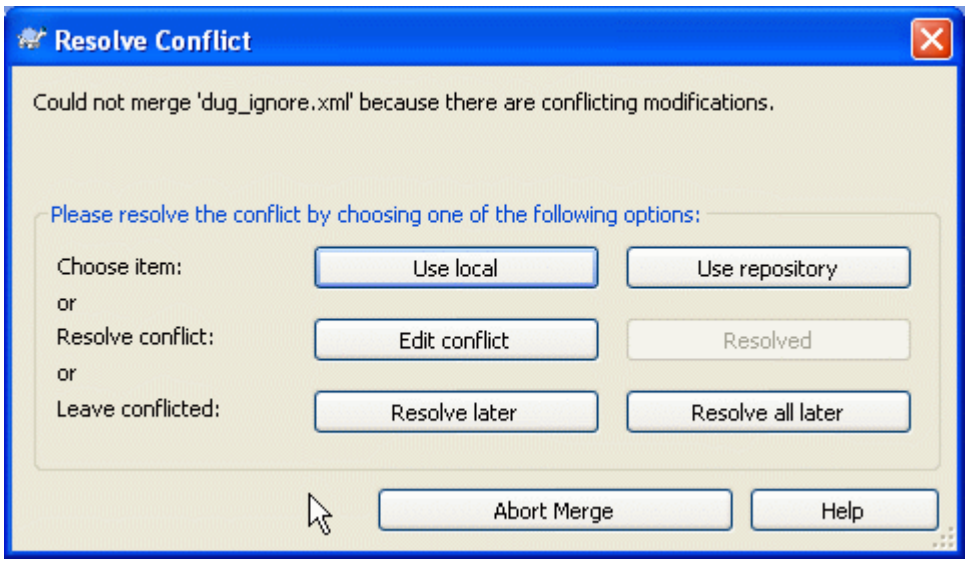


Figura 4.39. El diálogo Información de conflicto de fusión

Cuando ocurre un conflicto durante la fusión, tiene tres formas de manejarlo.

1. Puede decidir que sus cambios locales son mucho más importantes, por lo que querrá descartar la versión del repositorio y mantener su versión local. O puede querer descartar sus cambios locales en favor de la versión del repositorio. De cualquier forma, no se intenta fusionar los cambios - simplemente elige una u otra.

2. Normalmente querrá ver los conflictos y resolverlos. En ese caso, seleccione Editar conflicto, lo que iniciará su herramienta de fusionado. Cuando esté satisfecho con el resultado, pulse en Resuelto.

3. La última opción es posponer la resolución y continuar con el fusionado. Puede elegir hacerlo para el fichero en conflicto actual, o para todos los ficheros durante el resto de la fusión. Sin embargo, si hay más cambios en ese fichero, no será posible completar la fusión.

Si no desea utilizar esta llamada interactiva, hay una casilla en el diálogo de progreso de fusión Fusión no-interactiva. Si esto se marca para una fusión y la fusión resulltaría en un conflicto, el fichero se marca como en conflicto y la fusión continua. Tendrá que resolver los conflictos después de que se complete la fusión. Si no se marca, entonces tendrá la oportunidad de resolver el conflicto *durante* la fusión antes de que se marque como en conflicto. Esto tiene la ventaja de que si un fichero tiene múltiplas fusiones (múltiples revisiones que aplican un cambio sobre ese fichero), puede que las siguientes fusiones sean satisfactorias dependiendo de qué líneas se vean afectadas. Pero por supuesto no puede irse a tomar café mientras se ejecuta la fusión ;)

4.20.8. Fusionar una rama completada

Si desea fusionar todos los cambios desde una rama de característica al tronco, entonces puede utilizar el comando TortoiseSVN → Fusionar reintegración... desde el menú contextual extendido (mantenga pulsada la tecla **Mayúsculas** mientras hace click con el botón derecho en el fichero).

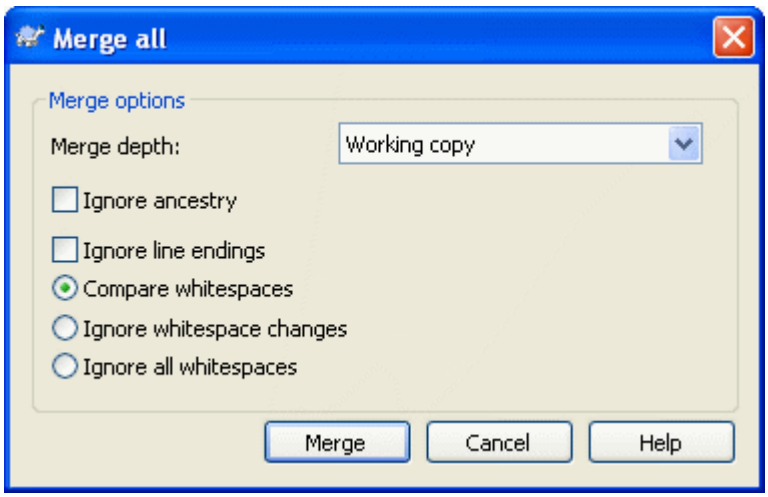


Figura 4.40. El diálogo Fusionar reintegración

Este diálogo es muy sencillo. Todo lo que tiene que hacer el establecer las opciones para la fusión, como se describe en Sección 4.20.4, “Opciones de fusión”. El resto se hace por TortoiseSVN automáticamente utilizando el registro de fusiones.

4.20.9. Mantenimiento de ramas de características

Cuando desarrolla una nueva característica en una rama separada es una buena idea mantener una política de re-integración cuando la característica está completa. Si se está modificando a la vez `trunk` (el tronco), puede que encuentre que las diferencias se vuelven significativas según pasa el tiempo, y la fusión se convierte en una pesadilla.

Si la característica es relativamente simple y el desarrollo no llevará mucho tiempo, puede adoptar un camino más simple, que es mantener la rama totalmente separada hasta que la característica se completa, y luego fusionar los cambios de la rama de nuevo en el tronco. En el asistente de fusión esto sería un simple Fusionar un rango de revisiones, donde el rango de revisiones es la rango de revisiones de la rama.

Si la característica va a llevar más tiempo y necesita tener en cuenta los cambios en `trunk`, entonces necesita mantener las ramas sincronizadas. Esto simplemente significa que periódicamente fusionará el tronco en la rama, para que la rama contenga todos los cambios de la rama *además* de la nueva característica. El proceso de sincronización utiliza Fusionar un rango de revisiones. Cuando la característica esté completa, podrá fusionarla de nueva en `trunk` utilizando o bien Reintegrar una rama o Fusionar dos árboles distintos.

4.21. Bloqueando

Subversion generalmente trabaja mejor sin bloqueos, utilizando los métodos “Copiar-Modificar-Fusionar” que se describieron anteriormente en el Sección 2.2.3, “La solución copiar-modificar-fusionar”. Sin embargo, hay algunas pocas situaciones en las que puede querer implementar alguna forma de política de bloqueo.

- Está utilizando ficheros “no fusionables”, por ejemplo, ficheros de imagen. Si dos personas cambian el mismo fichero, la fusión no es posible, así que alguno de ellos perderá sus cambios.
- Su compañía ha utilizado en el pasado un sistema de control de versiones bloqueante, y la dirección ha decidido que “bloquear es lo mejor”.

Primero necesita asegurarse de que su servidor de Subversion está actualizado al menos a la versión 1.2. Las versiones anteriores no tienen ningún soporte de bloqueos. Si está usando el acceso `file://`, sólo necesitará actualizar sus clientes, por supuesto.

4.21.1. Cómo trabaja el bloqueo en Subversion

Por defecto, nada se bloquea y todo el mundo que tiene acceso de confirmación puede confirmar cambios a cualquier fichero en cualquier momento. Los demás actualizarán sus copias de trabajo periódicamente y los cambios en el repositorio se fusionarán con los cambios locales.

Si *Obtiene un Bloqueo* en un fichero, entonces sólo usted puede confirmar ese fichero. Las confirmaciones de los demás se bloquearán hasta que quite el bloqueo. Un fichero bloqueado no puede ser modificado de ninguna forma en el repositorio, por lo que no puede ser siquiera borrado o renombrado, excepto por el dueño del bloqueo.

Sin embargo, los demás usuarios no necesariamente saben que usted ha obtenido un bloqueo. A menos que ellos comprueben el estado de bloqueo regularmente, la primera vez que sabrán sobre él es cuando sus confirmaciones fallen, lo que muchas veces no es muy útil. Para hacer más fácil el manejo de bloqueos, hay una nueva propiedad de Subversion `svn:needs-lock`. Cuando se establece esta propiedad (a cualquier valor) en un fichero, siempre que el fichero se obtiene o actualiza, la copia local se deja como sólo-lectura *a menos* que la copia de trabajo tenga un bloqueo para ese fichero. Esto actúa como una advertencia de que no debería modificar el fichero a menos que obtenga un bloqueo. Los ficheros que están versionados y marcados como sólo-lectura se marcan con un icono de sobreimpresión especial en TortoiseSVN para indicar que necesita obtener un bloqueo antes de editar.

Los bloqueos se graban con el lugar de la copia local y también con el propietario. Si tiene varias copias de trabajo (en casa, en el trabajo) entonces sólo puede obtener un bloqueo en *una* de esas copias de trabajo.

Si uno de sus compañeros de trabajo obtiene un bloqueo y luego se va de vacaciones sin quitarlo, ¿qué hace? Subversion proporciona un método para forcar bloqueos. Quitar un bloqueo que tiene otra persona se define como *Romper* el bloqueo, y obtener forzosamente un bloqueo que tiene otro se define como *Robar* el bloqueo. Naturlamente estas no son cosas que hacer a la ligera si desea continuar siendo amigo de sus compañeros de trabajo.

Los bloqueos se guardan en el repositorio, y se crea un token de bloqueo en su copia local de trabajo. Si hay una discrepancia, por ejemplo si alguien ha roto el bloqueo, el token local se convierte en inválido. El repositorio siempre es la referencia definitiva.

4.21.2. Obteniendo un bloqueo

Seleccione el fichero o ficheros en su copia de trabajo para los que desee obtener un bloqueo, y seleccione el comando TortoiseSVN → Obtener Bloqueo....

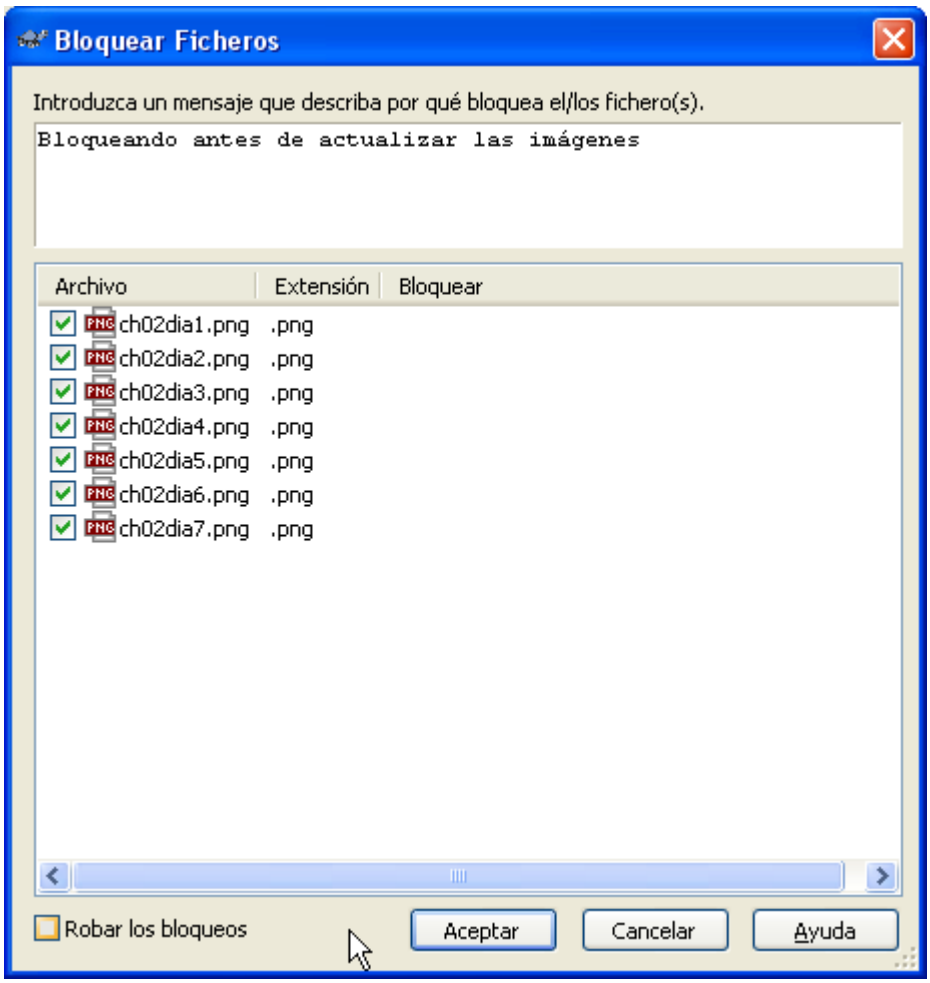


Figura 4.41. El diálogo Bloquear

Aparece un diálogo, que le permite introducir un comentario, para que los demás vean por qué ha bloqueado el fichero. El comentario es opcional, y de momento sólo se utiliza con repositorios basados en Svnserve. Si (y *sólo* si) necesita robar el bloqueo de alguien, marque la casilla Robar bloqueo, y luego pulse Aceptar.

Si selecciona una carpeta y luego utiliza TortoiseSVN → Obtener Bloqueo... se abrirá el diálogo de bloquear con *todos* los ficheros en *todas* las subcarpetas seleccionados para bloquearlos. Si realmente quiere bloquear una jerarquía completa, ésta es la forma de conseguirlo, pero puede ganarse la antipatía de sus compañeros de trabajo si les bloquea todo el proyecto. Utilícelo con cuidado...

4.21.3. Quitando un Bloqueo

Para asegurarse de que no se olvida de quitar un bloqueo que no necesita más, los ficheros bloqueados se muestran en el diálogo de confirmar y se seleccionan por defecto. Si continúa con la confirmación, los bloqueos que tenga sobre los ficheros seleccionados se quitan, incluso si los ficheros no se han modificado. Si no desea quitar el bloqueo en algunos ficheros, puede desmarcarlos (si no están modificados). Si desea mantener los bloqueos en un fichero que ha modificado, tiene que habilitar la casilla Mantener bloqueos antes de confirmar sus cambios.

Para quitar un bloqueo manualmente, seleccione el fichero o ficheros de su copia de trabajo para los que desee quitar el bloqueo, y luego seleccione el comando TortoiseSVN → Quitar Bloqueo. No hay nada más que introducir por lo que TortoiseSVN contactará con el repositorio y quita los bloqueos. También puede utilizar este comando en una carpeta para quitar todos los bloqueos recursivamente.

4.21.4. Comprobando el estado de los bloqueos

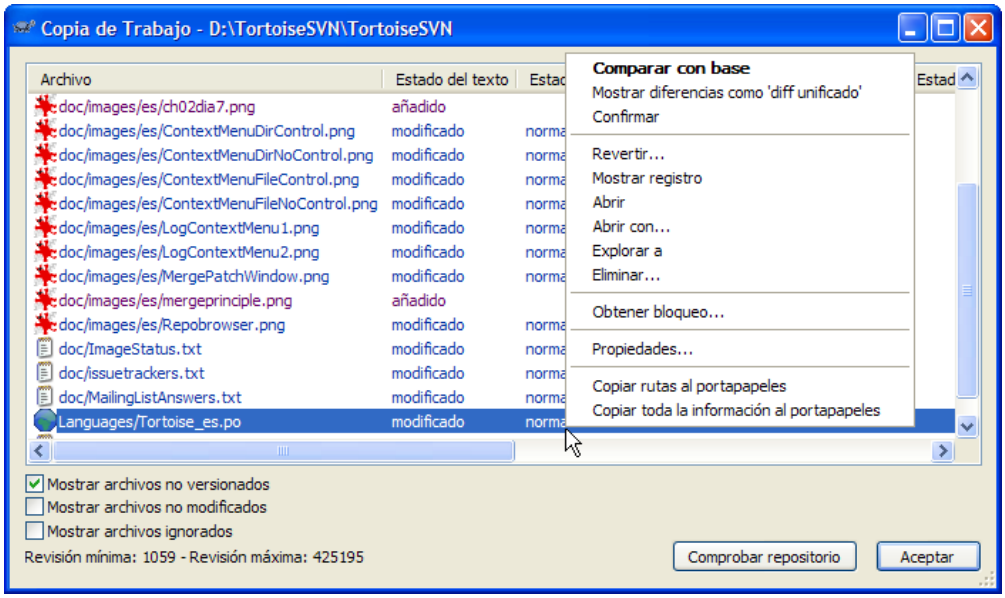


Figura 4.42. El diálogo Comprobar modificaciones

Para ver qué bloqueos tiene usted y los de los demás, puede utilizar el comando TortoiseSVN → Comprobar Modificaciones.... Los tokens de bloqueos que se tienen localmente se muestran inmediatamente. Para comprobar los bloqueos que tienen los demás (y para ver si alguno de sus bloqueos se han roto o han sido robados) necesita pulsar Comprobar Repositorio.

Desde el menú contextual aquí, también puede obtener y quitar bloqueos, además de romper y robar bloqueos que tienen otros.

Evite romper y robar bloqueos

Si ha roto o robado el bloqueo de otro sin decírselo, puede causar una pérdida potencial de trabajo. Si está trabajando con tipos de ficheros no fusionables y roba el bloqueo de otro, una vez que usted quite el bloqueo ellos son libres de confirmar sus cambios y sobreescribir los suyos. Subversion no pierde datos, pero ha perdido la protección para trabajo en equipo que el bloqueo le proporcionaba.

4.21.5. Haciendo ficheros no-bloqueados como sólo-lectura

Como se menciona arriba, la manera más efectiva de utilizar los bloqueos es establecer la propiedad `svn:needs-lock` en los ficheros. Consulte Sección 4.17, “Configuración del proyecto” para obtener instrucciones sobre cómo establecer propiedades. Los ficheros con esta propiedad establecida siempre se obtendrán y actualizarán con la marca de sólo-lectura a menos que su copia de trabajo tenga un bloqueo.

Como recordatorio, TortoiseSVN utiliza un icono de sobreimpresión especial para indicarlo.

Si opera una política donde todos los ficheros han de ser bloqueados, puede que encuentre más fácil utilizar la característica de Subversion auto-props para establecer la propiedad automáticamente cada vez que añada nuevos ficheros. Para más información, lea Sección 4.17.1.5, “Establecer propiedades automáticamente”.

4.21.6. Los scripts ganchos de bloqueo

Cuando crea un nuevo respositorio con Subversion 1.2 o superior, se crean cuatro plantillas en el directorio `hooks` del repositorio. Éstos se llaman antes y después de obtener un bloqueo, y antes y después de quitar un bloqueo.

Es una buena idea instalar un script gancho `post-lock` y `post-unlock` en el servidor que envíe un email indicando el fichero que se ha bloqueado. Poniendo este script en su sitio, todos sus usuarios serán informados si alguien bloquea/quita el bloqueo de un fichero. Puede encontrar un script gancho de ejemplo `hooks/post-lock.tmpl` en su carpeta del repositorio.

También puede utilizar ganchos para no permitir romper o robar bloqueos, o quizás para limitarlo a un administrador designado para ello. O quizás desea enviar un email al propietario cuando uno de sus bloqueos se haya roto o robado.

Para más información, lea la Sección 3.3, “Scripts gancho en el lado del servidor”.

4.22. Creando y aplicando parches

En proyectos de código abierto (como éste), todos tienen acceso de lectura al repositorio, y cualquiera puede hacer una contribución al proyecto. ¿Así que cómo se controlan esas contribuciones? Si cualquiera pudiera confirmar cambios, el proyecto estaría permanentemente inestable y probablemente roto de forma permanente. En esta situación, el cambio se maneja mediante un fichero de *parche* enviado al equipo de desarrollo, que tienen acceso de escritura. Ellos pueden revisar el parche antes, y luego o bien confirmarlo en el repositorio o devolvérselo al autor.

Los ficheros de parche son simplemente ficheros de diff unificados que muestran las diferencias entre su copia de trabajo y la revisión base.

4.22.1. Creando un fichero parche

Primero necesita hacer *y probar* sus cambios. Luego en vez de utilizar el comando TortoiseSVN → Confirmar... sobre la carpeta padre, seleccione TortoiseSVN → Crear Parche...

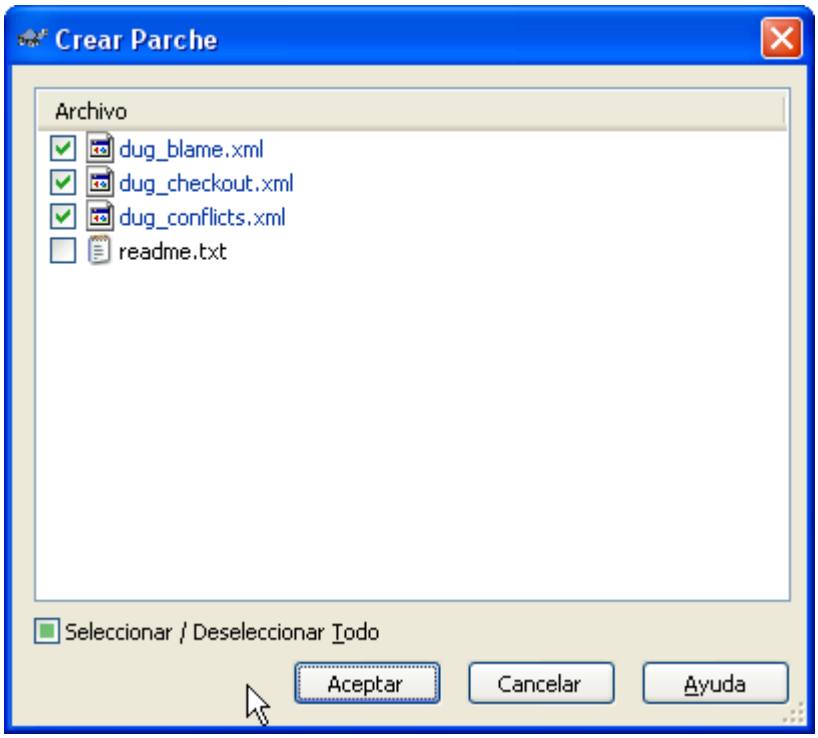


Figura 4.43. El diálogo Crear parche

ahora puede seleccionar los ficheros que desea incluir en el parche, del mismo modo a como lo haría con una confirmación completa. Esto producirá un único fichero que contendrá un resumen de todos los cambios que ha hecho a los ficheros seleccionados desde la última vez que se actualizó desde el repositorio.

Las columnas en este diálogo pueden ser personalizadas de la misma forma que las columnas en el diálogo Comprobar modificaciones. Para más detalles, lea Sección 4.7.3, “Estado local y remoto”.

Puede producir parches separados que contengan cambios a diferentes conjuntos de ficheros. Por supuesto, si crea un fichero de parche, hace algunos cambios más a los *mismos* ficheros y luego crea otro parche, el segundo fichero de parche incluirá *ambos* conjuntos de cambios.

Simplemente grabe el fichero utilizando un nombre de fichero de su elección. Los ficheros de parche pueden tener la extensión que desee, pero por convención se suele utilizar las extensiones `.patch` o `.diff` extension. Ya está preparado para enviar su fichero de parches.

También puede almacenar el parche en el portapapeles en vez de en un fichero. Puede querer hacer esto para poder pegarlo en un email de forma que lo revisen otros. O si tiene dos copias de trabajo en una máquina y quiere transferir cambios de una a otra, un parche en el portapapeles es una forma cómoda de hacerlo.

4.22.2. Aplicando un fichero parche

Los ficheros de parches se aplican en su copia de trabajo. Esto debe hacerse desde el mismo nivel de carpetas que se utilizó para crear el parche. Si no está seguro de cuál es, mire la primera línea del fichero de parche. Por ejemplo, si el primer fichero en el que se trabajó era `doc/source/english/chapter1.xml` y la primera línea en el fichero de parche es `Index: english/chapter1.xml` entonces necesita aplicar el parche en la carpeta `english`. Sin embargo, suponiendo que esté en la copia de trabajo correcta, si ha seleccionado un nivel de carpeta erróneo, TortoiseSVN se dará cuenta y le sugerirá el nivel correcto.

Para aplicar un fichero de parche en su copia de trabajo, necesita al menos acceso de lectura al repositorio. La razón para esto es que el programa de fusión debe referenciar los cambios sobre la revisión contra la que se hicieron por el desarrollador remoto.

Desde el menú contextual de esa carpeta, pulse en TortoiseSVN → Aplicar Parche... Esto mostrará un diálogo de abrir fichero que le permitirá seleccionar el fichero de parche a aplicar. Por defecto sólo aparecen los ficheros `.patch` o `.diff`, pero puede elegir “Todos los ficheros”. Si había guardado previamente un parche en el portapapeles, puede usar Abrir desde el portapapeles... en el diálogo de abrir fichero.

Alternativamente, si el fichero de parche tiene una extensión `.patch` o `.diff`, puede hacer click con el botón derecho en él directamente y seleccioanr TortoiseSVN → Aplicar Parche.... En este caso se le preguntará la ruta de la copia de trabajo.

Estos dos métodos le ofrecen formas diferentes de hacer lo mismo. Con el primer método selecciona la copia de trabajo y busca el fichero de parche. Con el segundo selecciona el fichero de parche y busca la copia de trabajo.

Una vez que haya seleccionado el fichero de parche y la ruta de la copia de trabajo, se ejecuta TortoiseMerge para fusionar los cambios del fichero de parche contra su copia de trabajo. Una pequeña ventana le muestra los ficheros que han sido cambiados. Haga doble click por turnos en cada uno de ellos, compruebe los cambios y grabe los ficheros fusionados.

El parche del desarrollador remoto ya ha sido aplicado a su copia de trabajo, por lo que necesita confirmar para que todos los demás accedan a los cambios desde el repositorio.

4.23. ¿Quién cambió qué línea?

A veces necesita saber no sólo qué lineas han cambiado, sino también exactamente quién cambió líneas específicas en un fichero. Entonces es cuando el comando TortoiseSVN → Autoría..., a veces conocido como comando de *anotar*, tiene su utilidad.

Este comando muestra, por cada línea en un fichero, su autor y la revisión en la que se cambió la línea.

4.23.1. Autoría de ficheros

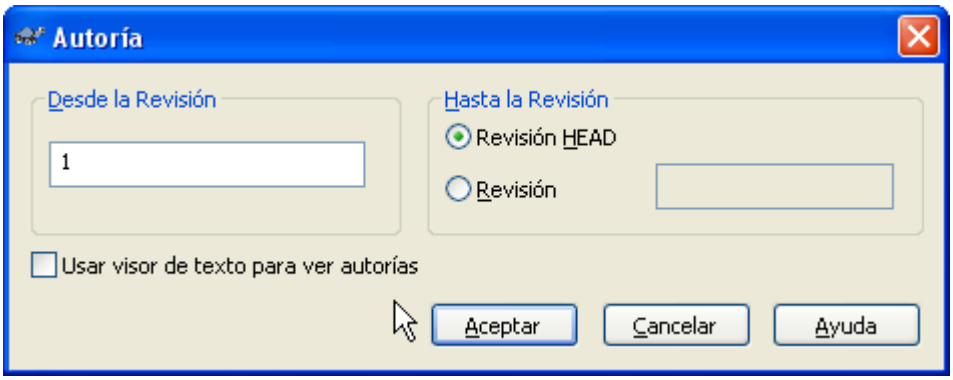


Figura 4.44. El diálogo Anotar / Autoría

Si no está interesado en cambios de revisiones anteriores puede establecer la revisión desde la cual debe empezar la autoría. Ponga este valor a `1` si desea ver la autoría de *cada* revisión.

Por defecto, el fichero de autoría se ve utilizando *TortoiseBlame*, que remarca las diferentes revisiones para hacerlas más fáciles de leer. Si desea imprimir o editar el fichero de autoría, seleccione Utilizar visor de texto para ver autorías

Puede especificar la forma en la que se manejarán los cambios en los finales de línea y en los espacios en blanco. Estas opciones se describen en Sección 4.10.2, “Opciones de fín de línea y espacios en blanco”. El comportamiento por defecto es tratar todas las diferencias en los espacios en blanco y en los finales de línea como cambios reales, pero si desea ignorar un cambio en la indentación y encontrar al autor original, puede elegir la opción apropiada aquí.

Una vez que pulse Aceptar, TortoiseSVN empieza a recoger la información para crear el fichero de autoría. Tenga ésto en cuenta: esto puede llevar varios minutos para completarse, dependiendo en cuánto haya cambiado el fichero y por supuesto de su conexión de red con el repositorio. Una vez que el proceso de autoría ha terminado, el resultado se escribe en un fichero temporal y puede ver los resultados.

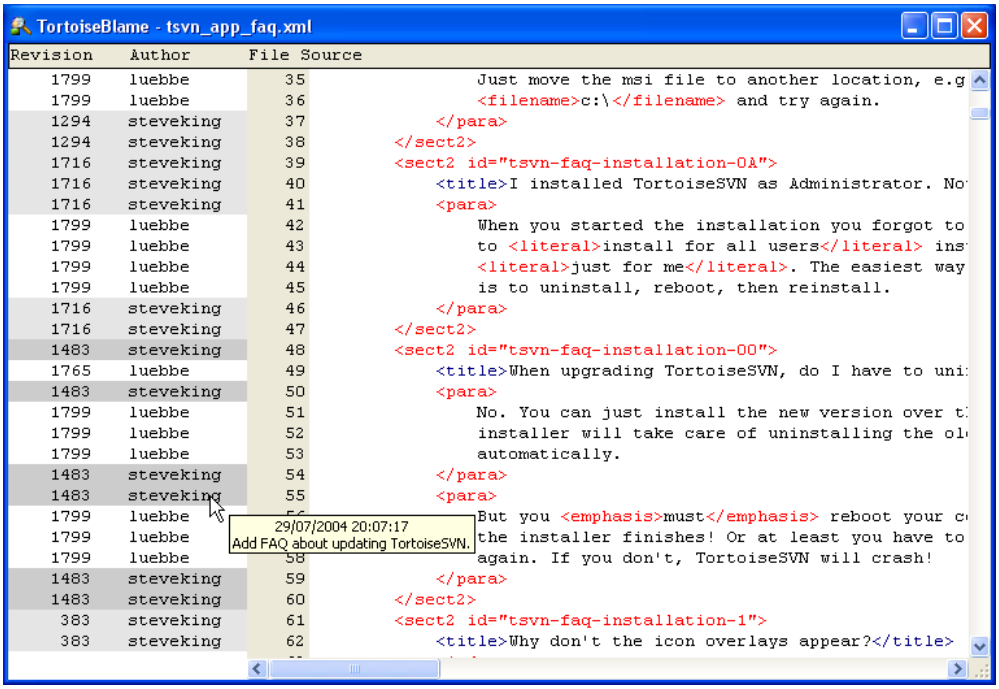


Figura 4.45. TortoiseBlame

TortoiseBlame, que se incluye con TortoiseSVN, hace más fáciles de leer los ficheros de autoría. Cuando pasa el ratón por encima de una línea en la columna de información de autoría, todas las líneas con la misma revisión se señalan con un fondo más oscuro. Las líneas de otras revisiones que fueron cambiadas por el mismo autor se señalan con un fondo claro. Los colores pueden no funcionar de forma muy eficiente si su pantalla está en el modo de 256 colores.

Si hace click en una línea, todas las líneas con la misma revisión se señalan, y las líneas de otras revisiones del mismo autor se señalan en un color más claro. Este señalado se mantiene, permitiéndole mover el ratón sin perder los señalados. Pulse en esa revisión de nuevo para desactivar el señalado.

Los comentarios de la revisión (mensaje de registro) se muestra en un texto de ayuda cuando se pasa el cursor sobre la columna de información de autoría. Si quiere copiar el mensaje de registro de esa revisión, utilice el menú contextual que aparece cuando hace click con el botón derecho sobre la columna de información de autoría.

Puede buscar dentro del informe de Autoría utilizando Edición → Buscar.... Esto le permite buscar por números de revisión, autores y por el contenido del fichero en si mismo. Los mensajes de registro no se incluyen en la búsqueda - deberá utilizar el diálogo de Mostrar Registro para buscar en ellos.

También puede ir directamente a un número de línea concreto utilizando Editar → Ir a la línea....

Cuando el ratón se coloca encima de las columnas de información de autoría, tiene disponible un menú contextual que le ayuda a comparar revisiones y examinar la historia, utilizando el número de revisión bajo el ratón como referencia. Menú contextual → Ver autoría de la revisión anterior genera un informe de autoría para el mismo fichero, pero utilizando la revisión anterior como límite superior. Esto le da el informe de autoría para el estado del fichero justo antes de que la línea que está viendo cambiara. Menú contextual → Mostrar cambios inicia su visor de diferencias, mostrándole lo que cambió en la revisión referenciada. Menú contextual → Mostrar registro le muestra el diálogo de registro de revisiones empezando en la revisión referenciada.

Si necesita un indicador visual mejor de dónde están los cambios más antiguos y más nuevos, seleccione Ver → Colorear la antigüedad de las líneas. Esto utilizará un gradiente de color para mostrar las

líneas más nuevas en rojo y las más antiguas en azul. La paleta de color por defecto es bastante suave, pero puede cambiarla utilizando la configuración de TortoiseBlame.

Si está utilizando Registro de fusiones, donde las líneas hayan cambiado como resultado de una fusión desde otra ruta, TortoiseBlame le mostrará la revisión y el autor del último cambio en el fichero original, en vez de la revisión en la que se realizó la fusión. Estas líneas se muestran con la revisión y el autor en cursiva. Si no desea que las líneas fusionadas se muestren de esta forma, desmarque la casilla Incluir información de fusión.

Si quiere ver las rutas involucradas en la fusión, seleccione Ver → Rutas de fusión.

Las configuraciones para TortoiseBlame se pueden acceder utilizando TortoiseSVN → Configuración... en la pestaña TortoiseBlame. Para más información, vea Sección 4.30.9, “Configuración de TortoiseBlame”.

4.23.2. Autoría de las diferencias

Una de las limitaciones del informe de Autoría es que sólo muestra el fichero tal y como estaba en una revisión en concreto, y muestra la última persona que cambió cada línea. A veces querrá saber qué cambio se hizo, y también quién lo hizo. Lo que necesita aquí es una combinación de los informes de diferencias y autoría.

El diálogo del historial de revisiones incluye varias opciones que le permiten hacerlo.

Autoría de las revisiones
: En el panel superior, seleccione dos revisiones, y luego seleccione Menú Contextual → Ver autoría de las revisiones. Esto obtendrá los datos de autoría de las dos revisiones, y luego utilizará el visor de diferencias para comparar los dos ficheros de autoría.

Ver la autoría de los cambios
: Seleccione una revisión en el panel superior, luego seleccione un fichero en el panel inferior y seleccione Menú contextual → Autoría de los cambios. Esto obtendrá los datos de autoría de la revisión seleccionada y la anterior, y utilizará el visor de diferencias para comparar los dos ficheros de autoría.

Comparar y ver autoría con la BASE de trabajo
: Muestre el registro de un único fichero, y en el panel superior, seleccione una única revisión, y luego seleccione Menú contextual → Comparar y ver autoría con la BASE de trabajo. Esto obtendrá los datos de autoría de la revisión seleccionada, y también del fichero en la BASE de trabajo, y utilizará el visor de diferencias para comparar los dos ficheros de autoría.

4.24. El navegador de repositorios

A veces necesitará trabajar directamente en el repositorio, sin tener una copia de trabajo. Ésa es la razón por la que existe el *Navegador de Repositorios*. Igual que el Explorador y los iconos sobreimpresionados le permiten ver su copia de trabajo, el navegador de repositorios le permite ver la estructura y el estado del repositorio.

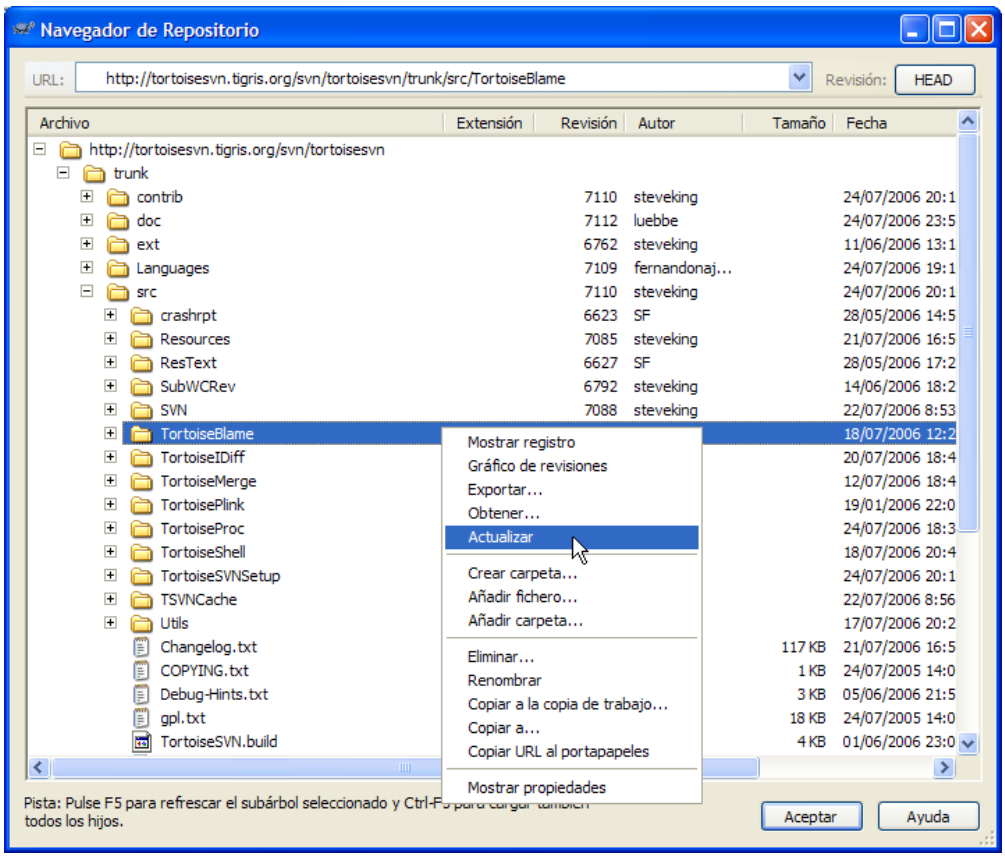


Figura 4.46. El navegador de repositorios

Con el navegador de repositorios puede ejecutar comandos como copiar, mover, renombrar... directamente en el repositorio.

El navegador de repositorios se parecerá al explorador de Windows, excepto que está mostrando el contenido del repositorio en una revisión concreta en vez de los ficheros de su ordenador. En el panel izquierdo puede ver un árbol de directorios, y en el panel derecho están los contenidos del directorio seleccionado. En la parte superior de la ventana del visor de repositorios puede introducir la URL del repositorio y la revisión que desea visualizar.

Al igual que el explorador de Windows, puede pulsar sobre los encabezados de las columnas en el panel derecho si desea establecer la ordenación. Y como en el explorador hay menús contextuales en ambos paneles.

El menú contextual para un fichero le permite:

- Abrir el fichero seleccionado, bien con el visor por defecto para ese tipo de fichero, o bien con el programa que elija.
- Guardar una copia no versionada del fichero en su disco duro.
- Mostrar el registro de las revisiones para ese fichero, o mostrar un gráfico de todas las revisiones para que pueda ver de dónde vino el fichero.
- Ver la autoría del fichero, para saber quién cambió qué línea y cuándo.
- Eliminar o renombrar el fichero.
- Hacer una copia del fichero, bien a una parte diferente del repositorio, o bien a una copia de trabajo incluída dentro del mismo repositorio.
- Ver/editar las propiedades del fichero.

El menú contextual para una carpeta le parmite:

- Mostrar el registro de revisiones para la carpeta, o mostrar un gráfico de todas las revisiones para que pueda ver de dónde vino la carpeta.
- Exportar la carpeta a una copia local no versionada de su disco duro.
- Obtener la carpeta para producir una copia de trabajo local en su disco duro.
- Crear una nueva carpeta en el repositorio.
- Añadir ficheros o carpetas directamente al repositorio.
- Borrar o eliminar la carpeta.
- Hacer una copia de la carpeta, bien a una parte diferente del repositorio, o bien a una copia de trabajo dentro del mismo repositorio.
- Ver/Editar las propiedades de la carpeta.
- Marcar la carpeta para comparación. La carpeta marcada se mostrará en negrita.
- Comparar la carpeta con una carpeta marcada previamente, bien como diff unificado, o bien como una lista de ficheros que pueden diferenciarse visualmente utilizando la herramienta de diferencias por defecto. Esto puede ser particularmente útil para comparar dos etiquetas, o el tronco y una rama para ver qué se ha cambiado.

Si selecciona dos carpetas en el panel derecho, puede ver las diferencias bien como diff unificado, o bien como una lista de ficheros que pueden diferenciarse visualmente utilizando la herramienta de diferencias por defecto.

Si selecciona múltiples carpetas en el panel derecho, puede obtenerlos todos a la vez en una carpeta padre común.

Si selecciona dos etiquetas que se han copiado desde la misma raíz (tipicamente, `/trunk/`), puede utilizar Menú Contextual → Mostrar Registro... para ver la lista de revisiones entre los dos puntos etiquetados.

Puede utilizar **F5** para refrescar la vista como siempre. Esto refrescará todo lo que se esté viendo actualmente. Si quiere precargar o refrescar la información para los nodos que aún no han sido abiertos, utilice **Ctrl-F5**. Después de eso, la expansión de cualquier nodo ocurrirá automáticamente sin que haya una pausa por la red mientras se recoge la información.

También puede utilizar el navegador de repositorios para operaciones de arrastrar-y-soltar. Si arrastra una carpeta desde el Explorador al navegador de repositorios, se importará esa carpeta en el repositorio. Tenga en cuenta que si arrastra múltiples items, se importarán en confirmaciones separadas.

Si desea mover un ítem dentro del repositorio, simplemente arrastre con el botón izquierdo el ítem a su nuevo lugar. Si desea crear una copia en vez de mover el ítem, entonces deberá **Ctrl**-arrastrar con el botón izquierdo. Cuando copia, el cursor tiene un símbolo “más” en él, tal y como lo hace en el Explorador.

Si desea copiar/mover un fichero o una carpeta a otro lugar y darle también otro nombre a la vez, puede arrastrar con el botón derecho o **Ctrl**-arrastrar con el botón derecho el ítem en vez de arrastrarlo con el botón izquierdo. En ese caso, aparece un diálogo en el que puede introducir un nuevo nombre para el fichero o la carpeta.

Siempre que haga cambios en el repositorio utiizando uno de estos métodos, se le presentará un diálogo para introducir un mensaje de registro. Si ha arrastrado algo por error, esta es su oportunidad para cancelar la acción.

A veces cuando intenta abrir una ruta obtendrá un mensaje de error en vez de los detalles del ítem. Esto puede ocurrir si ha especificado una URL inválida, si no tiene permisos de acceso, o si hay algún

otro problema en el servidor. Si necesita copiar este mensaje para incluirlo en un email, simplemente haga click con el botón derecho sobre él y utilice Menú Contextual → Copiar mensaje de error al portapapeles, o simplemente utilice **Ctrl+C**.

4.25. Gráficos de revisión

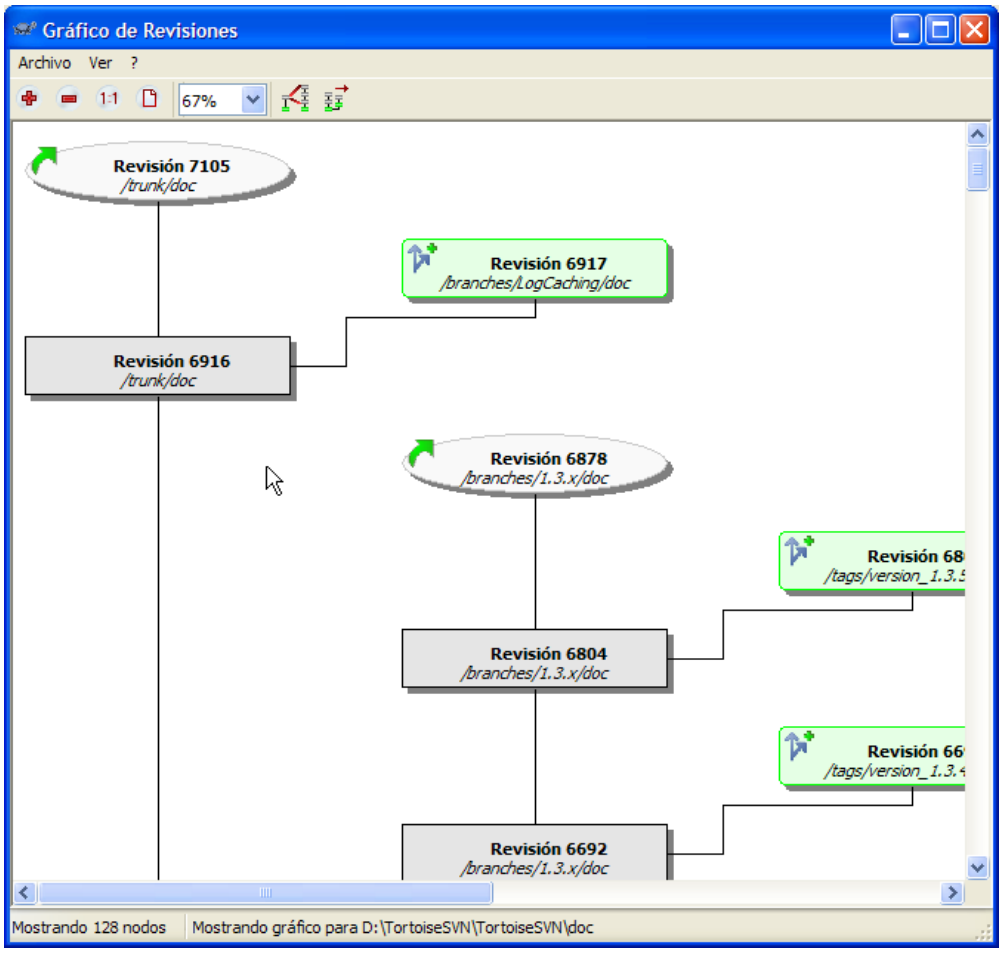


Figura 4.47. Un gráfico de revisiones

A veces necesita saber desde qué revisión del tronco se tomaron las ramas y las etiquetas, y la forma ideal de ver este tipo de información es en un gráfico o una estructura de árbol. Ahí es cuando necesita utilizar TortoiseSVN → Gráfico de Revisiones...

Este comando analiza la historia de las revisiones e intenta crear un árbol mostrando los puntos en los que se tomaron las copias, y cuando se borraron las ramas/etiquetas.

Importante

Para generar el gráfico, TortoiseSVN debe obtener todos los mensajes de registro desde la raíz del repositorio. No es necesario decir que esto puede llevar varios minutos incluso con un repositorio con unos pocos miles de revisiones, dependiendo de la velocidad del servidor, el ancho de banda de la red, etc. Si intenta esto con algo como el proyecto *Apache*, que actualmente tiene más de 500.000 revisiones, puede estar esperando un buen rato.

The good news is that if you are using log caching, you only have to suffer this delay once. After that, log data is held locally. Log caching is enabled in TortoiseSVN's settings.

4.25.1. Nodos del gráfico de revisión

Cada nodo de gráfico de revisión representa una revisión en el repositorio donde algo cambió en el árbol que está viendo. Los diferentes tipos de nodos se pueden distinguir por forma y color. Las formas son fijas, pero los colores se pueden cambiar utilizando TortoiseSVN → Configuración

Ítems añadidos o copiados
: Items which have been added, or created by copying another file/folder are shown using a rounded rectangle. The default colour is green. Tags and trunks are treated as a special case and use a different shade, depending on the TortoiseSVN → Settings

Ïtems eliminados
: Los ítems borrados, por ejemplo una rama que no se necesita más, se muestran utilizando un octágono (rectángulo con las esquinas cortadas). El color por defecto es rojo.

Ítems renombrados
: Los ítems renombrados también se muestran utilizando un octágono, pero el color por defecto es azul.

Revisión punta de la rama
: El gráfico normalmente se restringe para mostrar los puntos de ramas, pero a menudo es útil poder ver las revisiones HEAD respectivas de cada rama. Si selecciona Mostrar revisiones HEAD, cada nodo de revisión HEAD se mostrará como una elipse. Tenga en cuenta que HEAD aquí hace referencia a la última revisión confirmada en esa ruta, no a la revisión HEAD del repositorio.

Revisión de la copia de trabajo
: Si ha invocado el gráfico de revisiones desde una copia de trabajo, puede optar por mostrar la revisión BASE del gráfico utilizando Mostrar revisión copia de trabajo, que marca el nodo BASE con un borde remarcado.

Copia de trabajo modificada
: Si ha invocado el gráfico de revisiones desde una copia de trabajo, puede optar por mostrar un nodo adicional representando su copia de trabajo modificada utilizando Mostrar en la copia de trabajo. Todos los dem

Tenga en cuenta que por defecto el gráfico sólo muestra los puntos en los que se añadieron, copiaron o borraron ítems. Mostrar cada revisión de un proyecto generará un gráfico muy grande para casos no triviales. Si realmente desea ver *todas* las revisiones en las que se hicieron cambios, hay una opción para conseguirlo en el menú Ver y en la barra de herramientas.

La vista por defecto (sin agrupación) pone los nodos con su posición vertical en estricto orden de revisión, para que tenga una pista visual del órden en el que se hicieron las cosas. Cuando dos nodos están en la misma columna, el órden es muy obvio. Cuando dos nodos están en columnas contiguas la separación es mucho menor porque no hay necesidad de evitar que los nodos se sobrepongan, y como resultado el órden es un poco menos obvio. Estas optimizaciones son necesarias para mantener gráficos complejos en un tamaño razonable. Por favor tenga en cuenta que esta ordenación utiliza el *borde* del nodo en el lado *más antiguo* como referencia, es decir, el borde inferior del nodo cuando el gráfico se muestra con el nodo más antiguo en la parte inferior. El borde de referencia es importante porque no todas las formas de los nodos tienen la misma altura.

4.25.2. Cambiando la vista

Dado que los gráficos de revisiones son a menudo complejos, hay un número de funcionalidades que pueden utilizarse para ajustar la vista a lo que desea obtener. Estas opciones están disponibles en el menú Ver y en la barra de herramientas.

Agrupar ramas
: The default behavior (grouping off) has all rows sorted strictly by revision. As a result, long-living branches with sparse commits occupy a whole column for only a few changes and the graph becomes very broad.

Este modo agrupa los cambios por ramas, de forma que no hay una ordenación global por revisiones: las revisiones consecutivas en una rama se mostrarán (a menudo) como líneas consecutivas. Las sub-ramas, sin embargo, se colocarán de forma que las ramas más nuevas se muestren encima de las ramas más antiguas para mantener el gráfico más delgado. Como resultado, una fila cualquier puede contener cambios de diferentes revisiones.

Más antiguo arriba
: Normalmente el gráfico muestra la revisión más antigua abajo, y el árbol crece hacia arriba. Utilice esta opción para hacer crecer el árbol desde arriba.

Alinear árboles en la parte superior
: Cuando un gráfico se parte en varios árboles más peuqeños, los árboles pueden aparecer bien en su órden de revisión natural, o alineados en la parte inferior de la ventana, dependiendo de si utiliza la opción Agrupar por ramas. Utilice esta opción para hacer crecer todos los árboles hacia abajo desde arriba.

Reducir líneas cruzadas
: If the layout of the graph has produced a lot of crossing lines, use this option to clean it up. This may make the layout columns appear in less logical places, for example in a diagonal line rather than a column, and the graph may require a larger area to draw.

Nombres de ruta diferenciales
: Long path names can take a lot of space and make the node boxes very large. Use this option to show only the changed part of a path, replacing the common part with dots. E.g. if you create a branch `/branches/1.2.x/doc/html` from `/trunk/doc/html` the branch could be shown in compact form as `/branches/1.2.x/..` because the last two levels, `doc` and `html`, did not change.

Mostrar todas las revisiones
: Esto hace exactamente lo que espera, y muestra cada revisión donde algo (en el árbol del que está mostrando el gráfico) ha cambiado. Para historiales largos esto puede producir un árbol realmente enorme.

Mostrar revisiones HEAD
: Esto asegura que la última revisión de cada rama se muestra siempre en el gráfico.

Orígenes de copia exactos
: When a branch/tag is made, the default behaviour is to show the branch as taken from the last node where a change was made. Strictly speaking this is inaccurate since the branches are often made from the current HEAD rather than a specific revision. So it is possible to show the more correct (but less useful) revision that was used to create the copy. Note that this revision may be younger than the HEAD revision of the source branch.

Plegar etiquetas
: When a project has many tags, showing every tag as a separate node on the graph takes a lot of space and obscures the more interesting development branch structure. At the same time you may need to be able to access the tag content easily so that you can compare revisions. This option hides the nodes for tags and shows them instead in the tooltip for the node that they were copied from. A tag icon on the right side of the source node indicates that tags were made.

Ocultar rutas eliminadas
: Oculta las rutas que ya no están presentes en la revisión HEAD del repositorio, por ejemplo ramas eliminadas.

Ocultar ramas no modificadas
: Hides branches where no changes were committed to the respective file or sub-folder. This does not necessarily indicate that the branch was not used, just that no changes were made to *this* part of it.

Mostrar la revisión de la copia de trabajo
: Marca la revisión en el gráfico que corresponde a la revisión de actualización del ítem para el que obtuvo el gráfico. Si acaba de actualizar, esto será HEAD, pero si otros han confirmado cambios

desde su última actualización, su copia de trabajo puede estar unas revisiones por debajo. Este nodo se marca dándole un borde remarcado.

Mostrar modificaciones en la copia de trabajo
: Si su copia de trabajo contiene cambios locales, esta opción lo dibuja como un nodo separado elíptico, enlazado al nodo al que su copia de trabajo fue actualizado por última vez. El color de resalte por defecto es rojo. Puede necesitar refrescar el gráfico utilizando **F5** para capturar cambios recientes.

Filtro
: A veces los gráficos de revisión contienen más revisiones de las que desea ver. Esta opción abre un diálogo que le permitirá restringir el rango de revisiones mostrado, y ocultar rutas concretas por su nombre.

Bandas de árboles
: Where the graph contains several trees, it is sometimes useful to use alternating colours on the background to help distinguish between trees.

Mostrar resumen
: Shows a small picture of the entire graph, with the current view window as a rectangle which you can drag. This allows you to navigate the graph more easily. Note that for very large graphs the overview may become useless due to the extreme zoom factor and will therefore not be shown in such cases.

4.25.3. Usando el gráfico

Para hacer más fácil la navegación en un gráfico grande, utilice la ventana sobreimpresionada. En ella se muestra el gráfico completo en una pequeña ventana, con la parte que se muestra actualmente iluminada. Puede arrastrar el área iluminada para cambiar la región que se muestra.

La fecha de la revisión, el autor y los comentarios se muestran en una caja de ayuda cuando se mueve el ratón encima de una caja de revisión.

Si selecciona dos revisiones (mediante **Ctrl**-click con el botón izquierdo), puede utilizar el menú contextual para mostrar las diferencias entre esas revisiones. Puede elegir mostrar las diferencias en los puntos de creación de las ramas, pero normalmente querrá mostrar las diferencias en los puntos finales de las ramas, esto es, en la revisión HEAD.

Puede ver las diferencias como un fichero de Diff Unificado, que le muestra todas las diferencias en un fichero único con contexto mínimo. Si opta por Menú Contextual → Comparar Revisiones se le presentará una lista de ficheros cambiados. Haga doble click en un nombre de fichero para obtener ambas revisiones y compararlas utilizando una herramienta de diferencias visual.

Si hace click con el botón derecho en una revisión, puede utilizar Menú Contextual → Mostrar registro para ver la historia.

También puede fusionar cambios de la(s) revision(es) seleccionada(s) en una copia de trabajo distinta. Un diálogo de selección de carpeta le permitirá elegir la copia de trabajo donde desea fusionar, pero después de eso no hay diálogo de confirmación, ni oportunidad de probar la fusión sin ejecutarla realmente. Es una buena idea fusionar en una copia de trabajo sin cambios, ¡y así poder revertir los cambios si no funcionan! Esta es una funcionalidad útil si desea fusionar las revisiones seleccionadas de una rama a otra.

Aprenda a leer el Gráfico de Revisiones

Los nuevos usuarios pueden sorprenderse al notar que el gráfico de la revisión no coincide con lo que se imaginaba o esperaba. Si una revision modifica varias copias o ramas de un archivo o carpeta, en ese caso se veran varios nodos para esa revisión. Es un buen hábito comenzar utilizando las opciones que se encuentran mas a la izquierda en la barra de herramientas y personalizarel grafico paso a paso hasta que se vuelva mas parecido a lo que el usuario espera.

> Todas las opciones de filtro intentan perder la menor cantidad de información posible. Esto puede causar eventualmente, que algunos nodos pierdan su color. Cuando el resultado sea inesperado, deshaga la ultima operacion de filtrado e intente comprender que es lo extraordinario en la revisión o ramar particular. En la mayoría de los casos, la inicial salida inesperada de la operación de filtrado, será imprecisa y/o errónea.

4.25.4. Refrescando la vista

Si desea comprobar el servidor de nuevo para obtener la información más reciente, puede simplemente refrescar la vista utilizando **F5**. Si está utilizando la caché de registro (habilitada por defecto), esto buscará en el repositorio las confirmaciones más recientes y traerá sólo las nuevas. Si la caché de registro estaba en modo desconectado, esto intentará volver a ponerla en línea.

Si está utilizando la caché de registros y cree que el contenido del mensaje o el autor han podido cambiar, debería utilizar el diálogo de registro para refrescar los mensajes que necesita. Dado que el gráfico de revisiones trabaja desde la raíz del repositorio, tendríamos que invlidar la caché completa de registro, y recargarla podría llevar *muchísimo* tiempo.

4.25.5. Podando árboles

Un árbol grande puede ser dificil para navegar y a veces querrá ocultar partes de él, o partirlo en un bosque de árboles más pequeños. Si mueve el ratón sobre el punto donde un enlace de nodo entra o sale del nodo, verá que aparecen uno o más botones que le permitirán hacer esto.

Pulse en el botón menos para colapsar el subárbol adjunto.

Pulse el botón más para expandir un árbol colapsado. Cuando un árbol se ha colapsado, este botón permanece visible para indicar un subárbol oculto.

Pulse en el botón cruz para separar el subárbol adjunto y mostrarlo como un árbol separado en el gráfico.

Pulse en el botón círculo para readjuntar un árbol separado Cuando se ha separado un árbol, este botón permanece visible para indicar que hay un subárbol separado.

Haga clic en el fondo del gráfico para acceder al menú contextual principal, el cual ofrece opciones para Expandir todo y Unir todo. Si no se dividió o colapsó ninguna rama, el menú contextual no se mostrará.

4.26. Exportando una copia de trabajo de Subversion

A veces querrá una copia de su árbol de trabajo sin ninguno de esos directorios `.svn`, por ejemplo para crear un fichero comprimido de sus fuentes, o para exportarlo a un servidor web. En vez de hacer una copia y borrar todos esos directorios `.svn` manualmente, TortoiseSVN ofrece el comando TortoiseSVN → Exportar.... Las operaciones exportar desde una URL y exportar desde una copia de trabajo se tratan de forma ligeramente distinta.

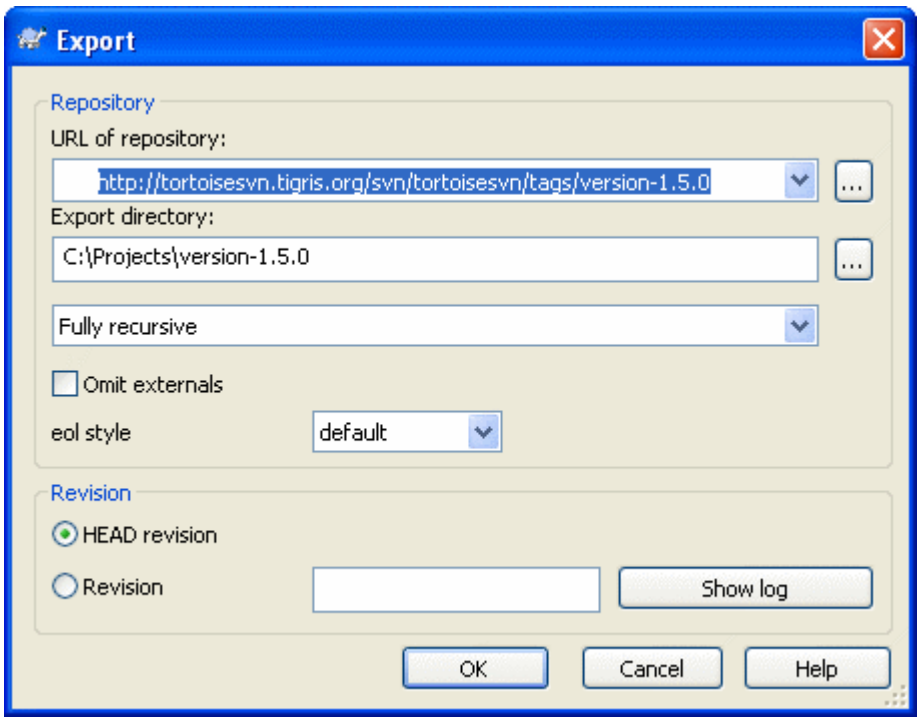


Figura 4.48. El diálogo Exportar-desde-URL

Si ejecuta este comando sobre una carpeta sin versionar, TortoiseSVN asumirá que la carpeta de destino es el objetivo, y abrirá un diálogo para que introduzca la URL y la revisión desde la que desea realizar la exportación. Este diálogo tiene opciones para exportar sólo la carpeta de más alto nivel, para omitir las referencias externas, y para obligar un estilo de fin de línea concreto en aquellos ficheros que tuvieran establecida la propiedad `svn:eol-style`.

Por supuesto también puede exportar directamente desde el repositorio. Utilice el Navegador de repositorios para navegar al subárbol relevante en su repositorio, y luego utilice Menú contextual → Exportar. Obtendrá el diálogo Exportar desde URL descrito más arriba.

Si ejecuta este comando en su copia de trabajo, se le preguntará por el lugar donde guardar la copia de trabajo *limpia* sin las carpetas `.svn`. Por defecto, sólo se exportan los ficheros versionados, pero puede utilizar la casilla Exportar también los ficheros o versionados para incluir cualquier otro fichero no versionado que exista en su copia de trabajo y no esté en el repositorio. Las referencias externas utilizando `svn:externals` pueden omitirse si es necesario.

Otra forma de exportar su copia de trabajo es arrastrar con el botón derecho la carpeta con la copia de trabajo a otro lugar y elegir Menú contextual → SVN Exportar aquí o Menú contextual → SVN Exportar todo aquí. La segunda opción incluirá también los ficheros sin versionar.

Cuando exporte desde una copia de trabajo, si en la carpeta de destino ya existe otra con el nombre de la que está exportando, se le dará la opción de sobreescribir el contenido existente, o de crear una nueva carpeta con un nombre generado automáticamente, por ejemplo, `Destino (1)`.

Exportando ficheros sueltos

El diálogo de exportar no permite exporar ficheros sueltos, aunque Subversion puede hacerlo.

Para exportar ficheros sueltos con TortoiseSVN, tendrá que utilizar el navegador de respositorios (Sección 4.24, “El navegador de repositorios”). Simplemente arrastre el o los ficheros que desea exportar desde el navegador de repositorios a donde los desee tener en

el explorador, o utilice el menú contextual del navegador de repositorios para exportar los ficheros.

Exportando un árbol de cambios

Si quiere exportar una copia de su estructura de árbol del proyecto pero conteniendo sólo los ficheros que han cambiado en una revisión en concreto, o entre dos revisiones, utilice la característica de comparar revisiones descrita en Sección 4.10.3, “Comparando carpetas”.

4.26.1. Eliminando una copia de trabajo del control de versiones

A veces tiene una copia de trabajo que desea reconvertir en una carpeta normal sin los directorios `.svn`. Lo que realmente necesita es un comando exportar-en-el-sitio, que simplemente elimine las carpetas de control, en vez de generar un nuevo árbol limpio.

La respuesta es sorprendentemente sencilla - ¡exporte el árbol en si mismo! TortoiseSVN detecta este caso especial y le pregunta si quiere convertir su copia de trabajo en no-versionada. Si responde *sí*, los directorios de control se eliminan y tendrá un simple árbol sin directorios de control.

4.27. Relocalizando una copia de trabajo

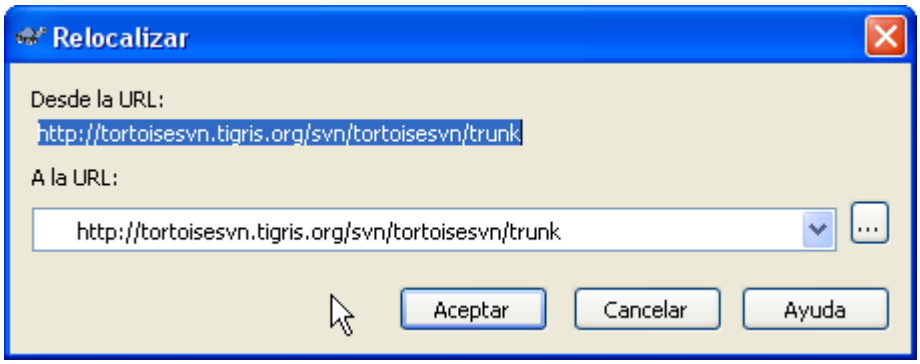


Figura 4.49. El diálogo Relocalizar

Si su repositorio ha cambiado por algún motivo de lugar (IP/URL); o quizás está estancado y no puede confirmar y no quiere obtener de nuevo su copia de trabajo de la nueva localización y mover todos sus datos cambiados a la copia de trabajo nueva: el comando TortoiseSVN → Relocalizar es lo que está buscando. Básicamente hace muy poco: escanea todos los ficheros `entries` en las carpetas `.svn` y cambia la URL de las entradas al nuevo valor.

Puede sorprenderse al ver que TortoiseSVN contacta el repositorio como parte de esta operación. Todo lo que hace es realizar algunas comprobaciones sencillas para asegurarse que la nueva URL realmente se refiere al mismo repositorio que la copia de trabajo existente.

Aviso

Esta es una operación que se utiliza realmente poco. El comando relocalizar se utiliza *sólo* si la URL de la raiz del repositorio ha cambiado. Estas son algunas posibles razones:

- La dirección IP del servidor ha cambiado.
- El protocolo ha cambiado (por ejemplo, de http:// a https://).
- La ruta raíz del repositorio ha cambiado en la configuración del servidor.

De otra forma, necesita relocalizar cuando su copia de trabajo se refiere al mismo lugar en el mismo repositorio, pero es el propio repositorio el que se ha movido.

Esto no se aplica si:

- Quiere moverse a un repositorio de Subversion diferente. En ese caso debería realizar una obtención limpia desde la nueva localización del repositorio.
- Quiere cambiar a una rama o a un directorio diferente dentro del mismo repositorio. Para hacer eso debería utilizar TortoiseSVN → Cambiar.... Para más información, lea Sección 4.19.2, “Obtener o cambiar...”.

Si utiliza relocalizar en alguno de los casos anteriores, *corromperá su copia de trabajo* y obtendrá muchos mensajes de error inexplicables cuando actualice, confirme, etc. Una vez que ésto ha ocurrido, el único arreglo es hacer una obtención limpia.

4.28. Integración con sistemas de control de errores / seguimiento de incidencias

Es muy común en el Desarrollo de Software que los cambios se refieran a un ID de error o incidencia específico. A los usuarios de los sistemas de control de errores (seguimiento de incidencias) les gustaría asociar los cambios que hacen en Subversion con un ID específico en su programa de seguimiento de incidencias. La mayoría de programas de seguimiento de incidencias proporcionan un script gancho pre-commit que parsea el mensaje de log para encontrar el ID del error con el que se asocia la confirmación. Esto es de alguna forma propenso a errores, porque se basa en el que usuario escriba el mensaje de registro de forma correcta para que el script gancho pre-commit pueda parsearlo correctamente.

TortoiseSVN puede ayudar al usuario de dos formas:

1. Cuando el usuario introduce un mensaje de registro, puede añadirse una línea bien definida incluyendo el número de incidencia asociada a la confirmación. Esto reduce el riesgo de que el usuario introduzca el número de incidencia que las herramientas de control de errores no puedan parsear correctamente.

   O TortoiseSVN puede remarcar la parte del mensaje de registro introducido que el programa de seguimiento de incidencias reconoce. De esta forma el usuario sabrá que el mensaje de registro introducido puede parsearse correctamente.

2. Cuando el usuario navega por los mensajes de registro, TortoiseSVN crea un enlace desde cada ID de error en el mensaje de registro, que lanza el navegador con la incidencia mencionada.

4.28.1. Añadiendo números de incidencia en los mensajes de registro

Puede integrar una herramienta de control de errores de su elección con TortoiseSVN. Para hacerlo, necesita definir algunas propiedades, que empiezan por `bugtraq:`. Deben establecerse sobre las carpetas: (Sección 4.17, “Configuración del proyecto”)

Hay dos formas de integrar TortoiseSVN con los programas de control de incidencias. Una se basa en simples cadenas de texto, la otra se basa en *expresiones regulares*. Las propiedades que se utilizan en ambos casos son:

bugtraq:url
: Establezca esta propiedad con la URL de su herramienta de control de errores. Debe ser una URI codificada de forma correcta y debe contener `%BUGID%`. `%BUGID%` se reemplazará por el número de incidencia que haya introducido. Esto permite que TortoiseSVN muestre un vínculo en el diálogo de registro, para que cuando esté mirando un registro de revisión pueda saltar directamente a su herramienta de control de errores. No necesita proporcionar esta propiedad, pero entonces TortoiseSVN sólo mostrará el número de incidencia y no el vínculo a ella. Por ejemplo, el proyecto TortoiseSVN utiliza `http://issues.tortoisesvn.net/?do=details&id=%BUGID%`

También puede utilizar URLs relativas en vez de absolutas. Esto es útil en caso de que su control de incidencias esté en el mismo dominio/servidor que su repositorio. En caso de que el nombre de dominio cambie alguna vez, no tendrá que ajustar la propiedad `bugtraq:url`. Hay dos formas de especificar una URL relativa:

Si empieza con la cadena `^/` se asume que es relativa a la raíz del repositorio. Por ejemplo, `^/../?do=details&id=%BUGID%` se resolverá como `http://tortoisesvn.net/?do=details&id=%BUGID%` si su repositorio está en `http://tortoisesvn.net/svn/trunk/`.

Una URL que comience con la cadena `/` se asume que es relativo al nombre del servidor. Por ejemplo, `/?do=details&id=%BUGID%` se resolverá como `http://tortoisesvn.net/?do=details&id=%BUGID%` si su repositorio está localizado en cualquier parte de `http://tortoisesvn.net`.

bugtraq:warnifnoissue
: Establezca esta propiedad a `true` si desea que TortoiseSVN le avise cuando deje en blanco el campo de texto del número de incidencia. Los valores válidos son `true/false`. *Si no se define, se supone `false`.*

4.28.1.1. Número de incidencia en caja de texto

En el caso simple, TortoiseSVN muestra al usuario un campo de entrada separado donde se puede introducir un ID del error. En ese caso se añade una línea adicional al final o al principio del mensaje de registro que el usuario introduce.

bugtraq:message
: Esta propiedad activa el sistema de control de errores en el modo *campo de entrada*. Si se establece esta propiedad, TortoiseSVN le preguntará por el número de incidencia cuando confirme sus cambios. Se utiliza para añadir una línea al final del mensaje de registro. Debe contener `%BUGID%`, que se reemplaza por el número de la incidencia al confirmar. Esto asegura que su registro de confirmación contiene una referencia al número de incidencia que siempre está en un formato consistente y que puede ser parseado por su herramienta de control de errores para asociar el número de incidencia con una confirmación en concreto. Un ejemplo que puede que le sirva es `Incidencia : %BUGID%`, pero esto depende de su herramienta.

bugtraq:append
: Esta propiedad define si el ID del error se añade (true) al final del mensaje de registro o se inserta (false) al inicio del mensaje de registro. Los valores válidos son `true/false`. *Si no se define, se asume `true`, para que los proyectos ya existentes no se rompan.*

bugtraq:label
: Éste es el texto que TortoiseSVN muestra en el diálogo de confirmar para etiquetar el cuadro de texto en el que introduce el número de incidencia. Si no se establece, aparecerá `Bug-ID / Nº-Incid.:`. Tenga en cuenta, sin embargo, que la ventana no se cambiará de tamaño para acomodar esta etiqueta, por lo que debería mantener el tamaño de la etiqueta en menos de 20-25 caracteres.

bugtraq:number
: Si se establece a `true` sólo se permitirán números en el campo de número de incidencia. La coma es una excepción, para que pueda introducir varios números separados por comas. Los valores válidos son `true/false`. *Si no se define, se asumirá `true`.*

4.28.1.2. Números de incidencia utilizando expresiones regulares

En el caso de utilizar *expresiones regulares*, TortoiseSVN no muestra un campo de entrada separado sino que marca, del mensaje de registro que el usuario introduce, la parte que reconoce el programa de seguimiento de incidencias. Esto se realiza mientras el usuario introduce el mensaje de registro. ¡Esto también significa que el ID del error puede introducirse en cualquier parte dentro de un mensaje de registro! Este método es mucho más flexible, y es el que se utiliza en el propio proyecto TortoiseSVN.

bugtraq:logregex
: Esta propiedad activa el sistema de control de errores en modo *Regex*. Contiene una expresión regular, o dos (en líneas separadas).

  Si se dan dos expresiones, la primera expresión se utiliza como un pre-filtro para encontrar expresiones que contengan identificadores de incidencias. La segunda expresión se utiliza para extraer los identificadores individuales desde los resultados de la primera expresión regular. Esto le permite utilizar una lista de identificadores de incidencias y expresiones en lenguaje natural si lo desea, por ejemplo puede arreglar varias incidencias e incluir una cadena como esta: “Este cambio resuelve las incidencias #23, #24 y #25”

  Si desea capturar identificadores de incidencias utilizados como se muestran anteriormente dentro de un mensaje de registro, puede utilizar las siguientes cadenas de expresiones regulares, que son las que se utilizan en el proyecto TortoiseSVN (traducidas al inglés): `[Ii]ncidencias?:?(\s*(,|y)?\s*#\d+)+` y `(\d+)`

  La primera expresión extrae “incidencias #23, #24 y #25” de entre el resto del mensaje de registro. La segunda expresión regular extrae los números decimales individuales desde la salida de la primera expresión regular, por lo que devolverá “23”, “24” y “25” para utilizarlos como identificadores de incidencias.

  Profundizando en la primera expresión regular un poco, debe empezar con la palabra “incidencia”, posiblemente empezando por mayúscula. Opcionalmente puede ir seguida de una “s” (más de una incidencia) y opcionalmente dos puntos. A esto le sigue uno o más grupos, cada uno empezando por cero o más espacios en blanco, una coma opcional o “y” y más espacios opcionales. Finalmente, hay un “#” obligatorio y un número decimal obligatorio.

  Si sólo se establece una expresión, entonces los identificadores individuales de incidencias deben concordar con los grupos de la expresión regular. Por ejemplo: `[Ii]ncidencia(?:s)? #?(\d+)` Este método se necesita por algunos gestores de incidencias, por ejemplo trac, pero la expresión regular es más dificil de construir. Le recomendamos que sólo utilice este método si la documentación de su gestor de incidencias se lo indica.

  Si no está familiarizado con las expresiones regulares, eche un vistazo a la introducción en *http://en.wikipedia.org/wiki/Regular_expression*, y a la documentación y al tutorial online en *http://www.regular-expressions.info/*.

Si se establecen las dos propiedades `bugtraq:message` y `bugtraq:logregex`, `logregex` tiene preferencia.

Sugerencia

¡Incluso si no tiene un programa de seguimiento de incidencias con un gancho pre-commit parseando sus mensajes de log, puede seguir utilizando ésto para convertir las incidencias mencionadas en su mensajes de registro en vínculos!

E incluso si no necesita los vínculos, los números de incidencias se muestran como una columna separada en el diálogo de registro, facilitándole la búsqueda de los cambios que se refieran a una incidencia en concreto.

Algunas propiedades `tsvn:` necesitan un valor `true/false` (verdadero/falso). TortoiseSVN también entiende `yes` (sí) como un sinónimo de `true` y `no` como un sinónimo de `false`.

Estableciendo las propiedades en las carpetas

Estas propiedades deben estar establecidas en carpetas para que el sistema funcione. Cuando confirma un fichero o una carpeta, se leen las propiedades de esa carpeta. Si no se encuentran

allí las propiedades, TortoiseSVN las buscará hacia arriba en el árbol de carpetas para encontrarlas, hasta que llega a una carpeta sin versionar, o se encuentra la raíz del árbol (por ejemplo, `C:\`). Si puede estar seguro de que cada usuario obtiene sólo desde por ejemplo `trunk/` y no desde alguna subcarpeta, entonces es suficiente establecer las propiedades en `trunk/`. Si no puede estar seguro, debería establecer las propiedades recursivamente en cada subcarpeta. Una propiedad establecida en una carpeta más profunda dentro de la jerarquía del proyecto tiene preferencia sobre las propiedades establecidas en niveles más altos (más cerca de `trunk/`).

Para las propiedades `tsvn:` *sólo* puede utilizar la casilla Recursivo para establecer la propiedad en todas las subcarpetas de la jerarquía, sin establecerla en todos los ficheros.

Sin información de seguimiento de incidencias desde el navegador de repositorios

Dado que la integración de seguimiento de incidencias depende del acceso a propiedades de Subversion, sólo verá los resultados cuando utilice una copia de trabajo obtenida. Acceder a las propiedades de forma remota es una operación lenta, por lo que no verá esta característica desde el navegador de repositorios.

La integración del seguimiento de incidencias no se restringe a TortoiseSVN; puede utilizarse con cualquier cliente de Subversion. Para más información, lea toda la *Especificación de integración con los seguimientos de incidencias* [http://tortoisesvn.googlecode.com/svn/trunk/doc/issuetrackers.txt] en el repositorio de fuentes de TortoiseSVN. (Sección 3, “¡TortoiseSVN es gratis!” le explica cómo acceder al repositorio).

4.28.2. Obteniendo información desde el gestor de incidencias

La sección anterior se refiere a cómo añadir información de incidencias en los mensajes de registros. Pero, ¿y si lo que necesita es obtener información desde el gestor de incidencias? El diálogo de confirmación tiene un interfaz COM que le permite la integración de un programa externo que pueda hablar con su gestor de incidencias. Típicamente, querrá interrogar al gestor para obtener una lista de incidencias abiertas que tenga asignadas, para que pueda seleccionar las incidencias de las que se va a hacer cargo en esta confirmación.

Any such interface is of course highly specific to your issue tracker system, so we cannot provide this part, and describing how to create such a program is beyond the scope of this manual. The interface definition and sample plugins in C# and C++/ATL can be obtained from the `contrib` folder in the *TortoiseSVN repository* [http://tortoisesvn.googlecode.com/svn/trunk/contrib/issue-tracker-plugins]. (Sección 3, “¡TortoiseSVN es gratis!” explains how to access the repository). A summary of the API is also given in Capítulo 6, *Interfase IBugtraqProvider* Another (working) example plugin in C# is *Gurtle* [http://code.google.com/p/gurtle/] which implements the required COM interface to interact with the *Google Code* [http://code.google.com/hosting/] issue tracker.

Por poner un ejemplo ilustrativo, supongamos que el administrador de su sistema le ha proporcionado un plugin de gestor de incidencias que ya ha instalado, y que ha preparado algunas de sus copias de trabajo para que utilicen el plugin en el diálogo de configuración de TortoiseSVN. Cuando abra la ventana de confirmación desde una copia de trabajo que tenga el plugin asignado, verá un nuevo botón en la parte superior del diálogo.

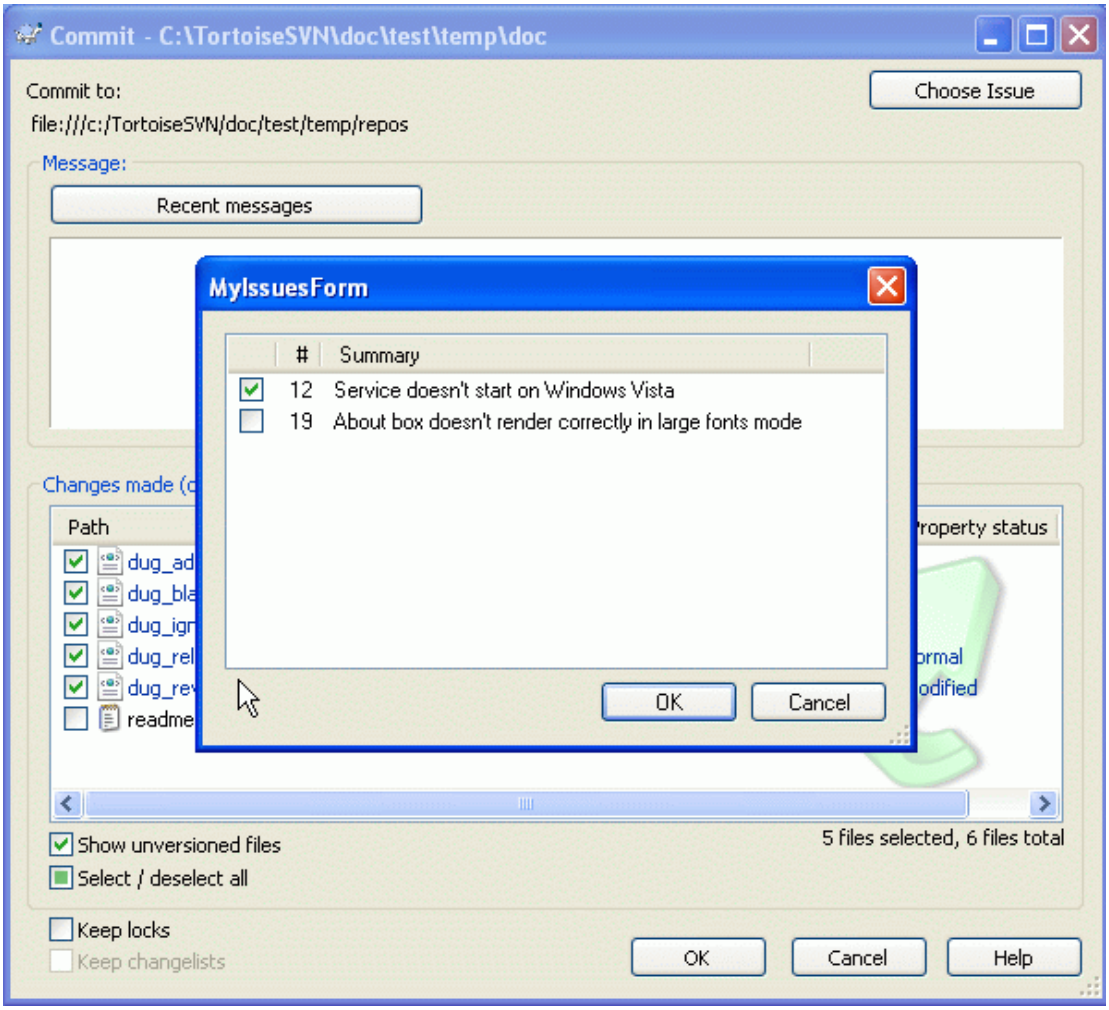


Figura 4.50. Diálogo de ejemplo de la interacción con el gestor de incidencias

En este ejemplo, puede seleccionar una o más incidencias abiertas. El plugin podrá entonces generar texto específicamente formateado para añadirlo a su mensaje de registro.

4.29. Integración con visores de repositorios basados en web

Hay varios visores de repositorios basados en web disponibles para utilizarlos con Subversion como *ViewVC* [http://www.viewvc.org/] y *WebSVN* [http://websvn.tigris.org/]. TortoiseSVN dispone de medios para enlazarse con estos visores.

Puede integrar un visor de repositorios de su elección en TortoiseSVN. Para hacerlo, debe definir algunas propiedades que definen la unión. Deben establecerse sobre las carpetas: (Sección 4.17, “Configuración del proyecto”)

webviewer:revision
: Establezca esta propiedad a la URL de su visor de repositorios para ver todos los cambios en una revisión en concreto. Debe ser una URI debidamente codificada y debe contener `%REVISION%`. `%REVISION%` se reemplaza con el número de revisión en cuestión. Esto permite que TortoiseSVN muestre una entrada en el menú contextual del diálogo de historial, Menú contextual → Ver la revisión en visor web

webviewer:pathrevision
: Establezca esta propiedad a la URL de su visor de repositorios para ver los cambios de un fichero específico en una revisión en concreto. Debe ser una URI debidamente codificada y debe contener

`%REVISION%` y `%PATH%`. `%PATH%` se reemplaza con la ruta relativa a la raíz del repositorio. Esto permite que TortoiseSVN muestre una entrada en el menú contextual del diálogo de historial, Menú contextual → Ver la revisión para la ruta en visor web. Por ejemplo, si hace click con el botón derecho en el panel inferior del diálogo de historial en una entrada de fichero llamada `/trunk/src/file`, la cadena `%PATH%` en la URL se reemplazará por `/trunk/src/file`.

También puede utilizar URLs relativas en vez de absolutas. Esto es útil en caso de que su control de incidencias esté en el mismo dominio/servidor que su repositorio. En caso de que el nombre de dominio cambie alguna vez, no tendrá que ajustar las propiedades `webviewer:revision` y `webviewer:pathrevision`. El formato es el mismo que el de la propiedad `bugtraq:url`. Vea Sección 4.28, “Integración con sistemas de control de errores / seguimiento de incidencias”.

Estableciendo las propiedades en las carpetas

Estas propiedades deben estar establecidas en carpetas para que el sistema funcione. Cuando confirma un fichero o una carpeta, se leen las propiedades de esa carpeta. Si no se encuentran allí las propiedades, TortoiseSVN las buscará hacia arriba en el árbol de carpetas para encontrarlas, hasta que llega a una carpeta sin versionar, o se encuentra la raíz del árbol (por ejemplo, `C:\`). Si puede estar seguro de que cada usuario obtiene sólo desde por ejemplo `trunk/` y no desde alguna subcarpeta, entonces es suficiente establecer las propiedades en `trunk/`. Si no puede estar seguro, debería establecer las propiedades recursivamente en cada subcarpeta. Una propiedad establecida en una carpeta más profunda dentro de la jerarquía del proyecto tiene preferencia sobre las propiedades establecidas en niveles más altos (más cerca de `trunk/`).

Para las propiedades `tsvn:` *sólo* puede utilizar la casilla Recursivo para establecer la propiedad en todas las subcarpetas de la jerarquía, sin establecerla en todos los ficheros.

Sin vínculos del visor de repositorios en el navegador de repositorios

Dado que la integración de visor de repositorio depende del acceso a propiedades de Subversion, sólo verá los resultados cuando utilice una copia de trabajo obtenida. Acceder a las propiedades de forma remota es una operación lenta, por lo que no verá esta característica desde el navegador de repositorios.

4.30. Configuración de TortoiseSVN

Para averiguar para qué sirven cada una de las diferentes opciones, deje el puntero del ratón un segundo sobre el cuadro de texto/casilla... y aparecerá un útil texto de ayuda.

4.30.1. Configuración general

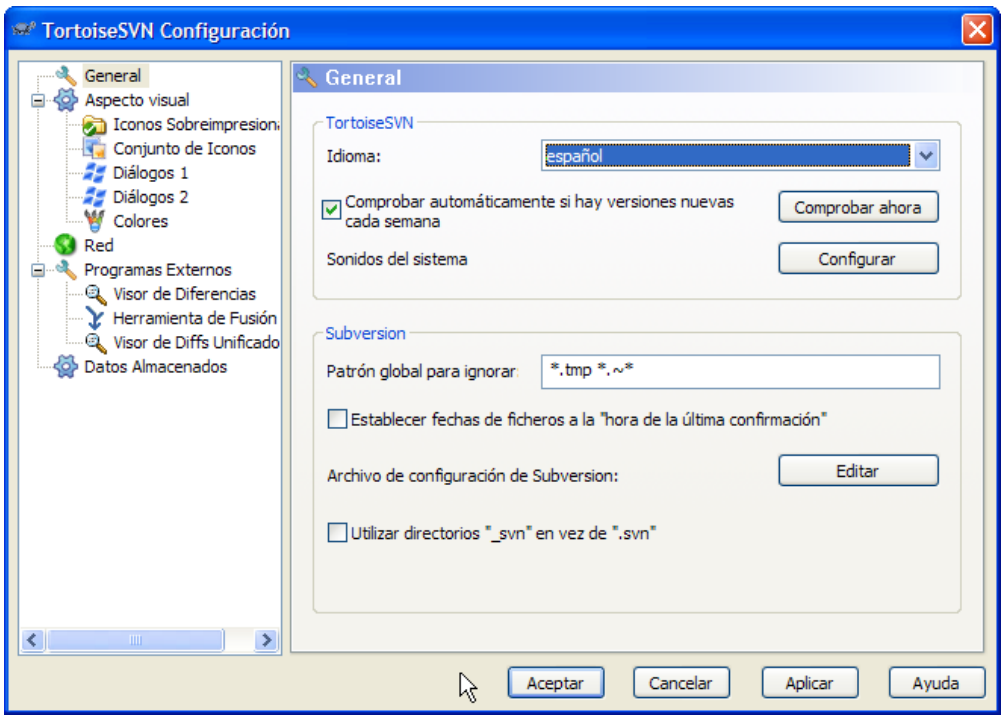


Figura 4.51. El diálogo de Configuración, página General

Este diálogo le permite especificar su idioma preferido, y las configuraciones específicas de Subversion.

Idioma
: Selecciona el idioma del interfaz de usuario. ¿Qué otra cosa esperabas?

Comprobar automáticamente si hay nuevas versiones todas las semanas
: Si se marca, TortoiseSVN contactará su sitio de descarga una vez a la semana para ver si hay disponible una versión más reciente del programa. Utilice el botón Comprobar ahora si desea una respuesta inmediata. No se descargará la versión más reciente; simplemente recibirá un diálogo de información diciéndole que hay una versión nueva disponible.

Sonidos del sistema
: TortoiseSVN tiene tres sonidos personalizados que se instalan por defecto.

  - Error
  - Información
  - Atención

  Puede seleccionar sonidos diferentes (o quitarlos completamente) utilizando el Panel de Control de Windows. Configurar es un acceso directo al Panel de Control.

Patrón global de ignorar

: Los patrones globales de ignorar se utilizan para evitar que aparezcan ficheros no versionados, por ejemplo, en el diálogo de confirmación. Los ficheros que concuerden con los patrones también se ignoran en las importaciones. Ignore ficheros o directorios escribiendo sus nombres o extensiones. Los patrones se separan por espacios, por ejemplo, `bin obj *.bak *.~?? *.jar *.[Tt]mp`. Estos patrones no deberían incluir ningún separador de rutas. Tenga en cuenta que no hay forma de diferenciar entre ficheros y directorios. Lea Sección 4.13.1, “Concordancia de patrones en las listas de ignorados” para obtener más información sobre la sintaxis de la concordancia de patrones.

Tenga en cuenta que los patrones de ignorar que especifique aquí también afectarán a otros clientes de Subversion que se ejecuten en su PC, incluyendo el cliente de línea de comandos.

Atención

Si utiliza el fichero de configuración de Subversion para establecer un patrón `global-ignores`, prevalecerá sobre las configuraciones que haga aquí. El fichero de configuración de Subversion se accede utilizando Editar como se describe a continuación.

Este patrón de ignorar afectará a todos sus proyectos. No se versiona, por lo que no afectará a otros usuarios. En el lado opuesto, puede utilizar también la propiedad versionada `svn:ignore` para excluir ficheros o directorios del control de versiones. Para más información, lea Sección 4.13, “Ignorando ficheros y directorios”.

Establecer las fechas de los ficheros a la “hora de la última confirmación”
: Esta opción le indica a TortoiseSVN que establezca las fechas de los ficheros al momento de la última confirmación cuando se hace una obtención o una actualización. Si no, TortoiseSVN utiliza la fecha actual. Si está desarrollando software generalmente es mejor que utilice la fecha actual porque los sistemas de compilación generalmente se fijan en las fechas de los ficheros para determinar qué ficheros necesitan ser compilados. Si utiliza “hora de la última confirmación” y revierte a una revisión anterior del fichero, su proyecto puede que no se compile como lo espera.

Fichero de configuración de Subversion
: Utilice Editar para editar el fichero de configuración de Subversion directamente. Algunas opciones no pueden ser modificadas directamente por TortoiseSVN, y necesitan establecerse aquí. Para más información sobre el fichero `config` de Subversion, lea *Área de configuración en tiempo de ejecución* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html]. La sección sobre *Configuración automática de propiedades* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.html#svn.advanced.props.auto] es especialmente interesante, y eso es se configura aquí. Tenga en cuenta que Subversion puede leer la información de configuración desde varios lugares, y necesita saber cuál tiene prioridad. Para obtener más información sobre esto, lea *Configuración y el registro de Windows* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html#svn.advanced.confarea.windows-registry].

Utiliza directorios `_svn` en vez de `.svn`
: VS.NET cuando se utiliza con proyectos web no puede manejar las carpetas `.svn` que Subversion utiliza para almacenar su información interna. Esto no es un error de Subversion. El error está en VS.NET y las extensiones de Frontpage que utiliza. Lea Sección 4.30.11, “Carpetas de trabajo de Subversion” para averiguar más sobre este problema.

  Si desea cambiar el comportamiento de Subversion y TortoiseSVN, puede utilizar esta casilla para establecer la variable de entorno que lo controla.

  Debería tener en cuenta que cambiar esta opción no convertirá automáticamente sus copias de trabajo para que usen ese nuevo directorio administrativo. Deberá hacerlo utilizando un script (vea nuestro FAQ) o simplemente obteniendo una copia de trabajo nueva.

4.30.1.1. Configuración del menú contextual

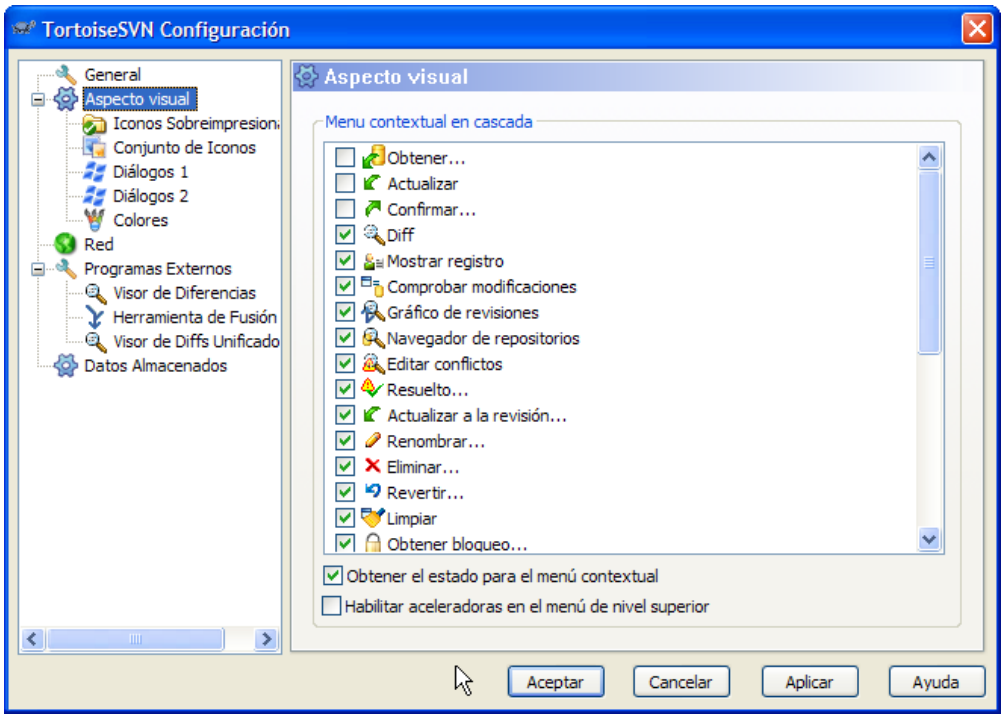


Figura 4.52. El diálogo Configuración, página de Menú contextual

Esta página le permite especificar qué entradas del menú contextual de TortoiseSVN se mostrarán en el menú contextual principal, y cuales aparecerán en el submenú TortoiseSVN. Por defecto, la mayoría de los ítems están desmarcados y aparecen en el submenú.

Hay un caso especial para Obtener bloqueo. Por supuesto puede promocionarlo al nivel superior utilizando la lista anterior, pero como la mayoría de ficheros no necesitan bloqueos esto molestaría. Sin embargo, un fichero con la propiedad `svn:needs-lock` necesita esta acción cada vez que se modifica, por lo que en ese caso sería útil tener la opción en el nivel superior. Activando esta casilla hace que cuando se selecciona un fichero que tenga establecida la propiedad `svn:needs-lock`, Obtener bloqueo se mostrará siempre en el nivel superior.

Si hay algunas rutas en su ordenador en las que simplemente no desea que aparezca el menú contextual de TortoiseSVN en absoluto, puede listarlas en la caja inferior.

4.30.1.2. Configuración de diálogos de TortoiseSVN 1

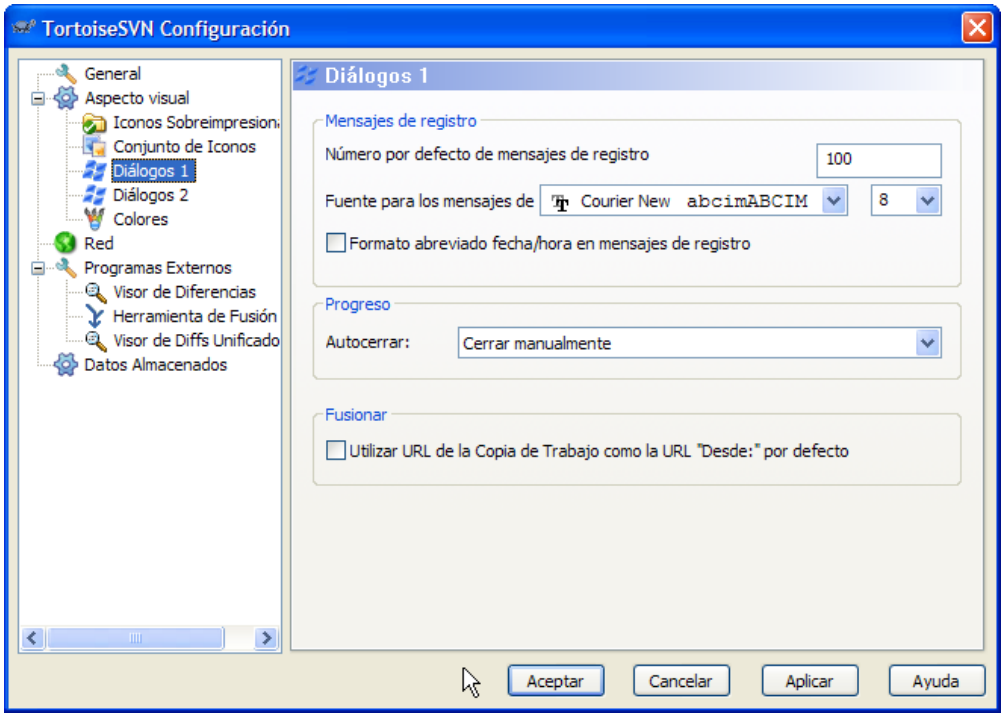


Figura 4.53. El diálogo Configuración, página de Diálogos 1

Este diálogo le permite configurar a su gusto algunos de los diálogos de TortoiseSVN.

Número por defecto de mensajes de registro
: Limita el número de mensajes de registro que TortoiseSVN obtiene la primera vez que seleccione TortoiseSVN → Mostrar registro Útil para conexiones lentas a servidores. Siempre puede utilizar Mostrar todos o Siguientes 100 para obtener más mensajes.

Fuente para los mensajes de registro
: Selecciona la fuente y el tamaño de la letra que se utiliza para mostrar el propio mensaje de registro en el panel del medio del diálogo Registro de Revisiones, y cuando se componen los mensajes de registro en el diálogo Confirmar.

Formato abreviado de fecha/hora en los mensajes de registro
: Si los mensajes estándar largos toman mucho espacio en la pantalla utilice la forma corta.

Doble-click en la lista de registro para comparar con revisión anterior
: Si se encuentra con que frecuentemente compara revisiones en el panel superior del diálogo de registro, puede utilizar esta opción para realizar esta acción en el doble click. Obtener las diferencias es a menudo un proceso largo, y muchos prefieren evitar la espera tras un doble click accidental, por lo que esta opción no está habilitada por defecto.

Diálogo de progreso
: TortoiseSVN puede cerrar automáticamente todos los diálogos de progreso cuando la acción se termina sin error. Esta configuración le permite seleccionar las condiciones para el cierre de diálogos. La configuración por defecto (recomendada) es Cerrar manualmente que le permite revisar todos los mensajes y comprobar qué ha pasado. Sin embargo, puede decidir que desea ignorar algunos tipos de mensajes y hacer que el cuadro de diálogo se cierre automáticamente si no hay cambios críticos.

  Auto-cerrar si no hay fusiones, adiciones o eliminaciones significa que el diálogo de progreso se cerrará si ha habido actualizaciones simples, pero si algún cambio del repositorio se ha fusionado con los suyos, o si se añadió o borró algún fichero, el diálogo seguirá abierto. También seguirá abierto si hubo algún conflicto o error durante la operación.

Auto-cerrar si no hay fusiones, adiciones o eliminaciones para operaciones locales significa que el diálogo de progreso se cerrará igual que para Auto-cerrar si no hay fusiones, adiciones o eliminaciones, pero sólo para operaciones locales como añadir ficheros o revertir cambios. Para las operaciones remotas el diálogo seguirá abierto.

Auto-cerrar si no hay conflictos relaja el criterio y cerrará el diálogo incluso si ha habido fusiones, adiciones o borrados. Sin embargo, si hay algún conflicto o error, el diálogo continuará abierto.

Auto-cerrar si no hay errores siempre cierra el diálogo, incluso si hay conflictos. La única condición que deja el diálogo abierto es una condición de error, que ocurre cuando Subversion no puede terminar una tarea. Por ejemplo, una actualización falla si el servidor es inaccesible, o una confirmación falla cuando la copia de trabajo no está actualizada.

Utilizar la papelera de reciclaje al revertir
: Cuando revierte las modificaciones locales, sus cambios se descartan. TortoiseSVN le ofrece una red de seguridad adicional al mandar el fichero modificado a la papelera de reciclaje antes de volver a la copia prístina. Si prefiere saltarse la papelera de reciclaje, desmarque esta opción.

Utilizar la URL de la copia de trabajo como la URL “Desde:” por defecto
: En el diálogo de fusión, el comportamiento por defecto es que la URL Desde: se mantenga entre fusiones. Sin embargo, algunas personas quieren realizar fusiones desde diferentes puntos en su jerarquía, y les resulta más cómodo empezar con la URl de la copia de trabajo actual. Esto puede luego cambiarse para referirse a una ruta paralela en otra rama.

Ruta de obtención por defecto
: Puede especificar la ruta por defecto para las obtenciones. Si mantiene todas sus obtenciones en un mismo lugar, es útil tener la unidad y la carpeta pre-rellenados para que así sólo tenga que añadir el nombre de la nueva carpeta al final.

URL de obtención por defecto
: También puede especificar la URL por defecto para las obtenciones. Si obtiene a menudo subprojectos de un projecto muy grante, puede ser útil tener la URL pre-rellenada de forma que sólo tenga que añadir el nombre del subprojecto al final.

4.30.1.3. Configuración de diálogos de TortoiseSVN 2

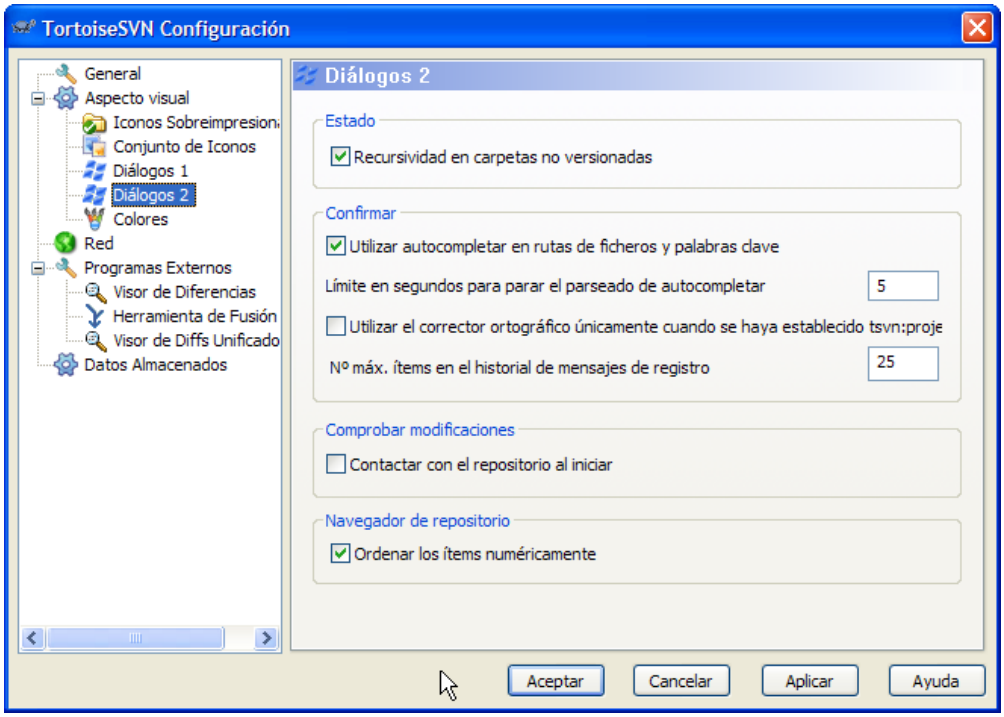


Figura 4.54. El diálogo Configuración, página de Diálogos 2

Recursividad en carpetas no versionadas
: Si se marca esta casilla (por defecto está marcada), entonces siempre que se muestre el estado de una carpeta sin versionar en los diálogos de Añadir, Confirmar o Comprobar Modificaciones, se muestran también sus ficheros y carpetas hijos. Si desmarca esta casilla, sólo se muestra la carpeta padre sin versionar, lo que reduce el desorden en esos diálogos. En ese caso, si selecciona una carpeta sin versionar para Añadir, se añadirá recursivamente.

Utilizar autocompletar en rutas de ficheros y palabras clave
: El diálogo confirmar incluye una ayuda para parsear la lista de nombres de ficheros que se van a confirmar. Cuando teclee las primeras 3 letras de un ítem en la lista, aparecerá la caja de autocompletar, y puede pulsar Enter para completar el nombre del fichero. Seleccione la caja para habilitar esta opción.

Tiempo límite en segundos para detener el parseado de autocompletar
: El parseador de autocompletar puede ser bastante lento si hay muchos ficheros grandes que comprobar. Este tiempo límite evita que el diálogo de confirmación se congele durante mucho tiempo. Si se está perdiendo información de autocompletar importante, puede extender el tiempo límite.

Utilizar el corrector ortográfico únicamente cuando se haya establecido `tsvn:projectlanguage`
: Si no desea utilizar el corrector ortográfico para todas las confirmaciones, marque esta casilla. Aun así, el corrector ortográfico se activará cuando las propiedades del proyecto así lo indiquen.

Número máximo de ítems para mantener en el historial de mensajes de registro.
: Cuando escribe un mensaje de registro en el diálogo de confirmación, TortoiseSVN lo almacena para poder reutilizarlo más tarde. Por defecto, almacen los últimos 25 mensajes de registro para cada repositorio, pero puede personalizar el número aquí. Si tiene muchos repositorios diferentes, puede querer reducirlo para evitar llenar su registro.

  Tenga en cuenta que esta configuración sólo se aplica a los mensajes que escriba en este ordenador. No tiene nada que ver con la caché de registro.

Re-abrir los diálogos de confirmación y rama/etiqueta después de una confirmación fallida
: Cuando una confirmación falla por alguna razón (la copia de trabajo necesita actualizarse, un gancho pre-confirmación rechaza la confirmación, error de red, etc), puede seleccionar esta opción para mantener el diálogo de confirmación abierto y listo para intentarlo de nuevo. Sin embargo, debe tener en cuenta de que estoy puede provocar problemas. Si el fallo significa que debe actualizar su copia de trabajo, y esa actualización conlleva conflictos, deberá resolver esos conflictos primero.

Seleccionar ítems automáticamente
: El comportamiento normal en el diálogo de confirmación es que todos los ítems (versionados) modificados se seleccionen automáticamente para la confirmación. Si prefiere empezar sin nada seleccionado y seleccionar los ficheros para la confirmación manualmente, desmarque esta casilla.

Contactar con el repositorio al iniciar
: El diálogo Comprobar Modificaciones comprueba la copia de trabajo por defecto, y sólo contacta con el repositorio cuando pulse el botón Comprobar repositorio. Si siempre quiere comprobar el repositorios, puede utilizar esta opción para que esa acción siempre ocurra automáticamente.

Mostrar el diálogo Bloquear antes de bloquear ficheros
: Cuando selecciona uno o más ficheros y utiliza TortoiseSVN → Bloquear para obtener un bloqueo en esos ficheros, en algunos projectos es costumbre escribir un mensaje de bloqueo explicando por qué ha bloqueado esos ficheros. Si no utiliza mensajes de bloqueo, puede desmarcar esta casilla para saltar este diálogo y bloquear los ficheros inmediatamente.

  Si utiliza el comando bloquear en una carpeta, siempre se le mostrará el diálogo de bloqueo ya que también le ofrece la posibilidad de seleccionar qué ficheros bloquear.

  Si su proyecto utiliza la propiedad `tsvn:lockmsgminsize`, verá el diálogo de bloqueo sin tener en cuenta esta opción ya que el proyecto *necesita* mensajes de bloqueo.

4.30.1.4. Configuración de colores de TortoiseSVN

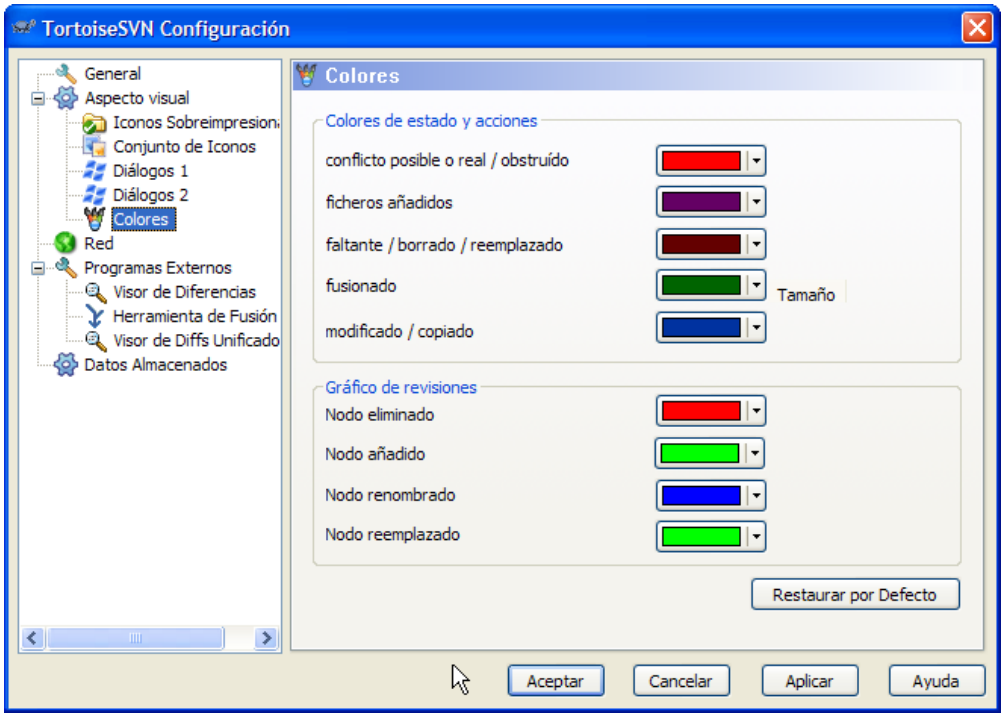


Figura 4.55. El diálogo Configuración, página de Colores

Este diálogo le permite configurar a su gusto los colores de los textos utilizados en los diálogos de TortoiseSVN.

Conflicto posible o real / obstruído
: Ha ocurrido un conflicto durante la actualización, o puede ocurrir durante la fusión. La actualización está obstruida por una carpeta o un fichero sin versionar que ya existe en su copia de trabajo, y tiene el mismo nombre que uno versionado.

  Este color se utiliza también para los mensajes de error en los diálogos de progreso.

Ficheros añadidos
: Ítems añadidos al repositorio.

Faltante / borrado / reemplazado
: Ítems borrados del repositorio, faltantes en la copia de trabajo, o borrados de la copia de trabajo y reemplazados con otro fichero del mismo nombre.

Fusionado
: Cambios del repositorio que se han fusionado satisfactoriamente con su copia de trabajo sin crear ningún conflicto.

Modificado / copiado
: Añadido con historia, o rutas copiadas en el repositorio. También se utiliza en el diálogo de registro para las entradas que incluyen ítems copiados.

Nodo eliminado
: Un ítem que ha sido eliminado del repositorio.

Nodo añadido
: Un ítem que ha sido añadido al repositorio, mediante una operación añadir, copiar o mover.

Nodo renombrado
: Un ítem que ha sido renombrado dentro del repositorio.

Nodo reemplazado
: El ítem original ha sido borrado y un nuevo ítem con el mismo nombre le reemplaza.

4.30.2. Configuración del gráfico de revisión

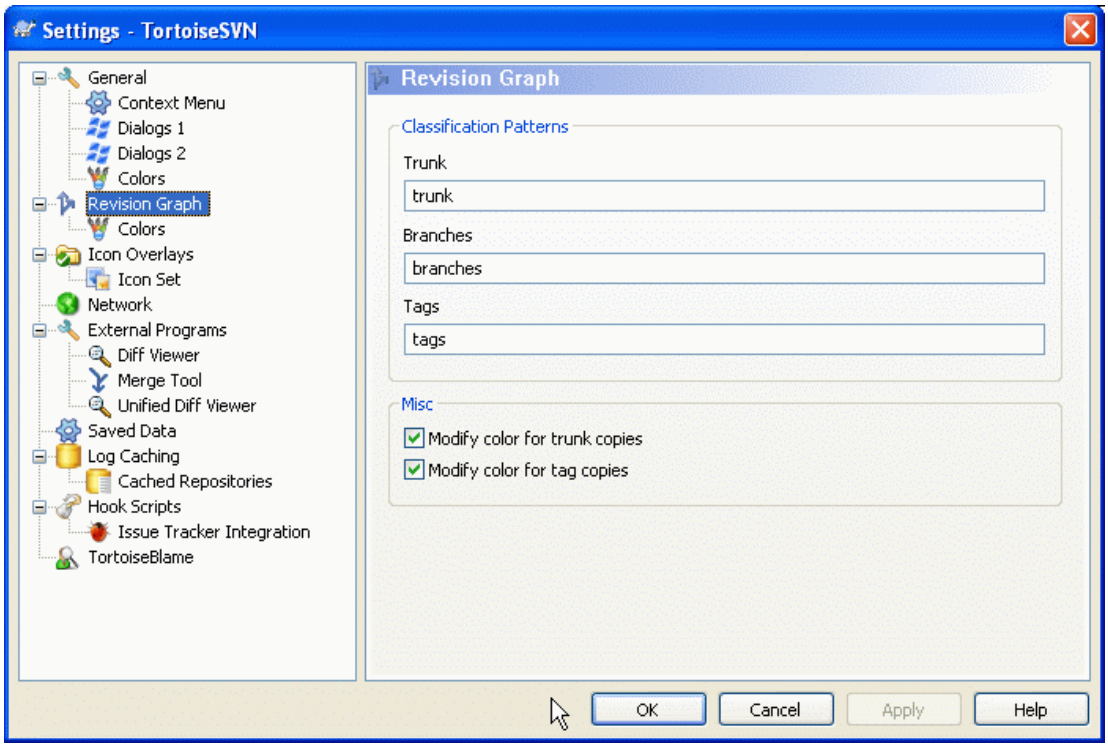


Figura 4.56. El diálogo de Configuración, página Gráfico de revisión

Patrones de clasificación
: El gráfico de revisiones intenta mostrar una imagen más clara de la estructura de su repositorio distinguiendo entre tronco, ramas y etiquetas. Dado que no existe dicha clasificación interna en Subversion, esta información se extrae desde los nombres de las rutas. Las configuraciones por defecto asumen que utiliza los nombres convencionales en inglés tal y como se sugieren en la documentación de Subversion (trunk, tags y branches), pero por supuesto puede cambiarlo.

  Especifique los patrones que se utilizarán para reconocer estos patrones en las tres cajas de texto propuestas. Los patrones se comprobarán sin tener en cuenta mayúsculas y minúsculas, pero debe especificarlas en minúsculas. Los caracteres comodín `*` y `?` funcionarán como siempre, y puede utilizar `;` para separar patrones múltiples. No incluya ningún espacio en blanco extra, porque se incluirá en la especifcación de concordancia.

Modificar colores
: Los colores se utilizan en el gráfico de revisiones para indicar el tipo de nodo, es decir, si un nodo se añadió, eliminó, renombró. Para ayudar a distinguir las clasificaciones de los nodos, puede permitir que el gráfico de revisiones fusione colores para dar una indicación de tanto el tipo de nodo y la clasificación. Si la casilla está marcada, se utiliza la fusión. Si la casilla se desmarca, el color se utiliza sólo para indicar el tipo de nodo. Utilice el diálogo de selección de colores para especificar los colores específicos usados.

4.30.2.1. Colores del gráfico de revisión

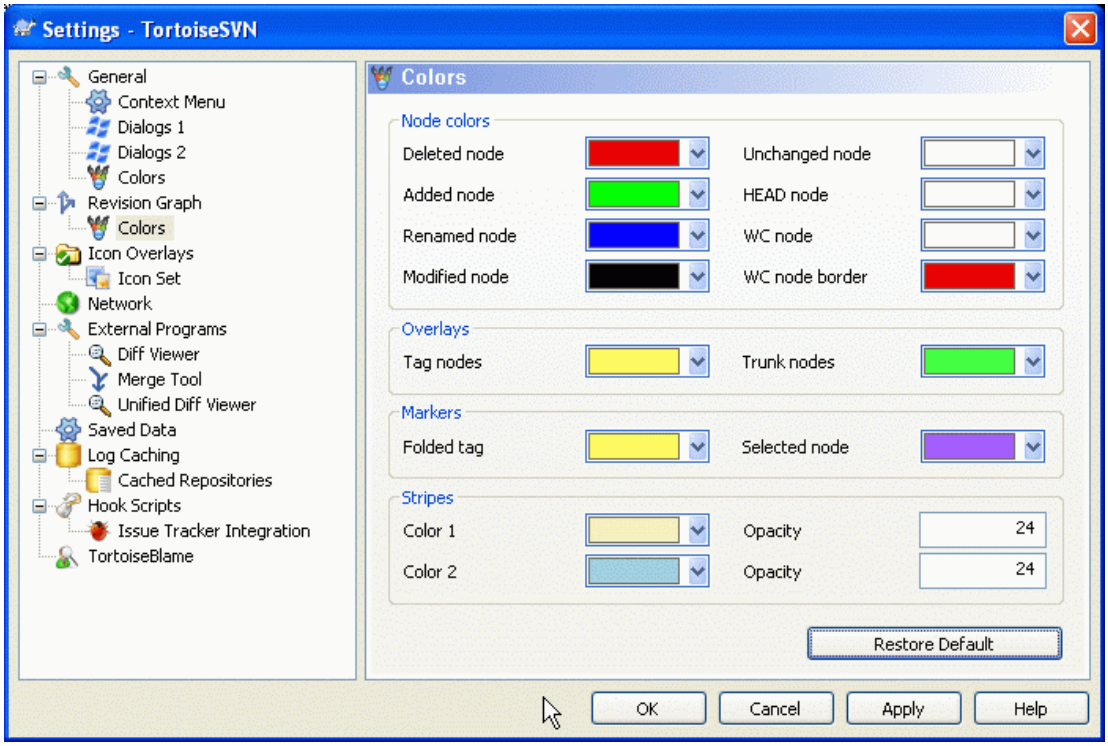


Figura 4.57. El diálogo Configuración, página Colores del gráfico de revisión

Esta página le permite configurar los colores utilizados. Tenga en cuenta que el color especificado aquí es el color sólido. la mayoría de los nodos se colorean utilizando una fusión del color del tipo de nodo, el color de fondo y opcionalmente el color de clasificación.

Nodo eliminado
: Ítems que se han eliminado y no se han copiado a ningún otro lugar en la misma revisión.

Nodo añadido
: Ítems añadidos nuevos, o copiados (añadidos con historia).

Nodo renombrado
: Ítems eliminados de un lugar y añadidos en otro en la misma revisión.

Nodo modificado
: Modificaciones simples sin ningún añadir o eliminar.

Nodo sin cambios
: Puede utilizarse para mostrar la revisión utilizada como origen de una copia, incluso cuando no hay ningún cambio (sobre el ítem cuyo gráfico se va a mostrar) en esa revisión.

Nodo HEAD
: HEAD actual en el repositorio.

Nodo WC
: If you opt to show an extra node for your modified working copy, attached to its last-commit revision on the graph, use this color.

Borde del nodo WC
: Si opta por mostrar si la copia de trabajo está modificada, utilice este borde de color en el nodo WC cuando se encuentren modificaciones.

Nodos etiqueta
: Los nodos clasificados como etiquetas pueden fundirse con este color.

Nodos tronco
: Los nodos clasificados como tronco pueden fundirse con este color.

Marcadores de etiquetas plegadas
: Si utiliza plegado de etiquetas para ahorrar espacio, las etiquetas se marcan en la copia origen utilizando un bloque de este color.

Marcadores del nodo seleccionado
: Cuando hace click con el botón izquierdo en un nodo para seleccionarlo, el marcador utilizado para indicar la selección es un bloque de este color.

Bandas
: Estos colores se utilizan cuando el gráfico se divide en sub-árboles y el fondo se colorea en bandas alternadas para ayudar a distinguir los distintos árboles.

4.30.3. Configuración de los iconos sobreimpresionados

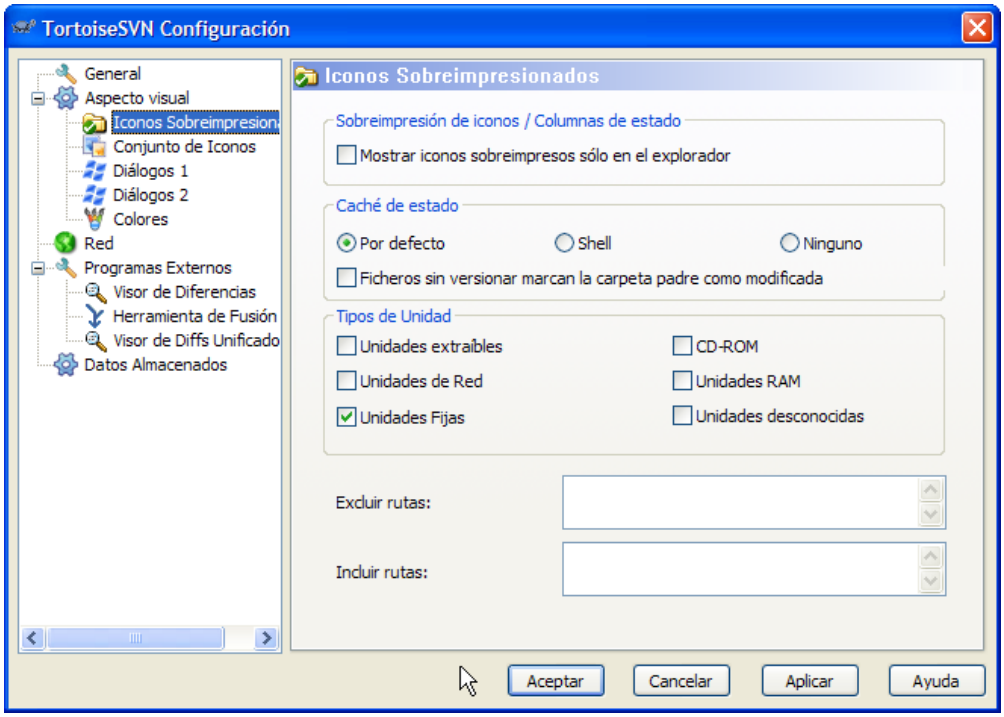


Figura 4.58. El diálogo Configuración, página de Sobreimpresión de iconos

Esta página le permite seleccionarlos ítems para los que TortoiseSVN mostrará iconos sobreimpresionados.

Por defecto, los iconos sobreimpresionados y los menús contextuales aparecerán tanto en todos los diálogos abrir/guardar como en el Explorador de Windows. Si desea que aparezcan *únicamente* en el Explorador de Windows, marque Mostrar iconos sobreimpresionados y menú contextual sólo en el explorador box.

Los ítems ignorados y no versionados normalmente no tienen una sobreimpresión. Si desea mostrar una sobreimpresión en estos casos, simplemente marque las casillas.

También puede elegir marcar las carpetas como modificadas si contienen ítems sin versionar. Esto puede ser útil para recordarle que ha creado nuevos ficheros que aún no están versionados. Esta opción sólo está disponible cuando utiliza la opción de la caché de estados *por defecto* (más abajo).

Dado que lleva bastante tiempo obtener el estado de una copia de trabajo, TortoiseSVN utiliza una caché para almacenar el estado de forma que el explorador no se vea demasiado atosigado cuando se muestren las sobreimpresiones. Puede elegir qué tipo de caché deberá utilizar TortoiseSVN de acuerdo a sus sistema y al tamaño de las copias de trabajo aquí:

Por defecto
: Cachea toda la información de estado en un proceso separado (`TSVNCache.exe`). Ese proceso observa todas las unidades en busca de cambios y obtiene el estado de nuevo si se modifican ficheros dentro de una copia de trabajo. El proceso se ejecuta con la menor prioridad posible para que el rendimiento de otros programas no se resientan por su culpa. Eso también significa que la información de estado no se actualiza en *tiempo real* sino que puede llevar unos pocos segundos hasta que las sobreimpresiones cambien.

  Ventaja: las sobreimpresiones le muestran el estado de forma recursiva, es decir, si un fichero muy dentro de su copia de trabajo se modifica, todas las carpetas padre hasta la raíz de su copia de trabajo mostrarán la sobreimpresión de modificación. Y dado que el proceso puede enviar notificaciones al shell, las sobreimpresiones del árbol de la izquierda normalmente también cambiarán.

  Desventaja: el proceso se ejecuta constantemente, incluso si no está trabajando en sus proyectos. Además utiliza alrededor de 10-50 MB de RAM dependiendo del número y tamaño de sus copias de trabajo.

Shell
: El cacheo se realiza directamente dentro de la dll de la extensión del shell, pero sólo para la carpeta actualmente visible. Cada vez que navega a otra carpeta, se obtiene de nuevo la información de estado.

  Ventaja: necesita únicamente muy poca memoria (alrededor de 1 MB de RAM) y puede mostrar el estado en *tiempo real*.

  Desventaja: Dado que sólo se hace caché de una carpeta, las sobreimpresiones no muestran el estado recursivamente. Para copias de trabajo grandes, mostrar una carpeta en el explorador puede llevar más tiempo que con la caché por defecto. Además, la columna tipo-mime no está disponible.

Ninguno
: Con esta configuración, TortoiseSVN no obtiene ningún estado en el Explorador. Por esa causa, los ficheros no tendrán sobreimpresiones y las carpetas sólo tendrán una sobreimpresión 'normal' si están versionadas. No se mostrará ninguna otra sobreimpresión, y tampoco estarán disponibles las columnas extra.

  Ventaja: no utiliza memoria adicional en absoluto y no ralentiza de ninguna forma el Explorador mientras se navega.

  Desventaja: La información de estado de los ficheros y carpetas no se muestra en el Explorador. Para ver si sus copias de trabajo están modificadas, tendrá que utilizar el diálogo “Comprobar modificaciones”.

El siguiente grupo le permite seleccionar qué tipos de almacenamiento deberían mostrar sobreimpresiones. Por defecto, sólo los discos duros están seleccionados. Incluso puede deshabilitar todos los iconos sobreimpresionados, pero ¿qué gracia tendría eso?

Las unidades de red pueden ser muy lentas, así que por defecto no se muestran los iconos para las copias de trabajo que se encuentren en unidades de red.

Las unidades USB Flash parecen ser un caso especial en el que el tipo de unidad lo identifica el propio dispositivo. Algunas aparecen como discos duros, y otros como discos extraíbles.

Excluir Rutas se usa para decirle a TortoiseSVN en qué rutas *no* debe mostrar iconos sobreimpresionados ni columnas de estado. Esto es útil si tiene algunas copias de trabajo muy grandes que contienen

únicamente librerías que no cambian en absoluto y por tanto no necesitan las sobreimpresiones. Por ejemplo:

`f:\desarrollo\SVN\Subversion` desabilitará las sobreimpresiones *sólo* en esa ruta concreta. Todavía podrá ver las sobreimpresiones en todos los ficheros y carpetas dentro de esa carpeta.

`f:\desarrollo\SVN\Subversion*` deshabilitará las sobreimpresiones en *todos* los ficheros y carpetas cuya ruta empiece por `f:\desarrollo\SVN\Subversion`. Esto significa que no verá sobreimpresiones para ningún fichero y carpetas debajo de esa ruta.

Lo mismo se aplica a Incluir rutas. Excepto que para esas rutas las sobreimpresiones se muestran incluso si están deshabilitadas para ese tipo de unidad en concreto, o por una ruta de exclusión especificada más arriba.

Los usuarios a veces preguntan cómo funcionan estas tres configuraciones, y la respuesta definitiva es:

```
si (la ruta está en la lista de inclusión)
  mostrar sobreimpresiones
si (la ruta está en un tipo de unidad permitido) Y (la ruta no está en la lista d
  mostrar sobreimpresiones
```

La lista de incluídos *siempre* hace que las sobreimpresiones se muestren. En otro caso, las sobreimpresiones se muestran para todos los tipos de unidad marcados *excepto* si la ruta se ha excluido.

TSVNCache.exe también utiliza estas rutas para restringir su escaneo. Si desea que sólo mire en determinadas carpetas, deshabilite todos los tipos de unidades e incluya sólo las carpetas que específicamente quiere que sean escaneadas.

Excluir unidades SUBST

A menudo es conveniente utilizar una unidad SUBST para acceder a sus copias de trabajo, por ejemplo utilizando el comando

```
subst T: C:\TortoiseSVN\trunk\doc
```

Sin embargo esto puede ocasionar que las sobreimpresiones no se actualicen, ya que TSVNCache sólo recibirá una notificación cuando cambie un fichero, y eso es normalmente para la ruta original. Esto significa que sus sobreimpresiones sobre rutas `subst` puede que nunca se actualicen.

Una forma fácil de evitar esto es excluir de las sobreimpresiones la ruta original, de forma que las sobreimpresiones aparezcan ahora en la ruta `subst`.

A veces excluirá areas que contienen copias de trabajo, lo que evita que TSVNCache las escanee y monitorice los cambios, pero aún querrá una indicación visual de que dichas carpetas están versionadas. La casilla Mostrar carpetas excluidas como normales le permite hacer esto. Con esta opción, las carpetas versionadas en cualquier área excluída (tipo de unidad no marcada, o excluída explícitamente) se mostrarán como normal y como actualizadas, con una marca verde. Esto le recuerda que está viendo una copia de trabajo, incluso aunque las sobreimpresiones de las carpetas no sean correctas. Los ficheros no tendrán ninguna sobreimpresión. Tenga en cuenta que los menús contextuales seguirán funcionando, incluso aunque las sobreimpresiones no se muestren.

Como una excepción especial a esto, las unidades `A:` y `B:` nunca se consideran para la opción Mostrar carpetas excluidas como normales. El motivo es que Windows se ve obligado a mirar en la unidad, lo que puede resultar en un retraso de varios segundos cuando se inicia el Explorador, incluso si su PC no tiene una unidad de disquete.

4.30.3.1. Selección del conjunto de iconos

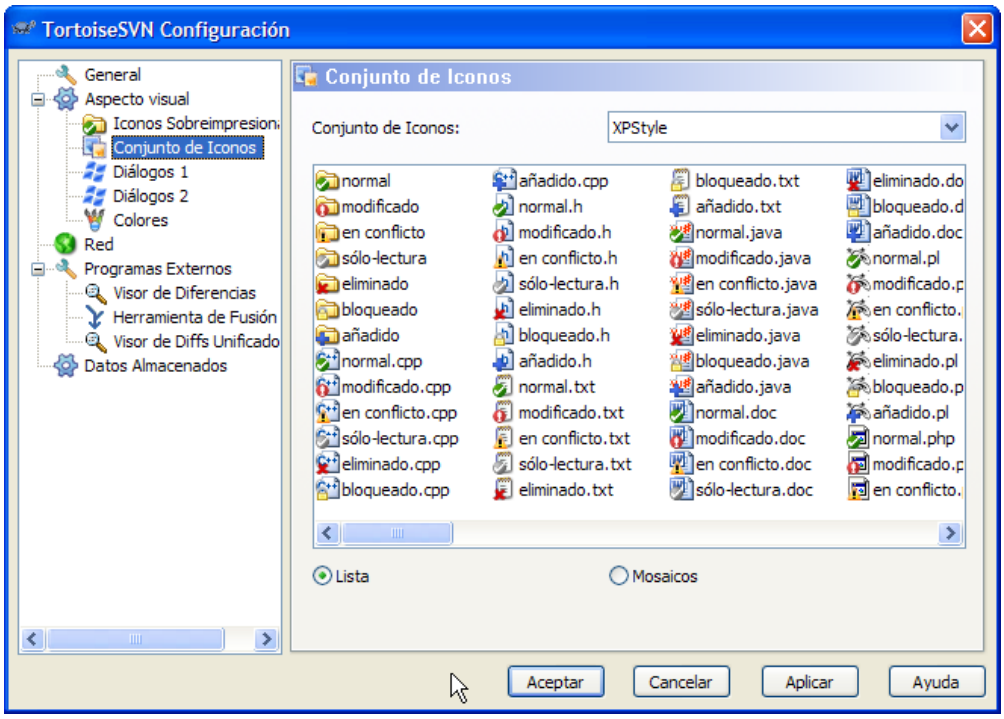


Figura 4.59. El diálogo Configuración, página de Conjunto de iconos

También puede cambiar el conjunto de iconos de sobreimpresión al que más le guste. Tenga en cuenta que si cambia el conjunto de sobreimpresiones, puede tener que reiniciar su ordenador para que estos cambios surtan efecto.

4.30.4. Configuración de red

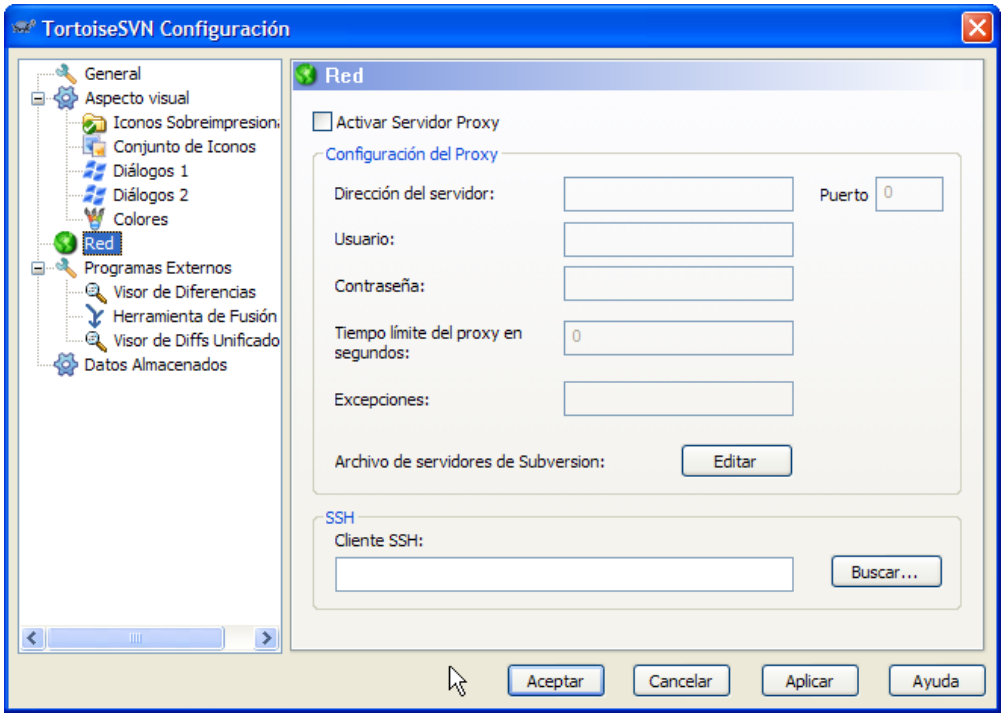


Figura 4.60. El diálogo Configuración, página de Red

Aquí puede configurar su servidor proxy, si necesita uno para atravesar el cortafuegos de su compañía.

Si necesita preparar configuraciones de proxy por cada repositorio, necesitará utilizar el fichero `servers` de Subversion para configurarlo. Utilice Editar para llegar allí directamente. Consulte *Área de configuración en tiempo de ejecución* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html] para ver los detalles sobre cómo utilizar este fichero.

También puede especificar qué programa debe utilizar TortoiseSVN para establecer una conexión segura a un repositorio svn+ssh. Le recomendamos que utilice TortoisePlink.exe. Ésta es una versión del popular programa Plink, y se incluye con TortoiseSVN, pero está compilado como una aplicación sin ventanas, por lo que no verá una ventana DOS molestando cada vez que se autentifica.

Debe especificar la ruta completa al ejecutable. En el caso de TortoisePlink.exe, esta es la carpeta bin estándar de TortoiseSVN. Utilice el botón Explorar para ayudarle a encontrarla. Tenga en cuenta que si la ruta contiene espacios, deberá rodearlos con comillas, por ejemplo

```
"C:\Archivos de programa\TortoiseSVN\bin\TortoisePlink.exe"
```

Un efecto colateral de no tener una ventana es que no hay sitio para que vayan los mensajes de error, por lo que si la autentificación falla símplemente obtendrá un mensaje diciendo algo como “Unable to write to standard output” (“No se puede escribir en la salida estándar”). Por esta razón le recomendamos que primero pruebe su configuración con el Plink estándar. Cuando todo funcione, puede utilizar TortoisePlink con exáctamente los mismos parámetros.

TortoisePlink no tiene ninguna documentación propia porque es una variante menor de Plink. Averigue los parámetros de línea de comando desde el *sitio web de PuTTY* [http://www.chiark.greenend.org.uk/~sgtatham/putty/]

Para evitar ser preguntado por una contraseña repetidamente, puede considerar utilizar una herramienta de caché de contraseñas como Pageant. También está disponible desde el sitio web de PuTTY.

Finalmente, configurar SSH en el servidor y los clientes en un proceso no-trivial que está fuera del alcance de este fichero de ayuda. Sin embargo, puede encontrar una guía en el FAQ de TortoiseSVN mostrado como *Subversion/TortoiseSVN SSH How-To* [http://tortoisesvn.net/ssh_howto].

4.30.5. Configuración de programas externos

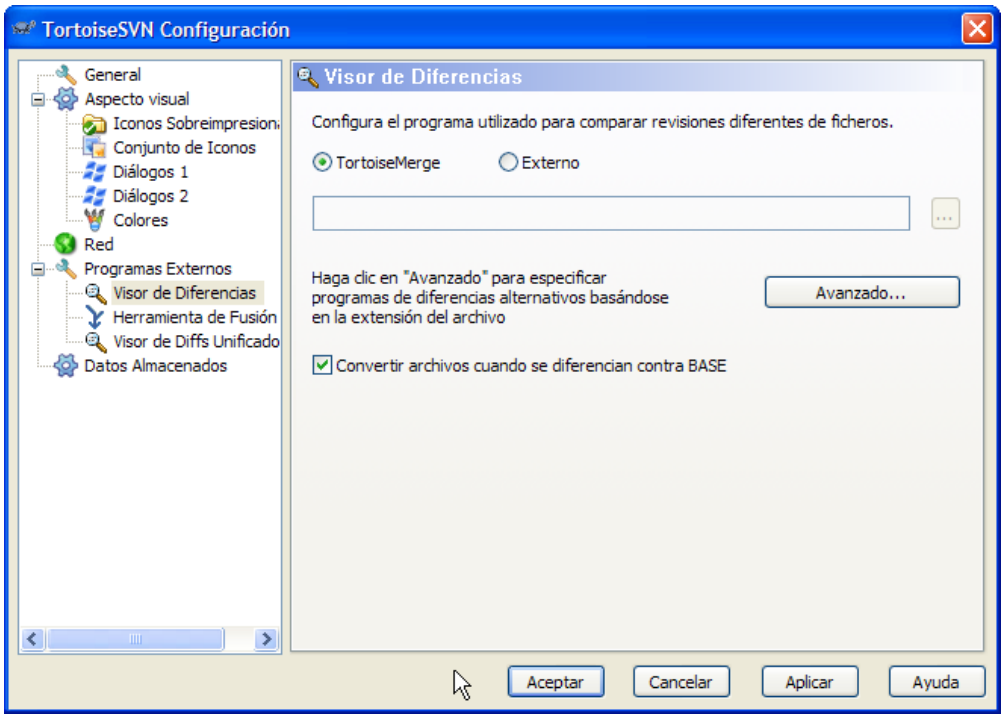


Figura 4.61. El diálogo Configuración, página de Visor de diferencias

Aquí puede definir sus propios programas de diferencias/fusión que TortoiseSVN debería utilizar. La configuración básica es utilizar TortoiseMerge que se instala junto con TortoiseSVN.

En Sección 4.10.5, “Herramientas externas de diferencias/fusión” encontrará una lista de algunos de los programas externos de diferencias / fusión que la gente está utilizando con TortoiseSVN.

4.30.5.1. Visor de diferencias

Se puede utilizar un programa externo de fierencias para comparar diferentes revisiones de ficheros. El programa externo necesitará obtener los nombres de los ficheros de la línea de comando, junto con otras opciones de la línea de comandos. TortoiseSVN usa sustitución de parámetros con prefijo %. Cuando se encuentra uno de estos se sustituirá por el valor apropiado. El orden de los parámetros dependerá del programa de Diferencias que utilice.

%base
: El fichero original sin sus cambios

%bname
: La ventana de título para el fichero base

%mine
: Su propio fichero, con sus cambios

%yname
: El título de la ventana para su fichero

Los títulos de las ventanas no son nombres de ficheros puros. TortoiseSVN lo trata como un nombre para mostrar y crea los nombres según eso. Por ejemplo, si está haciendo diferencias entre un fichero en la revisión 123 y un fichero en su copia de trabajo, los nombres pueden ser `nombre-de-fichero : revision 123` y `nombre-de-fichero : copia de trabajo`

Por ejemplo, con ExamDiff Pro:

```
C:\Ruta-A\ExamDiff.exe %base %mine --left_display_name:%bname
    --right_display_name:%yname
```

o con KDiff3:

```
C:\Ruta-A\kdiff3.exe %base %mine --L1 %bname --L2 %yname
```

o con WinMerge:

```
C:\Ruta-A\WinMerge.exe -e -ub -dl %bname -dr %yname %base %mine
```

o con Araxis:

```
C:\Ruta-A\compare.exe /max /wait /title1:%bname /title2:%yname
    %base %mine
```

Si utiliza la propiedad `svn:keywords` para expandir palabras clave, y en particular la `revision` de un fichero, entonces puede haber una diferencia entre ficheros que se debe únicamente al valor actual de esta palabra clave. También puede ocurrir que si utiliza `svn:eol-style = native` el fichero BASE tenga finales de líneas formados únicamente por `LF` mientras que sus ficheros tendrán finales de líneas `CR-LF`. TSVN normalmente oculta esas diferencias automáticamente ya que primero parsea el fichero BASE para expandir las palabras clave y los finales de líneas antes de realizar la operación de diferenciar. Sin embargo, esto puede tomar bastante tiempo con ficheros grandes. Si se desmarca Convertir ficheros cuando se diferencian contra BASE, TortoiseSVN se saltará el preprocesado de los ficheros.

También puede especificar una herramienta distinta de diferencias para utilizarla en las propiedades de Subversion. Dado que estas tienden a ser cadenas cortas de texto simple, puede que desee utilizar un visor más compacto.

Si ha configurado una herramienta de diferencias distinta, puede acceder a TortoiseMerge *y* a la herramienta de terceros desde los menús contextuales. Menú contextual → Diff utiliza la herramienta de diferencias primaria, y **Mayúsculas**+ Menú contextual → Diff utiliza la herramienta de diferencias secundaria.

4.30.5.2. Herramienta de fusión

Un programa de fusión externo que se utiliza para resolver ficheros en conflicto. La sustitución de parámetros se utiliza de la misma forma que el Programa de Diferencias.

%base
: El fichero original sin ningún cambio, ni suyo ni de otros

%bname
: La ventana de título para el fichero base

%mine
: Su propio fichero, con sus cambios

%yname
: El título de la ventana para su fichero

%theirs
: El fichero tal cual estaba en el repositorio

%tname
: El título de la ventana del fichero en el repositorio

%merged
El fichero en conflicto, el resultado de la operación de fusión

%mname
El título de la ventana para el fichero fusionado

Por ejemplo, con Perforce Merge:

```
C:\Ruta-A\P4Merge.exe %base %theirs %mine %merged
```

o con KDiff3:

```
C:\Ruta-A\kdiff3.exe %base %mine %theirs -o %merged
    --L1 %bname --L2 %yname --L3 %tname
```

o con Araxis:

```
C:\Ruta-A\compare.exe /max /wait /3 /title1:%tname /title2:%bname
    /title3:%yname %theirs %base %mine %merged /a2
```

o con WinMerge (2.8 o superior):

```
C:\Ruta-A\WinMerge.exe %merged
```

4.30.5.3. Configuración avanzada de diferencias / fusiones

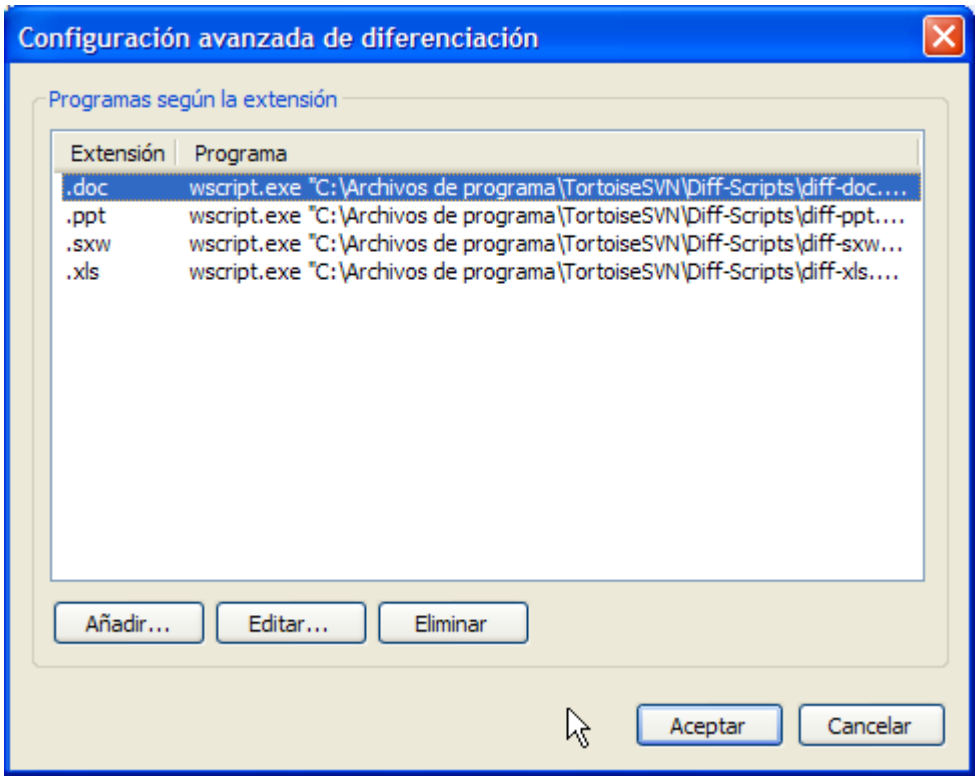


Figura 4.62. El diálogo Configuración, diálogo de Diferencias/Fusión avanzadas

En las configuraciones avanzadas, puede definir un programa de diferencias y fusión diferente por cada extensión de fichero. Por ejemplo, podría asociar Photoshop como el programa de “Diferencias” para ficheros `.jpg` :-) También puede asociar la propiedad `svn:mime-type` con un programa de diferencias o fusión.

Para asociar utilizando una extensión de fichero, necesita especificar la extensión. Utilice `.bmp` para describir ficheros de bitmaps de Windows. Para asociar utilizando la propiedad `svn:mime-type`, especifique el tipo MIME, incluyendo una barra, por ejemplo `text/xml`.

4.30.5.4. Visor para ficheros de diff unificado

Un programa visor de ficheros diff unificado (ficheros de parche). No se necesitan parámetros. La opción Por defecto es para buscar un programa asociado a los ficheros `.diff`, y si no, para ficheros `.txt`. Si no tiene un visor para los ficheros `.diff`, seguramente obtendrá el Bloc de notas.

El programa Bloc de notas original de Windows no se comporta bien con los ficheros que no tienen terminaciones de líneas CR-LF estándar. Dado que la mayoría de ficheros diff unificados tienen terminaciones de líneas LF puras, no se ven bien en el Bloc de Notas. Sin embargo, puede descargar un reemplazo del Bloc de Notas *Notepad2* [http://www.flos-freeware.ch/notepad2.html] que no sólo enseña las terminaciones de líneas correctamente, sino que también añade código de colores a las líneas añadidas y eliminadas.

4.30.6. Datos de configuración almacenados

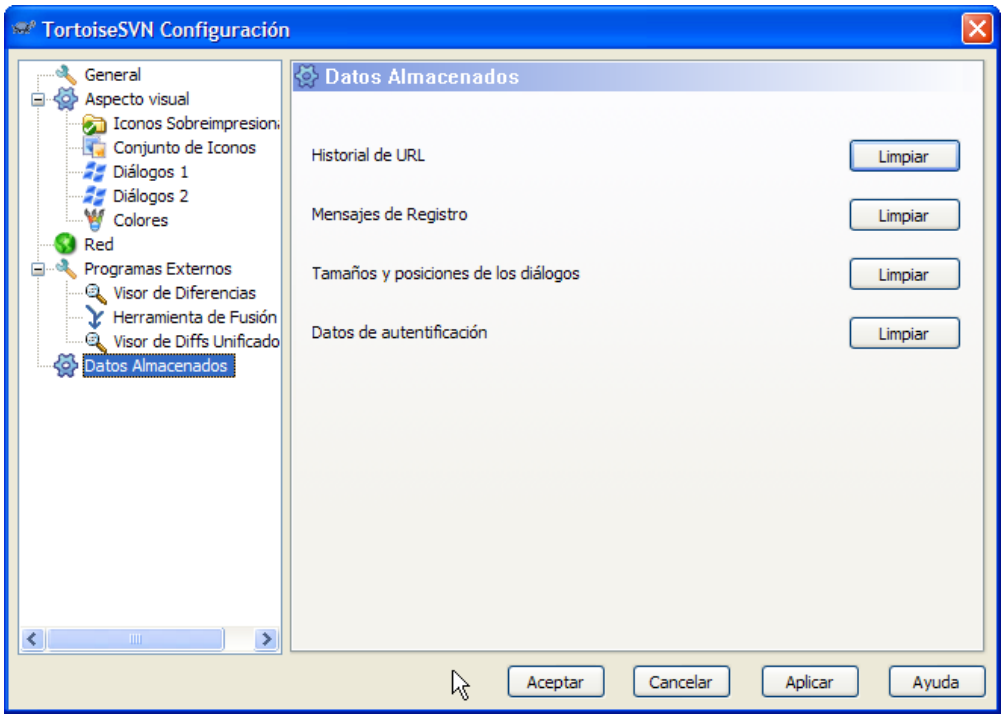


Figura 4.63. El diálogo Configuración, página de Datos almacenados

Por su comodidad, TortoiseSVN almacena muchas de las opciones que utiliza, y recuerda dónde ha estado últimamente. Si desea limpiar esa caché de datos, puede hacerlo aquí.

Historia de URL
: Siempre que obtiene una copia de trabajo, fusione cambios o utilice el navegador de repositorios, TortoiseSVN guarda un registro de las últimas URLs utilizadas y las ofrece en una caja desplegable. A veces esa lista se llena de URLs desfasadas por lo que puede ser util limpiarla periódicamente.

  Si quiere eliminar un único ítem de uno de los cuadro de texto desplegables puede hacerlo allí mismo. Simplemente haga click en la flecha para desplegar el cuadro, mueva el ratón sobre el ítem que quiere eliminar y pulse Mayúsculas+Supr.

Mensajes de registro (diálogo de entrada)
: TortoiseSVN almacena los mensajes de registro de confirmación que introduce. Estos se almacenan por cada repositorio, por lo que si accede a muchos repositorios la lista puede crecer bastante.

Mensajes de registro (diálogo Mostrar registro)
: TortoiseSVN almacena en caché los mensajes de registro obtenidos por el diálogo Mostrar registro para ahorrar tiempo la próxima vez que muestre el registro. Si alguien edita un mensaje de registro que ya estuviera en su caché, no verá el cambio hasta que limpie la caché. La caché de mensajes de registro se habilita en la pestaña Caché de registros.

Tamaños y posiciones de los diálogos
: Muchos diálogos almacenan el tamaño y la posición en pantalla que tenían la última vez.

Datos de autentificación
: Cuando se autentifica con un servidor de Subversion, el usuario y la contraseña se cachean localmente para que no tenga que seguir introduciéndolos. Puede querer limpiar esto por razones de seguridad, o porque quiere acceder al repositorio bajo un nombre de usuario diferente ... ¿sabe John que está utilizando su PC?

  Si desea eliminar los datos de autentificación de un único servidor en concreto, lea Sección 4.1.5, “Autentificación” para obtener instrucciones sobre cómo encontrar los datos en caché.

Registro de acciones
: TortoiseSVN almacena un registro de todo lo que ha escrito en sus diálogos de progreso. Esto puede ser útil si, por ejemplo, desea comprobar qué ocurrió en un comando de actualización reciente.

  El fichero de registro tiene un tamaño límite y cuando crece demasiado el contenido más antiguo se descarta. Por defecto se guardan 4000 líneas, pero puede personalizar este número.

  Desde aquí puede ver el contenido del fichero de registro, y también limpiarlo.

4.30.7. Caché de registro

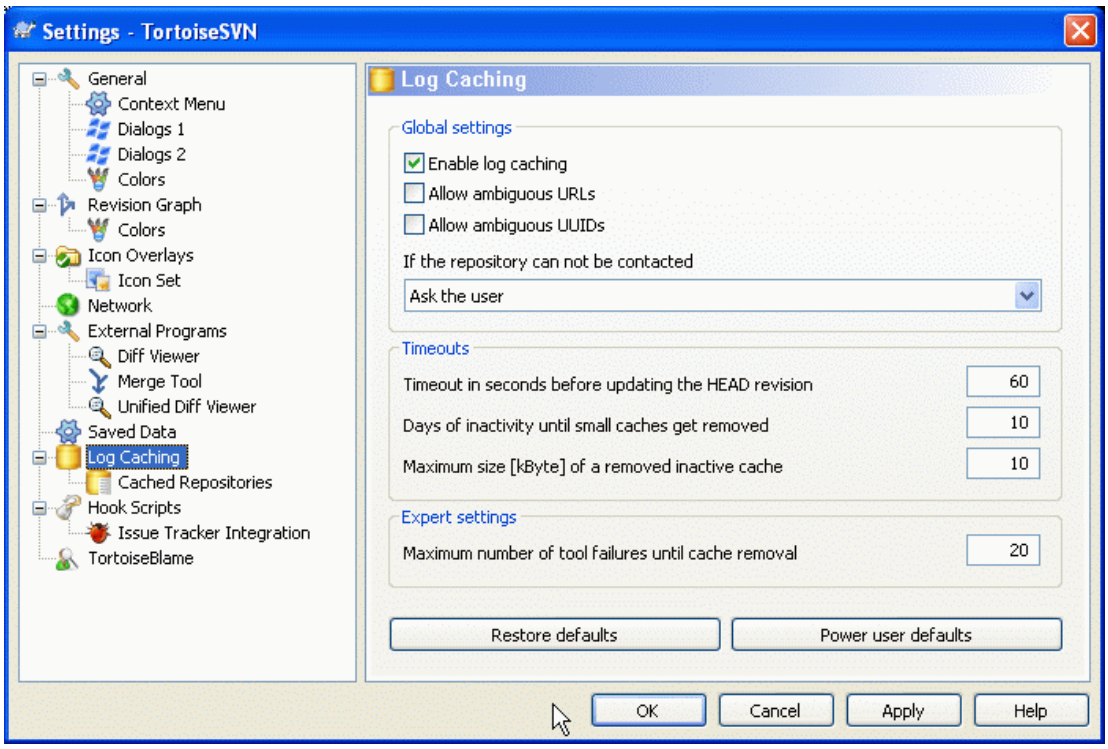


Figura 4.64. El diálogo Configuración, página Caché de registro

Este diálogo le permite configurar la característica de caché de registro de TortoiseSVN, que retiene una copia local de los mensajes de registro y rutas cambiadas para evitar descargas desde el servidor

que llevan tiempo. Al utilizar la caché de registro acelerará dramáticamente el diálogo de registro y el diálogo de revisiones. Otra característica útil es que los mensajes de registro serán accesibles cuando esté desconectado.

Habilitar la caché de registro
: Habilita la caché de registro siempre que se pidan datos de registro. Si se marca, los datos se recuperarán de la caché cuando estén disponibles allí, y cualquier mensaje que no esté en la caché se recuperará del servidor y se añadirá a la caché.

  Si la caché se deshabilita, los datos siempre se recuperarán directamente del servidor y no se almacenarán localmente.

Permitir URLs ambiguas
: Ocasionalmente puede tener que conectarse a un servidor que utiliza la misma URL para todos los repositorios. Las versiones antiguas de `svnbridge` harían esto. Si necesita acceder a este tipo de repositorios necesitará marcar esta opción. Si no lo necesita, deje esta casilla desmarcada para mejorar el rendimiento.

Permitir UUIDs ambiguas
: Algunos proveedores de hospedaje dan a todos sus repositorios el mismo UUID. Puede que incluso le haya pasado si ha copiado una carpeta de repositorio para crear otro. Esto es una mala idea por toda clase de motivos - un UUID debería ser *único*. Sin embargo, la caché de registro todavía podrá trabajar en esta situación si marca esta casilla. Si no lo necesita, déjela desmarcada para mejorar el rendimiento.

Si no se puede contactar con el repositorio
: Si está trabajando sin conexión, o si el servidor del repositorio está caído, la caché de registro aún puede utilizarse para suministrar mensajes de registro que estén ya almacenados en la caché. Por supuesto la caché puede no estar actualizada, por lo que hay opciones que le permiten seleccionar cuándo debería utilizarse esta característica.

  Cuando los datos de registro se toman de la caché sin contactar con el servidor, el diálogo que utiliza dichos mensajes mostrará el estado desconectado en su barra de título.

Tiempo límite antes de actualizar la revisión HEAD
: Cuando invoca al diálogo de registro normalmente querrá contactar con el servidor para comprobar si hay mensajes de registro nuevos. Si el tiempo límite establecido aquí es distinto de cero, sólo se contactará con el servidor cuando el tiempo límite haya pasado desde el último contacto. Esto puede reducir los viajes al servidor si abre el diálogo de registro frecuentemente y el servidor es lento, pero los datos mostrados pueden no estar completamente actualizados. Si desea utilizar esta característica le sugerimos un valor de compromiso de 300 (5 minutos).

Días de inactividad hasta que las cachés pequeñas se eliminan
: Si navega por muchos repositorios acumulará un montón de cachés de registro. Si no las utiliza de forma activa, las cachés no crecerán mucho, por lo que TortoiseSVN por defecto las purgará tras un periodo de tiempo. Utilize este ítem para controlar la purga de la caché.

Tamaño máximo de las cachés inactivas eliminadas
: Las cachés más grandes son más costosas de readquirir, por lo que TortoiseSVN sólo purga cachés pequeñas. Ajuste el límite con este valor.

Máximo número de fallos de herramienta hasta la eliminación de la caché
: Ocasionalmente algo puede ir mal con la caché y provocar un fallo. Si esto ocurre la caché normalmente se elimina automáticamente para evitar la recurrencia del problema. Si utiliza la versión noctura y menos estable puede optar por mantener la caché de todas formas.

4.30.7.1. Repositorios cacheados

En esta página puede ver una lista de repositorios que están almacenados en la caché local, y el espacio utilizado para la caché. Si selecciona uno de los repositorios podrá utilizar los botones inferiores.

Pulse en Actualizar para refrescar completamente la caché y rellenar cualquier hueco. Para un repositorio grande esto puede llevar mucho tiempo, pero es útil si va a desconectarse y quiere tener la mejor caché disponible.

Pulse en Exportar para exportar la caché completa como un conjunto de ficheros CSV. Esto puede ser útil si desea procesar los datos de registro utilizando un programa externo, aunque es útil principalmente para los desarrolladores.

Pulse en Eliminar para eliminar todos los datos de la caché de los repositorios seleccionados. Esto no deshabilita la caché para el repositorio, por lo que la próxima vez que pida datos de registro se creará una nueva caché.

4.30.7.2. estadísticas de la caché de registro

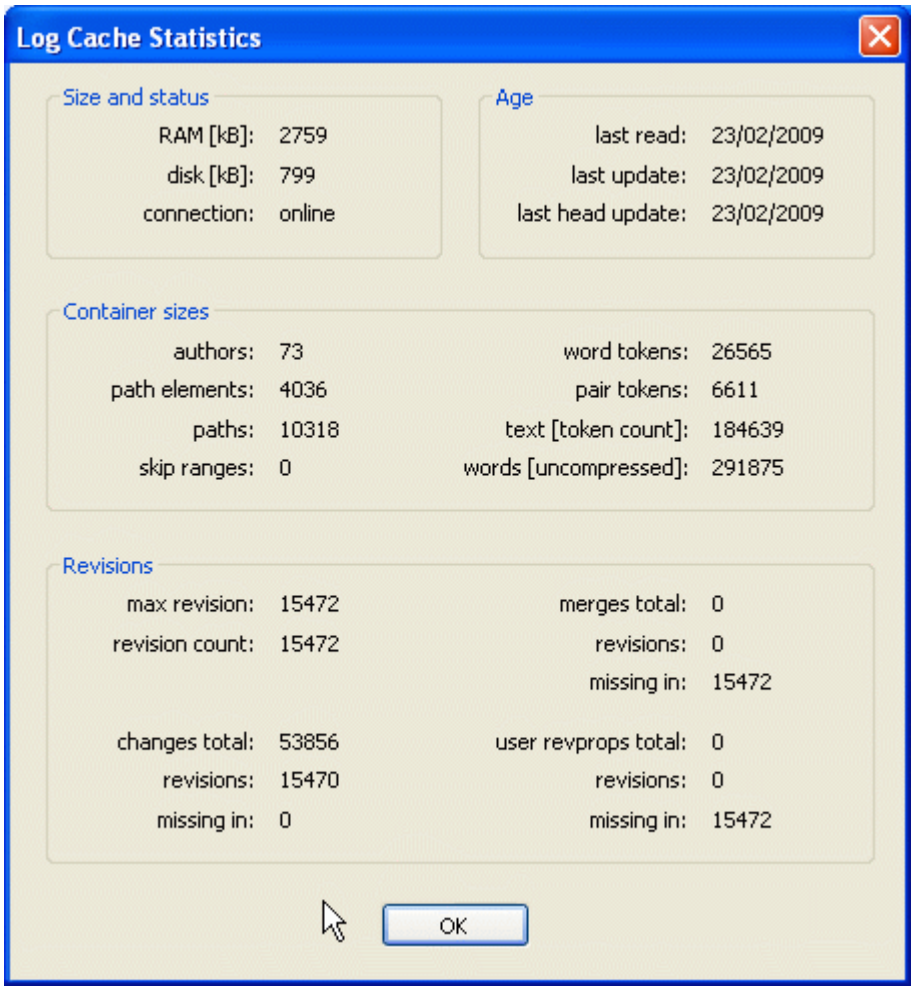


Figura 4.65. El diálogo Configuración, página Estadísticas de la caché de registro

Pulse en el botón Detalles para ver estadísticas detalladas de una caché en concreto. Muchos de los campos mostrados aquí son de interés principalmente para los desarrolladores de TortoiseSVN, así que no se describen todos en detalle aquí.

RAM
: La cantidad de memoria necesaria para dar servicio a esta caché.

Disco
: El espacio de disco utilizado para la caché. Los datos están comprimidos, así que el uso de disco generalmente es bastante modesto.

Conexión
: Muestra si el repositorio estaba disponible la última vez que se utilizó la caché.

Última actualización
: La última fecha y hora en la que se modificó el contenido de la caché.

Última actualización head
: La última vez que se pidió la revisión HEAD al servidor.

Autores
: El número de diferentes autores con mensajes almacenados en la caché.

Rutas
: El número de rutas mostradas, como las vería utilizando `svn log -v`.

Saltar rangos
: El número de rangos de revisiones que no se han obtenido, simplemente por no han sido pedidos. Esta es una medida del número de huecos en la caché.

Revisión máx
: El número de revisión más alto almacenado en la caché.

Cuenta de revisiones
: El número de revisiones almacenados en la caché. Esta es otra medida de la completitud de la caché.

4.30.8. Scripts gancho del lado del cliente

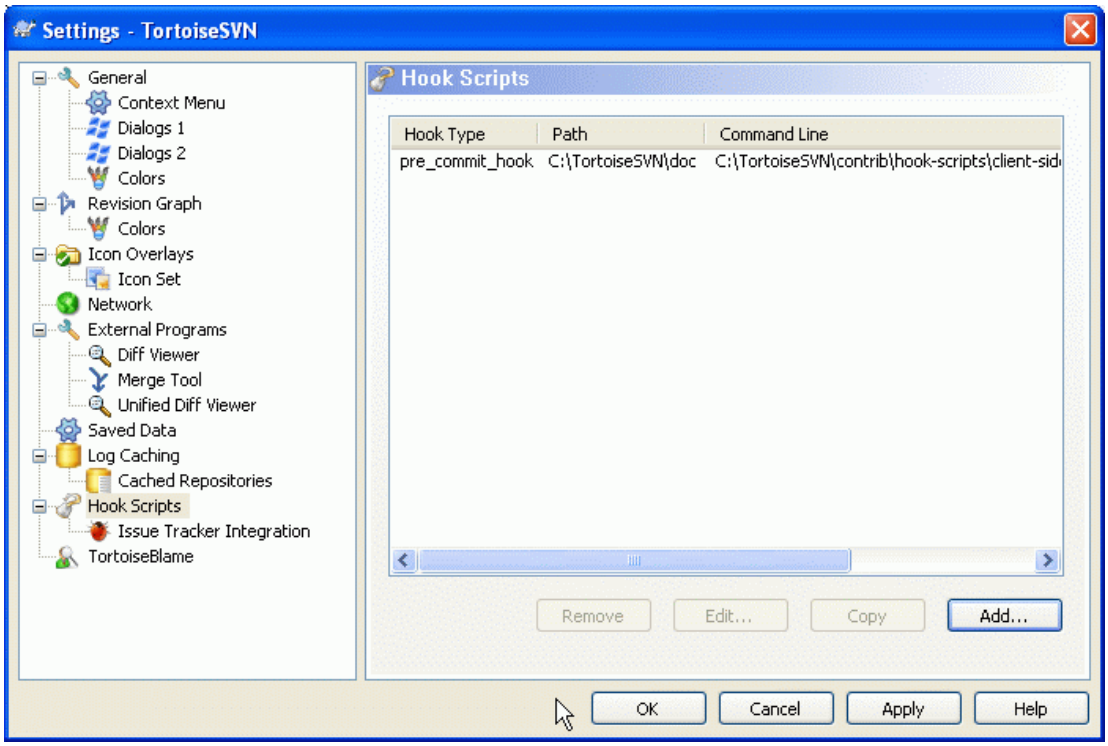


Figura 4.66. El diálogo Configuración, página de scripts gancho

Este diálogo le permite establecer scripts gancho que se ejecutarán automáticamente cuando se realicen ciertas acciones de Subversion. Al contrario que los scripts gancho explicados en Sección 3.3, “Scripts gancho en el lado del servidor”, estos scripts se ejecutan localmente en el cliente.

Una aplicación para este tipo de ganchos podría ser una llamada a un programa como `SubWCRev.exe` para actualizar los números de versión tras una confirmación, y quizás para provocar una recompilación.

Por varios motivos de seguridad e implementación, los scripts gancho se definen localmente en la máquina, en vez de ser propiedades de proyecto. Así define lo que ocurre, sin importar lo que otros confirmen en el repositorio. Por supuesto siempre puede elegir llamar a un script que ya esté bajo control de versiones.

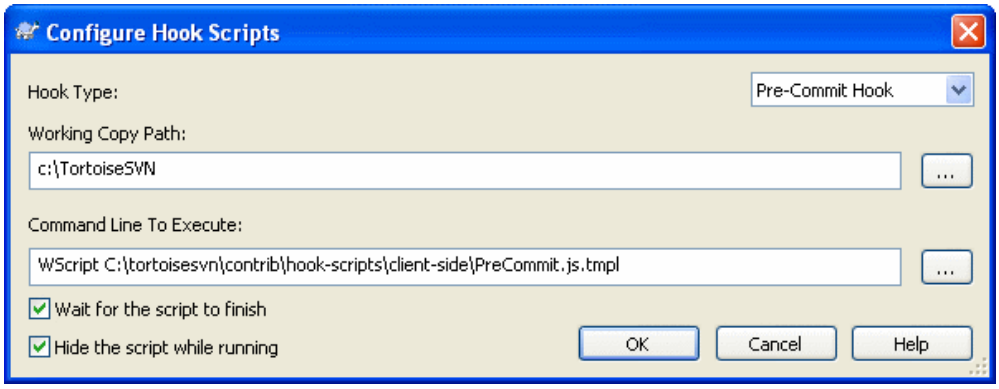


Figura 4.67. El diálogo Configuración, configurar scripts gancho

Para añadir un nuevo script gancho, simplemente pulse Añadir y rellene los detalles.

Actualmente hay seis tipos de scripts ganchos disponibles

Inicio-confirmación
: Llamado antes de que se muestre el diálogo de confirmación. Puede querer utilizarlo si el gancho modifica un fichero versionado y afecta a la lista de ficheros que necesitan ser confirmados y/o al mensaje de confirmación. Sin embargo, debería tener en cuenta que dado que el gancho se llama en una fase temprana, la lista completa de objetos seleccionados para confirmar no está disponible.

Pre-confirmación
: Llamado después de que el usuario hace click en el botón Aceptar en el diálogo de confirmación, y antes de que comience el proceso de confirmación. Este gancho tiene una lista exacta de lo que se va a confirmar.

Post-confirmación
: Llamado antes de que termine la confirmación (sea satisfactoria o no).

Inicio-actualización
: Llamado antes de que se muestre el diálogo actualizar-a-la-revisión.

Pre-actualización
: Llamado antes de que comience la actualización de Subversion.

Post-actualización
: Llamado después de que termine la actualización (sea satisfactoria o no).

Un gancho se define para una ruta de copia de trabajo en particular. Sólo necesita especificar la ruta de más alto nivel; si realiza una operación en una subcarpeta, TortoiseSVN automáticamente buscará hacia arriba una ruta que concuerde.

Después debe especificar la línea de comandos a ejecutar, comenzando con la ruta del script gancho o el ejecutable. Esto puede ser un fichero batch, un fichero ejecutable o cualquier otro fichero que tenga una asociación de fichero de Windows válida, como por ejemplo un script Perl.

La línea de comandos incluye diversos parámetros que se rellenarán por TortoiseSVN. Los parámetros que se pasan dependen de qué gancho se ejecuta. Cada gancho tiene sus propios parámetros que se pasan en el siguiente orden:

Inicio-confirmación
: `PATHMESSAGEFILECWD`

Pre-confirmación
: `PATHDEPTHMESSAGEFILECWD`

Post-confirmación
: `PATHDEPTHMESSAGEFILEREVISIONERRORCWD`

Inicio-actualización
: `PATHCWD`

Pre-actualización
: `PATHDEPTHREVISIONCWD`

Post-actualización
: `PATHDEPTHREVISIONERRORCWD`

El significado de cada una de estos parámetros se describe aquí:

PATH
: Una ruta a un fichero temporal que contiene todas las rutas para la operación que se ha iniciado. Cada ruta aparece en una línea distinta en el fichero temporal.

DEPTH
: La profundidad a la que se realiza la confirmación o actualización.

  Los valores posibles son:

  -2
  : `svn_depth_unknown`

  -1
  : `svn_depth_exclude`

  0
  : `svn_depth_empty`

  1
  : `svn_depth_files`

  2
  : `svn_depth_immediates`

  3
  : `svn_depth_infinity`

MESSAGEFILE
: Ruta a un fichero que contiene el mensaje de registro de la confirmación. El fichero contiene el texto con la codificación UTF-8. Tras la ejecución con éxito del gancho start-commit, el mensaje de registro se vuelve a leer, dándole al gancho una oportunidad para modificarlo.

REVISION
: La revisión del repositorio a la que se debe actualizar o después de que se completa una confirmación.

ERROR
: Ruta a un fichero que contiene el mensaje de error. Si no hubo error, el fichero estará vacío.

CWD
: El directorio de trabajo actual desde el que se ejecuta el script. Este directorio se establece al directorio raíz común de todas las rutas afectadas.

Tenga en cuenta que, aunque hemos dado estos nombres de parámetros por conveniencia, no tiene que referirse a estos nombres en la configuración de los ganchos. Todos los parámetros listados para un gancho en concreto se pasan siempre, los quiera o no ;-)

Si quiere que la operación de Subversion espere a que el gancho termine, marque Esperar a que el script termine.

Normalmente querrá ocultar esas feas ventanas DOS cuando se ejecute el script, por lo que la casilla Ocultar el script mientras se ejecuta está marcada por defecto.

Puede encontrar scripts gancho de cliente de ejemplo en la carpeta `contrib` del *repositorio de TortoiseSVN* [http://tortoisesvn.googlecode.com/svn/trunk/contrib/hook-scripts]. (Sección 3, "¡TortoiseSVN es gratis!" le explica cómo acceder al repositorio).

4.30.8.1. Integración con el gestor de incidencias

TortoiseSVN puede utilizar un plugin COM para interrogar a los gestores de incidencias dentro del diálogo de confirmación. El uso de estos plugins se describe en Sección 4.28.2, "Obteniendo información desde el gestor de incidencias". Si el administrador de su sistema le ha proporcionado uno de estos plugins, y ya está instalado y registrado, este es el lugar para especificar cómo se integra con su copia de trabajo.

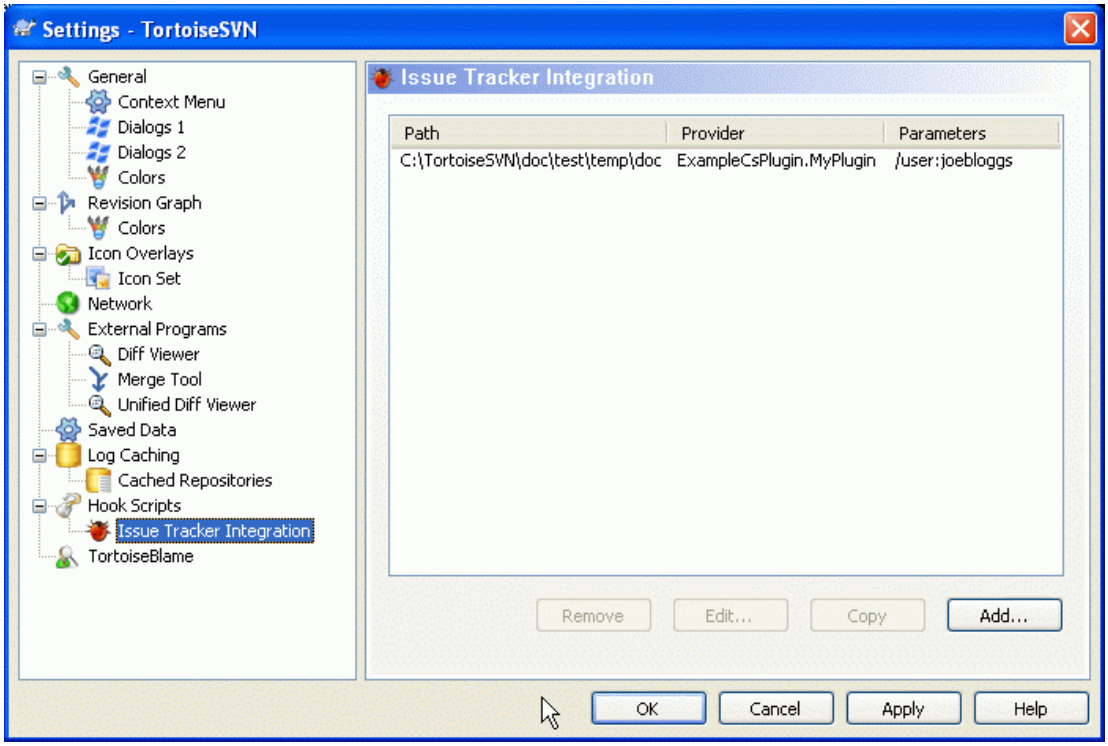


Figura 4.68. El diálogo de Configuración, página Integración con control de incidencias

Pulse en Añadir... para utilizar el plugin en una copia de trabajo en concreto. Aquí puede especificar la ruta de la copia de trabajo, elegir qué plugin desea utilizar desde la lista desplegable con todos los plugins de gestores de incidencia registrados, y cualquier parámetro que necesite. Los parámetros son específicos de cada plugin, pero pueden incluir su nombre de usuario en el gestor de incidencias, para que el plugin pueda preguntar sólo aquellas incidencias que tenga asignadas.

If you want all users to use the same COM plugin for your project, you can specify the plugin also with the properties `bugtraq:provideruuid` and `bugtraq:providerparams`.

bugtraq:provideruuid
: This property specifies the COM UUID of the IBugtraqProvider, for example `{91974081-2DC7-4FB1-B3BE-0DE1C8D6CE4E}`. (this example is the UUID of the *Gurtle*

bugtraq provider [http://code.google.com/p/gurtle/], which is a provider for the *Google Code* [http://code.google.com/hosting/] issue tracker).

bugtraq:providerparams
: This property specifies the parameters passed to the IBugtraqProvider.

Please check the documentation of your IBugtraqProvider plugin to find out what to specify in these two properties.

4.30.9. Configuración de TortoiseBlame

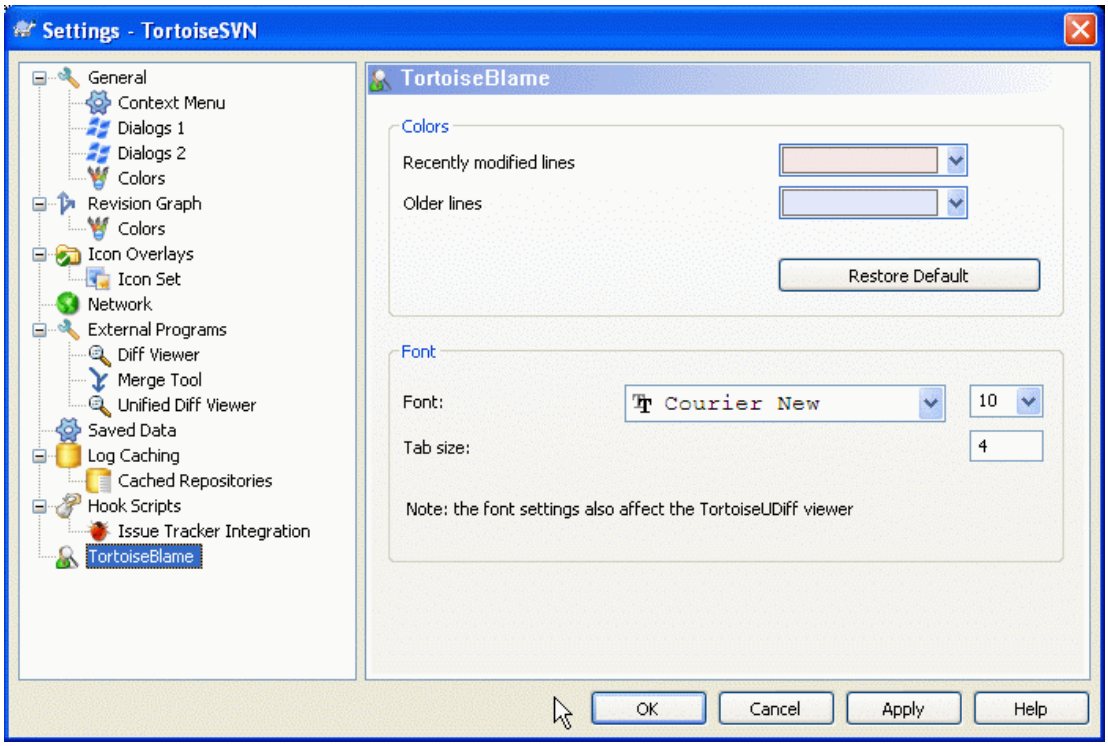


Figura 4.69. El diálogo Configuración, página TortoiseBlame

Las configuraciones utilizadas por TortoiseBlame se controlan desde el menú contextual principal, no directamente con el propio TortoiseBlame.

Colores
: TortoiseBlame puede utilizar el color de fondo para indicar la antiguedad de las líneas en un fichero. Proponga los colores de los extremos especificando los colores para la revisión más nueva y más antigua, y TortoiseBlame utiliza una interpolación lineal entre esos colores de acuerdo a la revisión del repositorio indicada en cada línea.

Fuente
: Puede seleccionar la fuente utilizada para mostrar el texto, y el tamaño de punto a utilizar. Esto se aplica tanto al contenido del fichero como al autor y la información de la revisión mostrados en el panel izquierdo.

Tabulaciones
: Define cuántos espacios se utilizarán para la expansión cuando se encuentre un carácter de tabulación en el contenido del fichero.

4.30.10. Configuraciones del registro

Unas pocas configuraciones poco utilizadas están disponibles sólo editando directamente el registro. No hace falta decirle que sólo debería editar estos valores del registro únicamente si sabe lo que está haciendo.

Configuración
: Puede especificar un lugar diferente para el fichero de configuración de Subversion utilizando la clave del registro `HKCU\Software\TortoiseSVN\ConfigDir`. Esto afectará a todas las operaciones de TortoiseSVN.

Icono de la bandeja de la caché
: Para añadir en la barra de tareas un icono de la caché para el programa TSVNCache, cree una entrada `DWORD` con valor 1 en `HKCU\Software\TortoiseSVN\CacheTrayIcon`. Esto realmente sólo es útil para desarrolladores dado que le permite terminar el programa de forma normal.

Debug
: Para mostrar los parámetros de línea de comandos pasados desde la extensión shell a TortoiseProc.exe cree una clave `DWORD` con valor 1 en `HKCU\Software\TortoiseSVN\Debug`.

Iconos del Menú contextual
: This can be useful if you use something other than the windows explorer or if you get problems with the context menu displaying correctly. create a `DWORD` key with a value of 0 at `HKCU\Software\TortoiseSVN\ShowContextMenuIcons` if you don't want TortoiseSVN to not show icons for the shell context menu items. Set this value to 1 to show the icons again.

Bloquear estado sobreimpresionado
: Si no desea que el explorador actualice las sobreimpresiones de estado mientras otro comando TortoiseSVN se esté ejecutando (por ejemplo Actualizar, Confirmar, ...), cree una clave `DWORD` con valor 1 en `HKCU\Software\TortoiseSVN\BlockStatus`.

URL de comprobación de actualizaciones
: `HKCU\Software\TortoiseSVN\UpdateCheckURL` contiene la URL que TortoiseSVN utiliza para intentar descargar un fichero de texto para averiguar si hay actualizaciones disponibles. También puede establecer esto bajo `HKLM` en vez de `HKCU` si lo desea, pero `HKCU` sobreescribe la configuración de `HKLM`. Esto puede ser útil para administradores de empresas que no desean que sus usuarios actualicen TortoiseSVN hasta que ellos lo aprueben.

Nombres de fichero sin extensiones en la lista de auto-completar
: La lista de auto-completar mostrada en el editor de mensajes de confirmación muestra los nombres de los ficheros que se proponen para confirmar. Para incluir también esos nombres sin sus extensiones, cree una clave `DWORD` con un valor de 1 en `HKCU\Software\TortoiseSVN\AutocompleteRemovesExtensions`.

Columnas del explorador en todo lugar
: Las columnas extra que TortoiseSVN añade a la vista Detalle en el Explorador de Windows normalmente sólo se activan en una copia de trabajo. Si desea que estén accesibles en cualquier lugar, no sólo en las copias de trabajo, cree una clave `DWORD` con un valor de 1 en `HKCU\Software\TortoiseSVN\ColumnsEveryWhere`.

Separador de registro de fusión
: Cuando fusiona revisiones desde otra rama, y hay información de registro de fusión disponible, los mensajes de las revisiones que fusiona se reunirán para crear un mensaje de registro de confirmación. Se utilizará una cadena de texto predefinida para separar los mensajes de registro individuales de las revisiones fusionadas. Si lo desea, puede crear una clave `SZ` en `HKCU\Software\TortoiseSVN\MergeLogSeparator` que contenga una cadena de separación de su elección.

Ver siempre la autoría de los cambios con TortoiseMerge
: TortoiseSVN le permite asignar visores de diferencias externos. La mayoría de estos visores, sin embargo, no están preparados para ver la autoría de los cambios (Sección 4.23.2, “Autoría de las diferencias”), por lo que quizás desee volver a utilizar TortoiseMerge en este caso.

Para hacerlo, cree una clave `DWORD` con un valor de 1 en `HKCU\Software\TortoiseSVN\DiffBlamesWithTortoiseMerge`.

Resalte de revisión actual para carpetas en el diálogo de registro
: The log dialog highlights the current working copy revision when the log is shown for a file. To do the same thing for a folder requires a working copy crawl, which is the default action, but it can be a slow operation for large working copies. If you want to change the operation of this feature you must create a `DWORD` registry key at `HKCU\Software\TortoiseSVN\RecursiveLogRev`. A value of 0 disables the feature (no highlighting for folders), a value of 1 (default) will fetch the status recursively (find the highest revision in the working copy tree), and a value of 2 will check the revision of the selected folder itself, but will not check any child items.

Obtener falla si un ítem del mismo nombre existe
: Por defecto, si obtiene una copia de trabajo sobre una estructura de carpetas sin versionar preexistente, tal y como podría hacer tras una importación, cualquier ítem que exista y que difiera del contenido del repositorio se dejará sin cambios y se marcará como modificado. Cuando vaya a confirmar, es su copia de trabajo la que se mandará de nuevo al repositorio. Hay quien prefiere que la confirmación falle si el contenido existente difiere, de forma que si dos personas añaden el mismo fichero, la versión de la seguna persona no sobreescriba la versión original por error. Si desea provocar que las obtenciones fallen en este caso debe crear una clave de registro `DWORD` con valor 0 en `HKCU\Software\TortoiseSVN\AllowUnversionedObstruction`.

4.30.11. Carpetas de trabajo de Subversion

VS.NET 2003, cuando se utilizan proyectos web, es incapaz de manejar las carpetas `.svn` que Subversion emplea para almacenar su información interna. Esto no es un error de Subversion. El error está en VS.NET 2003 y las extensiones de Frontpage que utiliza.

> Tenga en cuenta que el bug está solucionado en VS2005 y versiones posteriores.

Desde la versión 1.3.0 de Subversion y TortoiseSVN, puede establecer la variable de entorno `SVN_ASP_DOT_NET_HACK`. Si se establece esa variable, Subversion utilizará carpetas `_svn` en vez de carpetas `.svn`. Debe reiniciar el shell para que esa variable de entorno tenga efecto. Normalmente, eso significa reiniciar su PC. Para hacerlo más sencillo, ahora puede hacer esto desde la página de configuración general utilizando una simple casilla - lea Sección 4.30.1, “Configuración general”.

Para más información, y para ver otras formas de evitar este problema en primer lugar, compruebe el artículo sobre este tema en nuestro *FAQ* [http://tortoisesvn.net/aspdotnethack].

4.31. Último paso

> **¡Donar!**
>
> Incluso aunque TortoiseSVN y TortoiseMerge son gratuitos, puede ayudar a los desarrolladores enviándoles parches y tomando un rol activo en el desarrollo. También puede ayudar a alegrarnos las horas interminables que gastamos delante de nuestros ordenadores.
>
> Nos encanta escuchar música mientras trabajamos en TortoiseSVN. Y dado que gastamos muchas horas en el proyecto necesitamos *mucha* música. Por tanto, hemos preparado algunas listas de deseos con nuestros CDs y DVDs de música favoritos: *http://tortoisesvn.tigris.org/donate.html* Por favor mire también la lista de personas que han contribuido al proyecto enviando parches o traducciones.

Capítulo 5. El programa SubWCRev

SubWCRev es un programa de consola para Windows que puede utilizarse para leer el estado de una copia de trabajo local y opcionalmente realizar sustituciones de palabras clave en un fichero plantilla. A menudo se utiliza como parte del proceso de compilación como una forma de incorporar información de la copia de trabajo en el objeto que está construyendo. Típicamente, se puede utilizar para incluir el número de revisión en un diálogo “Acerca de”.

5.1. La línea de comandos de SubWCRev

SubWCRev lee el estado de Subversion de todos los ficheros en una copia de trabajo, excluyendo los externos por defecto. Apunta el número de revisión de confirmación más alto que encuentra, y la fecha de esa confirmación. También apunta si hay modificaciones locales en la copia de trabajo, o revisiones de actualización mezcladas. El número de revisión, el rango de revisiones de actualización y el estado de las modificaciones se muestra por la salida estandar stdout.

SubWCRev.exe se llama desde la línea de comandos o desde un script, y se controla utilizando parámetros de la línea de comandos.

```
SubWCRev WorkingCopyPath [SrcVersionFile DstVersionFile] [-nmdfe]
```

`WorkingCopyPath` es la ruta a la copia de trabajo que se va a comprobar. Sólo puede utilizar SubWCRev en copias de trabajo, no directamente en el repositorio. La ruta puede ser absoluta o relativa al directorio actual.

Si quiere que SubWCRev realice sustituciones de palabras clave, para que se graben los campos como la revisión del repositorio y la URL a un fichero de texto, debe proporcionar un fichero plantilla `SrcVersionFile` y un fichero de salida `DstVersionFile` que contendrá la versión sustituida de la plantilla.

Hay un número de modificadores opcionales que afectan la forma en la que SubWCRev trabaja. Si utiliza más de una, debe especificarlos como un único grupo, por ejemplo, `-nm` y no `-n -m`.

Cambiar	Descripción
-n	Si se especifica esta opción, SubWCRev terminará con `ERRORLEVEL 7` si la copia de trabajo tiene modificaciones locales. Esto puede utilizarse para evitar compilaciones cuando hay presentes cambios sin confirmar.
-m	Si se especifica esta opción, SubWCRev terminará con `ERRORLEVEL 8` si la copia de trabajo contiene revisiones mezcladas. Esto puede utilizarse para evitar compilaciones de una copia de trabajo parcialmente actualizada.
-d	Si se especifica esta opción, SubWCRev terminará con `ERRORLEVEL 9` si el fichero de destino ya existe.
-f	Si se especifica esta opción, SubWCRev incluirá la revisión del último cambio de las carpetas. El comportamiento por defecto es utilizar únicamente los ficheros cuando se obtienen los números de revisión.
-e	Si se especifica esta opción, SubWCRev examinará los directorios que se han incluído con `svn:externals`, pero sólo si son del mismo repositorio. El comportamiento por defecto es ignorar los externos.
-x	Si se especifica esta opción, SubWCRev mostrará los números de revisión en HEX.
-X	Si se especifica esta opción, SubWCRev mostrará los números de revisión en HEX, anteponiendo '0X'.

Tabla 5.1. Lista de opciones de línea de comandos disponible

5.2. Sustitución de palabras clave

Si se especifican ficheros de origen y destino, SubWCRev copia el fichero origen al destino, realizando las siguientes sustituciones de palabras clave:

Palabra clave	Descripción
$WCREV$	Se reemplaza con la revisión de confirmación más alta de la copia de trabajo.
$WCDATE$	Reemplazado con la fecha/hora de confirmación de la revisión de confirmación más alta. Por defecto, se utiliza el formato internacional: `yyyy-mm-dd hh:mm:ss`. Alternativamente, puede especificar un formato propio que se utilizará con `strftime()`, por ejemplo: `$WCDATE=%a %b %d %I:%M:%S %p$`. Para obtener una lista de los caracteres de formateo disponibles, mire en *referencia en línea* [http://www.cppreference.com/stddate/strftime.html].
$WCNOW$	Reemplazado por la fecha/hora actual del sistema. Esto puede utilizarse para indicar la fecha de compilación. El formato del tiempo es el descrito por `$WCDATE$`.
$WCRANGE$	Se reemplaza con el rango de revisiones de actualización de la copia de trabajo. Si la copia de trabajo está en un estado consistente, esto será una única revisión. Si la copia de trabajo contiene revisiones mezcladas, bien por estar desactualizada, o bien porque se ha realizado deliberadamente una actualización a una revisión concreta, se mostrará el rango de la forma 100:200.
$WCMIXED$	`$WCMIXED?TextoSI:TextoNO$` se reemplaza por `TextoSI` si hay revisiones de actualización mezcladas, o por `TextoNO` si no es así.
$WCMODS$	`$WCMODS?TextoSI:TextoNO$` se reemplaza por `TextoSI` si hay modificaciones locales, o por `TextoNO` si no es así.
$WCURL$	Se reemplaza con la URL del repositorio de la copia de trabajo pasada a SubWCRev.
$WCINSVN$	`$WCINSVN?TextoSI:TextoNO$` se reemplaza por `TextoSI` si la entrada está versionada, o por `TextoNO` si no es así.
$WCNEEDSLOCK$	`$WCNEEDSLOCK?TextoSI:TextoNO$` se reemplaza por `TextoSI` si la entrada tiene establecida la propiedad `svn:needs-lock`, o por `TextoNO` si no es así.
$WCISLOCKED$	`$WCISLOCKED?TextoSI:TextoNO$` se reemplaza por `TextoSI` si la entrada está bloqueada, o por `TextoNO` si no es así.
$WCLOCKDATE$	Se reemplaza con la fecha del bloqueo. El formato del tiempo se puede utilizar como se describe en `$WCDATE$`.
$WCLOCKOWNER$	Reemplazado con el nombre del propietario del bloqueo.
$WCLOCKCOMMENT$	Reemplazado por el comentario del bloqueo.

Tabla 5.2. Lista de opciones de línea de comandos disponible

Sugerencia

Some of these keywords apply to single files rather than to an entire working copy, so it only makes sense to use these when SubWCRev is called to scan a single file. This applies to `$WCINSVN$`, `$WCNEEDSLOCK$`, `$WCISLOCKED$`, `$WCLOCKDATE$`, `$WCLOCKOWNER$` and `$WCLOCKCOMMENT$`.

5.3. Ejemplo de palabras clave

El siguiente ejemplo le muestra cómo se sustituyen las palabras clave de un fichero plantilla en el fichero resultado.

```
// Fichero de prueba para SubWCRev: fichprueba.tmpl

char *Revision = "$WCREV$";
char *Modified = "$WCMODS?Modificado:No modificado$";
char *Date     = "$WCDATE$";
char *Range    = "$WCRANGE$";
char *Mixed    = "$WCMIXED?Copia de trabajo con revisiones mezcladas:No mezcladas
char *URL      = "$WCURL$";

#if $WCMODS?1:0$
#error El origen ha sido modificado
#endif

// Fin de fichero
```

Después de ejecutar `SubWCRev.exe ruta\a\la\copia\de\trabajo fichprueba.tmpl fichprueba.txt`, el fichero de salida `fichprueba.txt` se parecería a esto:

```
// Fichero de prueba para SubWCRev: fichprueba.txt

char *Revision = "3701";
char *Modified = "Modificado";
char *Date     = "2005/06/15 11:15:12";
char *Range    = "3699:3701";
char *Mixed    = "Copia de trabajo con revisiones mezcladas";
char *URL      = "http://proyecto.dominio.org/svn/trunk/src
trunk/src/SubWCRev";

#if 1
#error El código fuente ha sido modificado
#endif

// Fin de fichero
```

Sugerencia

Un fichero como este se incluirá en la compilación por lo que puede esperar que esté versionado. Asegúrese de que versiona el fichero de plantilla, no el fichero generado, porque si no cada vez que regenere el fichero de versión deberá confirmar el cambio, lo que a su vez significaría que el fichero de versión necesita ser actualizado.

5.4. interfaz COM

Si necesita acceder a la información de las revisiones de Subversion desde otros programas, puede utilizar el interfaz COM de SubWCRev. El objeto a crear es `SubWCRev.object`, y se soportan los siguientes métodos:

Método	Descripción
.GetWCInfo	Este método atraviesa la copia de trabajo obteniendo la información de revisión. Naturalmente debe llamarlo antes de que pueda acceder a la

Método	Descripción
	información utilizando los métodos restantes. El primer parámetro es la ruta. El segundo parámetro debe ser true si desea incluir las revisiones de las carpetas; equivalente al parámetro de la línea de comandos `-f`. El tercer parámetro debería ser true si desea incluir svn:externals; equivalente al parámetro de la línea de comandos `-e`.
.Revision	La revisión de confirmación más alta de la copia de trabajo. Equivalente a `$WCREV$`
.Date	La fecha/hora de la revisión de confirmación más alta. Equivalente a `$WCDATE$`
.Author	El autor de la revisión de confirmación más alta, esto es, la última persona que confirmó cambios en la copia de trabajo.
.MinRev	La revisión mínima de actualización, como se muestra en `$WCRANGE$`
.MaxRev	La revisión máxima de actualización, como se muestra en `$WCRANGE$`
.HasModifications	True si hay modificaciones locales
.Url	Se reemplaza con la URL del repositorio de la copia de trabajo usada en `GetWCInfo`. Equivalente a `$WCURL$`
.IsSvnItem	True si el ítem está versionado.
.NeedsLocking	True si el ítem tiene la propiedad `svn:needs-lock` establecida.
.IsLocked	True si el ítem está bloqueado.
.LockCreationDate	Cadena representando la fecha cuando el bloqueo se creó, o una cadena vacía si el ítem no está bloqueado.
.LockOwner	Cadena representando el propietario del bloqueo, o una cadena vacía si el ítem no está bloqueado.
.LockComment	El mensaje que se introdujo cuando se creó el bloqueo.

Tabla 5.3. métodos de automatización/COM soportados

El siguiente ejemplo muestra cómo se podría usar el interfaz.

```
// testCOM.js - javascript file
// test script for the SubWCRev COM/Automation-object

filesystem = new ActiveXObject("Scripting.FileSystemObject");

revObject1 = new ActiveXObject("SubWCRev.object");
revObject2 = new ActiveXObject("SubWCRev.object");
revObject3 = new ActiveXObject("SubWCRev.object");
revObject4 = new ActiveXObject("SubWCRev.object");

revObject1.GetWCInfo(
    filesystem.GetAbsolutePathName("."), 1, 1);
revObject2.GetWCInfo(
    filesystem.GetAbsolutePathName(".."), 1, 1);
revObject3.GetWCInfo(
    filesystem.GetAbsolutePathName("SubWCRev.cpp"), 1, 1);
revObject4.GetWCInfo(
    filesystem.GetAbsolutePathName("..\\.."), 1, 1);

wcInfoString1 = "Revision = " + revObject1.Revision +
    "\nMin Revision = " + revObject1.MinRev +
```

```
    "\nMax Revision = " + revObject1.MaxRev +
    "\nDate = " + revObject1.Date +
    "\nURL = " + revObject1.Url + "\nAuthor = " +
    revObject1.Author + "\nHasMods = " +
    revObject1.HasModifications + "\nIsSvnItem = " +
    revObject1.IsSvnItem + "\nNeedsLocking = " +
    revObject1.NeedsLocking + "\nIsLocked = " +
    revObject1.IsLocked + "\nLockCreationDate = " +
    revObject1.LockCreationDate + "\nLockOwner = " +
    revObject1.LockOwner + "\nLockComment = " +
    revObject1.LockComment;
wcInfoString2 = "Revision = " + revObject2.Revision +
    "\nMin Revision = " + revObject2.MinRev +
    "\nMax Revision = " + revObject2.MaxRev +
    "\nDate = " + revObject2.Date +
    "\nURL = " + revObject2.Url + "\nAuthor = " +
    revObject2.Author + "\nHasMods = " +
    revObject2.HasModifications + "\nIsSvnItem = " +
    revObject2.IsSvnItem + "\nNeedsLocking = " +
    revObject2.NeedsLocking + "\nIsLocked = " +
    revObject2.IsLocked + "\nLockCreationDate = " +
    revObject2.LockCreationDate + "\nLockOwner = " +
    revObject2.LockOwner + "\nLockComment = " +
    revObject2.LockComment;
wcInfoString3 = "Revision = " + revObject3.Revision +
    "\nMin Revision = " + revObject3.MinRev +
    "\nMax Revision = " + revObject3.MaxRev +
    "\nDate = " + revObject3.Date +
    "\nURL = " + revObject3.Url + "\nAuthor = " +
    revObject3.Author + "\nHasMods = " +
    revObject3.HasModifications + "\nIsSvnItem = " +
    revObject3.IsSvnItem + "\nNeedsLocking = " +
    revObject3.NeedsLocking + "\nIsLocked = " +
    revObject3.IsLocked + "\nLockCreationDate = " +
    revObject3.LockCreationDate + "\nLockOwner = " +
    revObject3.LockOwner + "\nLockComment = " +
    revObject3.LockComment;
wcInfoString4 = "Revision = " + revObject4.Revision +
    "\nMin Revision = " + revObject4.MinRev +
    "\nMax Revision = " + revObject4.MaxRev +
    "\nDate = " + revObject4.Date +
    "\nURL = " + revObject4.Url + "\nAuthor = " +
    revObject4.Author + "\nHasMods = " +
    revObject4.HasModifications + "\nIsSvnItem = " +
    revObject4.IsSvnItem + "\nNeedsLocking = " +
    revObject4.NeedsLocking + "\nIsLocked = " +
    revObject4.IsLocked + "\nLockCreationDate = " +
    revObject4.LockCreationDate + "\nLockOwner = " +
    revObject4.LockOwner + "\nLockComment = " +
    revObject4.LockComment;

WScript.Echo(wcInfoString1);
WScript.Echo(wcInfoString2);
WScript.Echo(wcInfoString3);
WScript.Echo(wcInfoString4);
```

Capítulo 6. Interfase IBugtraqProvider

Para lograr una integración más compacta con herramientas de seguimiento de problemas, que la lograda simplemente usando las propiedades `bugtraq:`, TortoiseSVN puede usar extensiones COM. Con estas extensiones es posible acceder a información directamente en el seguidor de problemas, interactuar con el usuario y proveer información a TortoiseSVN acerca de problemas no resueltos, verificar mensajes de registro introducidos por el usuario y hasta ejecutar acciones luego de una confirmación exitosa para, por ejemplo, cerrar un problema.

Nosotros no podemos proveer información y tutoriales acerca de como implementar un objeto COM en su lenguaje favorito, pero sí tenemos ejemplos de extensiones en C++/ATL y C# en nuestro repositorio in el directiorio `contrib/issue-tracker-plugins`. En ese directorio usted tambien puede encontrar los archivos requeridos para compilar su extensión. (Sección 3, “¡TortoiseSVN es gratis!” explica como acceder al repositorio).

6.1. La interfaz de IBugtraqProvider

TortoiseSVN 1.5 y posteriores puede usar extensiones que implementan la interfaz IBugtraqPorvider. La interfaz provee algunos métodos que las extensiones pueden usar para interactuar con la herramienta de seguimiento.

<placeholder-1> Este método es llamado desde el diálogo de configuración en el cual el usuario puede agregar y configurar la extensión. Los `parámetros` string pueden ser utilizados por la extensión para obtener información adicional requerida, ej., la URL al seguimiento de tareas, información de identificación, etc. La extensión debería verificar los `parámetros` string y mostrar un diálogo de error si la cadena de caracteres no es válida. El parámetro `hParentWnd` debería ser utilizado para cualquier diálogo que la extensión muestre como si fuese la ventana principal. La extensión debe retornar TRUE si la validación de los `parámetros` string es exitosa. Si la extensión retorna FALSE, el diálogo de configuración no le permitirá al usuario agregar la extensión a una ruta de una copia de trabajo.</ placeholder-1>

<placeholder-1> La extensión puede proveer un string aquí que sera utilizado en el diálogo de actualización del TortoiseSVN para el botón que invoca a la extensión, ej., "Elija tema" o "Seleccione tiquet". Asegúrese de que el string no sea demasiado largo, de lo contrario podría no caber en el botón. Si el método devuelve un error (ej., `E_NOTIMPL`), un texto por defecto será utilizado para el botón.</ placeholder-1>

```
HRESULT GetCommitMessage (
  // Ventana principal de su Interfaz proveedora.
  [in] HWND hParentWnd,

  // Parámetros para su proveedor.
  [in] BSTR parameters,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // El texto ya presente in el mensaje de actualización.
  // Su proveedor debería incluir este texto en los mensajes
  // nuevos, cuando corresponda.
  [in] BSTR originalMessage,

  // El nuevo texto para el mensaje de actualización.
  // Reemplazada el mensaje original.
  [out, retval] BSTR *newMessage
);
```

This is the main method of the plugin. This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button. The `parameters` string is the string the user has to enter in the settings dialog when he configures the plugin. Usually a plugin would use this to find the URL of the issue tracker and/or login information or more. The `commonRoot` string contains the parent path of all items selected to bring up the commit dialog. Note that this is *not* the root path of all items which the user has selected in the commit dialog. The `pathList` parameter contains an array of paths (as strings) which the user has selected for the commit. The `originalMessage` parameter contains the text entered in the log message box in the commit dialog. If the user has not yet entered any text, this string will be empty. The `newMessage` return string is copied into the log message edit box in the commit dialog, replacing whatever is already there. If a plugin does not modify the `originalMessage` string, it must return the same string again here, otherwise any text the user has entered will be lost.

6.2. La interfaz de IBugtraqProvider2

In TortoiseSVN 1.6 a new interface was added which provides more functionality for plugins. This IBugtraqProvider2 interface inherits from IBugtraqProvider.

```
HRESULT GetCommitMessage2 (
  // Parent window for your provider's UI.
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,
  // The common URL of the commit
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The text already present in the commit message.
  // Your provider should include this text in the new message,
  // where appropriate.
  [in] BSTR originalMessage,

  // You can assign custom revision properties to a commit
  // by setting the next two params.
  // note: Both safearrays must be of the same length.
  //       For every property name there must be a property value!

  // The content of the bugID field (if shown)
  [in] BSTR bugID,

  // Modified content of the bugID field
  [out] BSTR * bugIDOut,

  // The list of revision property names.
  [out] SAFEARRAY(BSTR) * revPropNames,

  // The list of revision property values.
  [out] SAFEARRAY(BSTR) * revPropValues,

  // The new text for the commit message.
  // This replaces the original message
  [out, retval] BSTR * newMessage
);
```

This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button. This method is called instead of `GetCommitMessage()`. Please refer to the documentation for `GetCommitMessage` for the parameters that are also used there. The parameter `commonURL` is the

parent URL of all items selected to bring up the commit dialog. This is basically the URL of the `commonRoot` path. The parameter `bugID` contains the content of the bug-ID field (if it is shown, configured with the property `bugtraq:message`). The return parameter `bugIDOut` is used to fill the bug-ID field when the method returns. The `revPropNames` and `revPropValues` return parameters can contain name/value pairs for revision properties that the commit should set. A plugin must make sure that both arrays have the same size on return! Each property name in `revPropNames` must also have a corresponding value in `revPropValues`. If no revision properties are to be set, the plugin must return empty arrays.

```
HRESULT CheckCommit (
  [in] HWND hParentWnd,
  [in] BSTR parameters,
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,
  [in] BSTR commitMessage,
  [out, retval] BSTR * errorMessage
);
```

This method is called right before the commit dialog is closed and the commit begins. A plugin can use this method to validate the selected files/folders for the commit and/or the commit message entered by the user. The parameters are the same as for `GetCommitMessage2()`, with the difference that `commonURL` is now the common URL of all *checked* items, and `commonRoot` the root path of all checked items. The return parameter `errorMessage` must either contain an error message which TortoiseSVN shows to the user or be empty for the commit to start. If an error message is returned, TortoiseSVN shows the error string in a dialog and keeps the commit dialog open so the user can correct whatever is wrong. A plugin should therefore return an error string which informs the user *what* is wrong and how to correct it.

```
HRESULT  OnCommitFinished (
  // Parent window for any (error) UI that needs to be displayed.
  [in] HWND hParentWnd,

  // The common root of all paths that got committed.
  [in] BSTR commonRoot,

  // All the paths that got committed.
  [in] SAFEARRAY(BSTR) pathList,


  // The text already present in the commit message.
  [in] BSTR logMessage,

  // The revision of the commit.
  [in] ULONG revision,


  // An error to show to the user if this function
  // returns something else than S_OK
  [out, retval] BSTR * error
);
```

This method is called after a successful commit. A plugin can use this method to e.g., close the selected issue or add information about the commit to the issue. The parameters are the same as for `GetCommitMessage2`.

```
HRESULT HasOptions(
  // Whether the provider provides options
  [out, retval] VARIANT_BOOL *ret
);
```

This method is called from the settings dialog where the user can configure the plugins. If a plugin provides its own configuration dialog with `ShowOptionsDialog`, it must return TRUE here, otherwise it must return FALSE.

```
HRESULT ShowOptionsDialog(
  // Parent window for the options dialog
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,

  // The parameters string
  [out, retval] BSTR * newparameters
);
```

This method is called from the settings dialog when the user clicks on the "Options" button that is shown if `HasOptions` returns TRUE. A plugin can show an options dialog to make it easier for the user to configure the plugin. The `parameters` string contains the plugin parameters string that is already set/entered. The `newparameters` return parameter must contain the parameters string which the plugin constructed from the info it gathered in its options dialog. That `paramameters` string is passed to all other IBugtraqProvider and IBugtraqProvider2 methods.

Apéndice A. Preguntas más frecuentes (FAQ)

Dado que el desarrollo de TortoiseSVN es continuo, a veces es complicado mantener la documentación completamente al día. Mantenemos un *FAQ online* [http://tortoisesvn.tigris.org/faq.html] que contiene una selección de las preguntas más frecuentes que recibimos en las listas de correo de TortoiseSVN <dev@tortoisesvn.tigris.org> y <users@tortoisesvn.tigris.org>.

También mantenemos un *Gestor de incidencias* [http://issues.tortoisesvn.net] que le puede informar de algunas de las cosas que tenemos en nuestra lista de tareas pendientes, y errores que ya han sido corregidos. Si cree que ha encontrado un error, o quiere pedir una nueva funcionalidad, mire aquí primero para ver si alguien se le adelantó.

Si tiene una pregunta que no está contestada en ningún otro lugar, el mejor lugar para preguntar es en una de las listas de correo. <users@tortoisesvn.tigris.org> es la que debe utilizar si tiene preguntas acerca del uso de TortoiseSVN. Si desea ayudar con el desarrollo de TortoiseSVN, entonces debería tomar parte en las discusiones de <dev@tortoisesvn.tigris.org>.

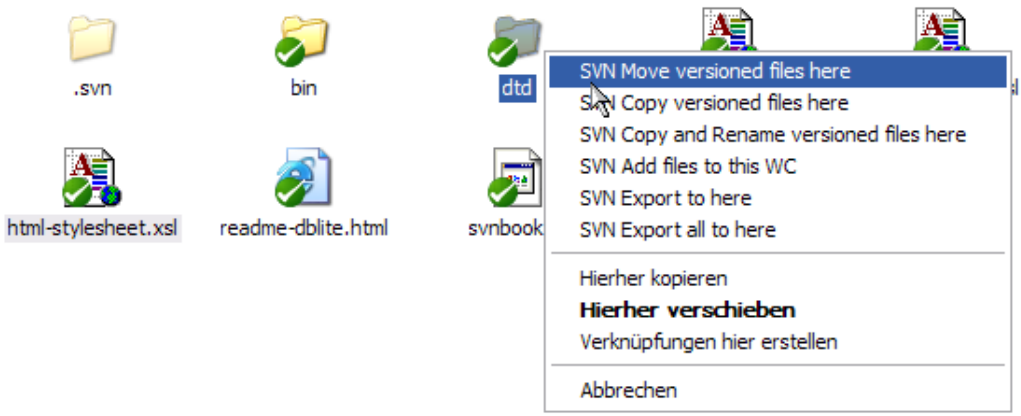
Apéndice B. ¿Cómo...?

Este apéndice contiene soluciones a los problemas o preguntas que puede tener mientras utiliza TortoiseSVN.

B.1. Mover/copiar muchos ficheros de golpe

Se pueden mover/copiar ficheros de uno en uno utilizando TortoiseSVN → Renombrar.... Pero si desea mover/copiar muchos ficheros, esta forma es muy lenta y trabajosa.

La forma recomendada es arrastrar con el botón derecho los ficheros a su nuevo destino. Simplemente haga click con el botón derecho en los ficheros que desea mover o copiar sin soltar el botón del ratón. Luego arrastre los ficheros a su nuevo destino y suelte el botón del ratón. Aparecerá un menú contextual donde puede elegir Menú contextual → SVN Copiar ficheros versionados aquí o Menú contextual → SVN Mover ficheros versionados aquí.



B.2. Obligar a los usuarios a introducir un mensaje de registro

Hay dos formas de evitar que los usuarios puedan confirmar con un mensaje de registro vacío. Uno es específico de TortoiseSVN, el otro funciona con todos los clientes de Subversion, pero necesita acceso directo al servidor.

B.2.1. Script gancho en el servidor

Si tiene acceso directo al servidor del repositorio, puede instalar un script gancho pre-commit que rechace todas las confirmaciones con mensajes de registro cortos o vacíos.

En la carpeta del repositorio en el servidor, hay una subcarpeta `hooks` que contiene algunos scripts ganchos de ejemplo que puede utilizar. El fichero `pre-commit.tmpl` contiene un script de ejemplo que rechaza las confirmaciones si no se ha introducido un mensaje de registro, o si ese mensaje es demasiado corto. El fichero también tiene comentarios sobre cómo instalar/utilizar este script. Simplemente siga las instrucciones de ese fichero.

Este método es la forma recomendada si sus usuarios también utilizan otros clientes de Subversion además de TortoiseSVN. La parte negativa es que la confirmación se rechaza por el servidor, y por tanto los usuarios obtendrán un mensaje de error. El cliente no puede saber antes de realizar la confirmación que se rechazará. Si desea que TortoiseSVN deshabilite el botón Aceptar mientras el mensaje de registro sea demasiado corto, utilice el método descrito a continuación.

B.2.2. Propiedades del proyecto

TortoiseSVN utiliza propiedades para controlar algunas de sus funciones. Una de esas propiedades es la propiedad `tsvn:logminsize`.

Si establece esa propiedad en una carpeta, TortoiseSVN deshabilitará el botón Aceptar en todos los diálogos de confirmación hasta que el usuario introduzca un mensaje de registro con al menos la longitud especificada en la propiedad.

Para obtener información más detallada sobre esas propiedades de proyectos, por favor lea Sección 4.17, “Configuración del proyecto”

B.3. Actualizar los ficheros seleccionados desde el repositorio

Normalmente actualizará su copia de trabajo utilizando TortoiseSVN → Actualizar. Pero si sólo desea obtener algunos ficheros nuevos que un colega ha añadido sin fusionar ningún cambio en los demás ficheros a la vez, necesitará otra forma de actuación.

Utilice TortoiseSVN → Comprobar Modificaciones. y pulse en Comprobar repositorio para ver lo que se cambió en el repositorio. Seleccione los ficheros que desea actualizar localmente, y utilice el menú contextual para actualizar sólo esos ficheros.

B.4. Deshacer revisiones en el repositorio

B.4.1. Utilice el diálogo Registro de revisiones

La manera más fácil de revertir los cambios de una revisión en concreto, o de un rango de revisiones, es utilizar el diálogo de registro de revisiones. Este es también el método a utilizar si desea descartar cambios recientes y hacer que una revisión antigua se convierta en la nueva HEAD.

1. Seleccione el fichero o la carpeta en la que desea revertir los cambios. Si desea revertir todos los cambios, esta debería ser la carpeta más alta.

2. Seleccione TortoiseSVN → Mostrar registro para mostrar una lista de revisiones. Puede que necesite utilizar Obtener Todo o Siguientes 100 para mostrar la o las revisiones en las que está interesado.

3. Seleccione la revisión que desea revertir. Si desea deshacer un rango de revisiones, seleccione la primera y pulse la tecla **Mayúsculas** mientras selecciona la última. Tenga el cuenta que, para múltiples revisiones, el rango debe ser contínuo, sin huecos. Haga click con el botón derecho en la o las revisiones seleccionadas, y luego seleccione Menú Contextual → Revertir cambios de esta revisión.

4. O si desea hacer que una revisión antigua se convierta en la nueva revisión HEAD, haga click con el botón derecho en la revisión seleccionada, luego seleccione Menú Contextual → Revertir a esta revisión. Esto descartará *todos* los cambios que se hicieron después de la revisión seleccionada.

Ha revertido los cambios dentro de su copia de trabajo. Compruebe los resultados, y luego confirme los cambios.

B.4.2. Utilice el diálogo Fusionar

Para deshacer un rango grande de revisiones, puede utilizar el diálogo Fusionar. El método anterior utiliza la fusión por detrás; este método la utiliza explícitamente.

1. En su copia de trabajo seleccione TortoiseSVN → Fusionar.

2. En el campo Desde: introduzca la URL completa de la carpeta de la rama o la etiqueta que contiene los cambios que desea revertir en su copia de trabajo. Debería aparecer como la URL por defecto.

3. En el campo Desde la Revisión introduzca el número de revisión en la que está actualmente. Si está seguro de que nadie más está haciendo cambios, puede utilizar la revisión HEAD.

4. Asegúrese de que la casilla Usar la URL "Desde:" esté marcada.

5. En el campo Hasta la Revisión introduzca el número de revisión a la que desea revertir, esto es, justo la *anterior* a la primera revisión que va a ser revertida.

6. Pulse Aceptar para completar la fusión.

Ha revertido los cambios dentro de su copia de trabajo. Compruebe los resultados, y luego confirme los cambios.

B.4.3. Utilice `svndumpfilter`

Dado que TortoiseSVN nunca pierde datos, sus revisiones "deshechas" aún existen como revisiones intermedias en el repositorio. Sólo se ha cambiado la revisión HEAD a su estado previo. Si desea hacer que las revisiones desaparezcan completamente de su repositorio, borrando toda traza de que alguna vez existieran, deberá utilizar medidas más extremas. A menos de que haya una buena razón para hacer esto, *no se recomienda*. Una razón posible sería que alguien haya confirmado un documento confidencial en un repositorio público.

La única forma de eliminar datos de un repositorio es utilizar la herramienta de línea de comandos de Subversion `svnadmin`. Puede encontrar una descripción sobre cómo funciona esto en *Mantenimiento del repositorio* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.maint.html].

B.5. Comparar dos revisiones de un fichero o carpeta

Si desea comparar dos revisiones en la historia de un ítem, por ejemplo las revisiones 100 y 200 del mismo fichero, utilice TortoiseSVN → Mostrar Registro para ver la historia de revisiones de ese fichero. Seleccione las dos revisiones que desea comparar y luego utilice Menú Contextual → Comparar Revisiones.

Si desea comparar el mismo ítem en dos árboles diferentes, por ejemplo el tronco y una rama, puede utilizar el navegador de repositorios para abrir ambos árboles, seleccionar el fichero en los dos lugares, y luego utilizar Menú Contextual → Comparar Revisiones.

Si desea comparar dos árboles para ver lo que ha cambiado, por ejemplo el tronco y una versión etiquetada, puede utilizar TortoiseSVN → Gráfico de Revisiones. Seleccione los dos nodos a comparar, y luego utilice Menú Contextual → Comparar Revisiones HEAD. Esto le mostrará una lista de ficheros cambiados, y luego podrá seleccionar los ficheros individuales para ver los cambios en detalle. También puede exportar una estructura de árbol conteniendo todos los ficheros cambiados, o simplemente una list ade todos los ficheros cambiados. Puede obtener más información en Sección 4.10.3, "Comparando carpetas". Alternativamente, utilice Menú contextual → Diff unificado de las revisiones HEAD para ver un resumen de todas las diferencias, con un contexto mínimo.

B.6. Incluir un sub-proyecto común

A veces querrá incluir otro proyecto en su copia de trabajo, quizás el código de alguna librería. No quiere hacer un duplicado de ese código en su repositorio, porque perdería la conexión con el código original (y mantenido). O quizás tiene varios proyectos que comparten el núcleo de código. Hay al menos 3 formas de lidiar con esto.

B.6.1. Utilice svn:externals

Establece la propiedad `svn:externals` para una carpeta en su proyecto. Esta propiedad contiene una o más líneas; cada línea tiene el nombre de una subcarpeta que quiere utilizar como la carpeta de obtención del código común, y la URL del repositorio que desea obtener ahí. Para saber todos los detalles lea Sección 4.18, “Ítems externos”.

Confirme la nueva carpeta. Ahora, cuando actualice, Subversion traerá una copia de ese proyecto desde su repositorio en su copia de trabajo. Las subcarpetas se crearán automáticamente si es necesario. Cada vez que actualice su copia de trabajo, también recibirá la última versión de todos los proyectos externos.

Si los proyectos externos están en el mismo repositorio, cualquier cambio que haga allí se incluirá en la lista de confirmación cuando confirme su proyecto principal.

Si los proyectos externos están en repositorios diferentes, cualquier cambio que haga en el proyecto externo se notificará cuando cnofirme el proyecto principal, pero tendrá que confirmar esos cambios externos de forma separada.

De los tres métodos descritos, éste es el único que no necesita preparación en el lado del cliente. Una vez que los externos se especifican en las propiedades de la carpeta, todos los clientes obtendrán carpetas rellenas cuando se actualicen.

B.6.2. Utilice una copia de trabajo anidada

Cree una nueva carpeta dentro de su proyecto para contener el código común, pero no la añada a Subversion

Seleccione TortoiseSVN → Obtener en la nueva carpeta, y obtenga una copia del código común en ella. Ahora tiene anidada una copia de trabajo separada dentro de su copia de trabajo principal.

Las dos copias de trabajo son independientes. Cuando confirme los cambios de la padre, los cambios en la copia de trabajo anidada se ignoran. De la misma forma, cuando actualice el padre, la copia de trabajo anidada no se actualiza.

B.6.3. Utilice una ruta relativa

Si utiliza el mismo código principal común en varios proyectos, y no quiere tener varias copias de trabajo de él en cada proyecto que lo utilice, puede obtenerlo en un lugar separado que se relacione con todos los otros proyectos que lo utilicen. Por ejemplo:

```
C:\Proyectos\Proy1
C:\Proyectos\Proy2
C:\Proyectos\Proy3
C:\Proyectos\Comun
```

y refiérase al código común con una ruta relativa, por ejemplo `..\..\Comun\DSPcore`.

Si sus proyectos están dispersos en lugares sin relación entre sí puede utilizar una variante de esto, que es poner el código común en un lugar y utilizar la sustitución de letras de unidades para mapear esa ruta a algo que puede codificar en sus proyectos, por ejemplo, obtenga el código común en `D:\Documentos\Framework` o `C:\Documents and Settings\{login}\Mis Documentos\framework` y luego utilice

```
SUBST X: "D:\Documentos\framework"
```

para crear el mapeo de unidades utilizado en su código fuente. Su código puede entonces utilizar rutas absolutas.

```
#include "X:\superio\superio.h"
```

Este método sólo funcionará en un entorno de todo-PCs, y necesitará documentar los mapeos de unidades requeridos para que su equipo sepa dónde están esos ficheros misteriosos. Este método debería utilizarse estrictamente en entornos de desarrollo cerrados, y no se recomienda para su uso generalizado.

B.7. Crear un acceso directo a un repositorio

Si necesita abrir el navegador de repositorios frecuentemente en un lugar en concreto, puede crear un acceso directo en el escritorio utilizando el interfaz de automatización de TortoiseProc. Simplemente cree un nuevo acceso directo y ponga como destino:

```
TortoiseProc.exe /command:repobrowser /path:"url/al/repositorio" /notempfile
```

Por supuesto necesitará incluir la URL real del repositorio.

B.8. Ignorar ficheros que ya están versionados

Si accidentalmente añadió algunos ficheros que deberían haber sido ignorados, ¿cómo puede sacarlos del control de versiones sin perderlos? Quizás tiene su propio fichero de configuración IDE que no es parte del proyecto, pero que le costó bastante tiempo ajustar a su gusto.

Si aún no ha confirmado los ficheros añadidos, todo lo que debe hacer es utilizar TortoiseSVN → Revertir... para deshacer la operación. Entonces debería añadir los ficheros a la lista de ignorados para que no se añadan de nuevo por error.

Si los ficheros ya están en el repositorio, tendrá que trabajar un poco más.

1. Mantenga la tecla **Mayúsculas** pulsada para obtener el menú contextual extendido y utilice TortoiseSVN → Eliminar (mantener local) para marcar el fichero/carpeta para su eliminación del repositorio sin perder la copia local.
2. TortoiseSVN → Confirmar la carpeta padre.
3. Añadir el fichero o carpeta a la lista de ignorados para que no vuelva a tener este problema en el futuro.

B.9. Desversionar una copia de trabajo

Si tiene una copia de trabajo que quiere convertir de nuevo a un árbol de carpetas plano sin los directorios `.svn`, simplemente puede exportarla sobre si misma. Lea Sección 4.26.1, “Eliminando una copia de trabajo del control de versiones” para averiguar cómo hacerlo.

B.10. Eliminar una copia de trabajo

Si tiene una copia de trabajo que ya no necesita, ¿cómo puede eliminarla limpiamente? ¡Fácil - simplemente elimínela en el Explorador de Windows! Las copias de trabajo son entidades locales, y son auto-contenidas.

Apéndice C. Trucos útiles para los administradores

Este apéndice contiene soluciones a los problemas o preguntas que pueda tener cuando es reponsable de distribuir TortoiseSVN a múltiples ordenadores clientes.

C.1. Instalar TortoiseSVN utilizando políticas de grupo

El instalador de TortoiseSVN viene en un fichero MSI, lo que significa que no debería tener problemas para añadir ese fichero MSI a las políticas de grupo de su controlador de dominio.

Puede encontrar una buena guía sobre cómo hacerlo en el artículo de la base de datos de conocimiento 314934 de Microsoft: *http://support.microsoft.com/?kbid=314934*.

Las versiones 1.3.0 y posteriores de TortoiseSVN deben instalarse desde *Configuración del equipo* y no bajo *Configuración del usuario*. Esto es así porque esas versiones necesitan las nuevas DLLs de CRT y MFC, que sólo pueden ser distribuidas *por equipo* y no *por usuario*. Si realmente desea instalar TortoiseSVN por usuario, primero debe instalar el paquete MFC y CRT versión 8 de Microsoft en cada ordenador en el que quiera instalar TortoiseSVN por usuario.

C.2. Redirigir la comprobación de actualización

TortoiseSVN comprueba si hay una nueva versión disponible cada pocos días. Si hay una nueva versión disponible, aparece un diálogo informando al usuario sobre ello.

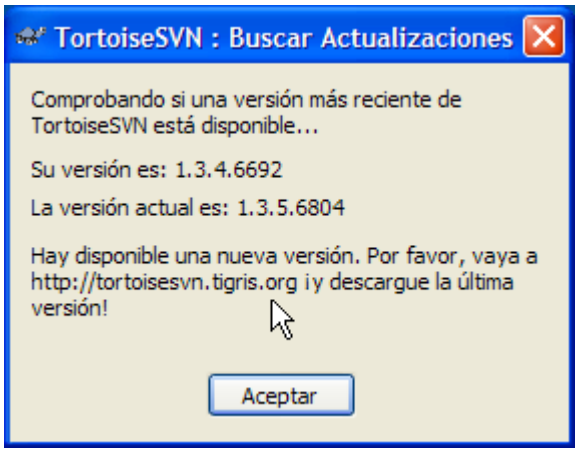


Figura C.1. El diálogo de Actualización

Si es responsable de muchos usuarios en su dominio, puede querer que sus usuarios utilicen sólo versiones que haya aprobado y que no siempre se instalen la última versión. Probablemente no querrá que aparezca el diálogo de actualización para que sus usuarios no vayan y se actualicen inmediatamente.

Las versiones 1.4.0 y superiores de TortoiseSVN le permiten redirigir esa comprobación de actualización a su servidor de la intranet. Puede utilizar la clave de registro `HKCU\Software\TortoiseSVN\UpdateCheckURL` (valor de cadena) a una URL que apunte a un fichero de texto en su intranet. Ese fichero de texto debe tener el siguiente formato:

```
1.4.1.6000
```

```
¡Hay una nueva versión de TortoiseSVN disponible para descargar!
http://192.168.2.1/downloads/TortoiseSVN-1.4.1.6000-svn-1.4.0.msi
```

La primera línea en ese fichero es la cadena de la versión. Debe asegurarse de que concuerde con la cadena de versión exacta del paquete de instalación de TortoiseSVN. La segunda línea es un texto personalizado, que se muestra en el diálogo de actualización. Puede escribir ahí lo que desee. Pero tenga en cuenta que el espacio en el diálogo de actualización es limitado. ¡Los mensajes demasiado largos se truncarán! La tercera línea es la URL del nuevo paquete de instalación. Esta URL se abre cuando el usuario hace click en la etiqueta del mensaje personalizado en el diálogo de actualización. También puede llevar al usuario a una página web en vez de al fichero MSI directamente. La URL se abre con el navegador web predeterminado, por lo que si especifica una página web, esa página se abre y se muestra al usuario. Si especifica el paquete MSI, el navegador le pedirá al usuario guardar el fichero MSI localmente.

C.3. Estableciendo la variable de entorno `SVN_ASP_DOT_NET_HACK`

Desde la versión 1.4.0, el instalador de TortoiseSVN ya no ofrece al usuario la opción de establecer la variable de entorno `SVN_ASP_DOT_NET_HACK`, dado que eso ha causado muchos problemas y confusiones a algunos usuarios que siempre instalan *todo* sin importar si saben para lo que sirve.

Pero esa opción sólo está oculta para el usuario. Aún puede forzar al instalador de TortoiseSVN para que ponga esa variable de entorno estableciendo la propiedad `ASPDOTNETHACK` a TRUE. Por ejemplo, puede iniciar el instalador así:

```
msiexec /i TortoiseSVN-1.4.0.msi ASPDOTNETHACK=TRUE
```

C.4. Deshabilitar entradas del menú contextual

Empezando en la versión 1.5.0, TortoiseSVN le permite deshabilitar (realmente, ocultar) entradas del menú contextual. Dado que esta característica no debería usarse a la ligera sino únicamente cuando hay una razón necesaria, no hay interfaz de usuario para hacerlo y debe modificarse directamente el registro de Windows. Esto puede utilizarse para deshabilitar ciertos comandos que los usuarios no deberían usar. Pero por favor tenga en cuenta que sólo se ocultan las entradas del menú contextual en el *Explorador*, pero los comandos aún están disponibles por otros medios, por ejemplo la línea de comandos ¡o incluso mediante otros diálogos en el propio TortoiseSVN!

Las claves de registro que almacenan la informacíon sobre qué menús contextuales mostrar son `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow` y `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Cada una de estas entradas de registro es un valor `DWORD`, en el que cada bit corresponde a una entrada de menú específica. Un bit a 1 significa que la entrada de menú correspondiente se desactiva.

Valor	Entrada de menú
0x0000000000000001	Obtener
0x0000000000000002	Actualizar
0x0000000000000004	Confirmar
0x0000000000000008	Añadir
0x0000000000000010	Revertir
0x0000000000000020	Limpieza
0x0000000000000040	Resolver
0x0000000000000080	Cambiar

Valor	Entrada de menú
0x0000000000000100	Importar
0x0000000000000200	Exportar
0x0000000000000400	Crear repositorio aquí
0x0000000000000800	Ramas / Etiqueta
0x0000000000001000	Fusionar
0x0000000000002000	Eliminar
0x0000000000004000	Renombrar
0x0000000000008000	Actualizar a la revisión
0x0000000000010000	Diff
0x0000000000020000	Mostrar registro
0x0000000000040000	Editar conflictos
0x0000000000080000	Relocalizar
0x0000000000100000	Comprobar modificaciones
0x0000000000200000	Ignorar
0x0000000000400000	Navegador de repositorios
0x0000000000800000	Autoría
0x0000000001000000	Crear parche
0x0000000002000000	Aplicar parche
0x0000000004000000	Gráfico de revisiones
0x0000000008000000	Bloqueo
0x0000000010000000	Eliminar bloqueo
0x0000000020000000	Propiedades
0x0000000040000000	Diferenciar con URL
0x0000000080000000	Eliminar ítems no versionados
0x2000000000000000	Configuración
0x4000000000000000	Ayuda
0x8000000000000000	Acerca de

Tabla C.1. Entradas de menú y sus valores

Ejemplo: para deshabilitar las entradas de menú “Relocalizar”, “Eliminar ítems no versionados” y “Configuración”, añada los valores asignados a estas entradas como se muestra:

```
  0x0000000000080000
+ 0x0000000080000000
+ 0x2000000000000000
= 0x2000000080080000
```

El valor `DWORD` inferior (`0x80080000`) debe almacenarse en `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow`, mientras que el valor `DWORD` (`0x20000000`) superior se guarda en `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Para habilitar las entradas de menú de nuevo, simplemente elimine las dos entradas del registro.

Apéndice D. Automatizando TortoiseSVN

Dado que todos los comandos de TortoiseSVN se controlan a través de parámetros de línea de comandos, puede automatizarlo con scripts batch o iniciando comandos y diálogos específicos desde otros programas (por ejemplo, desde su editor de textos favorito).

Importante

Recuerde que TortoiseSVN es un cliente GUI, y esta guía de automatización le muestra cómo conseguir que aparezcan los diálogos de TortoiseSVN para recolectar información del usuario. Si desea escribir un script que no requiera entradas del usuario, debería utilizar el cliente de línea de comandos oficial de Subversion.

D.1. Comandos de TortoiseSVN

El programa de interfaz gráfico de TortoiseSVN se llama `TortoiseProc.exe`. Todos los comandos se especifican con el parámetro `/command:abcd` donde `abcd` es el nombre obligatorio del comando. La mayoría de estos comando necesitan al menos un argumento de ruta, que se proporciona con `/path:"alguna\ruta"`. En la siguiente tabla el comando se refiere al parámetro `/command:abcd` y la ruta se refiere al parámetro `/path:"alguna\ruta"`.

Como algunos comandos pueden tomar una lista de rutas de destino (por ejemplo, si se confirman varios ficheros específicos) el parámetro `/path` puede admitir varias rutas, separadas con un caracter `*`.

TortoiseSVN utiliza ficheros temporales para pasar múltiples argumentos entre la extensión del shell y el programa principal. Desde TortoiseSVN 1.5.0 en adelante, el parámetro `/notempfile` es obsoleto y ya no hay necesidad de añadirlo.

El diálogo de progreso que se utiliza para las confirmaciones, actualizaciones y muchos más comandos normalmente se queda abierto después de que el comando termina, hasta que el usuario pulsa el botón Aceptar. Esto puede cambiarse marcando la opción correspondiente en el diálogo de configuración. Pero utilizando esa opción se cerrará el diálogo de progreso, sin importar si se inició el comando desde el fichero batch o desde el menú contextual de TortoiseSVN.

Para especificar una localización diferente del fichero de configuración, utilice el parámetro `/configdir:"ruta\al\directorio\de\configuración"`. Esto tomará preferencia sobre la ruta por defecto, incluyendo cualquier configuración en el registro.

Para cerrar el diálogo de progreso al final de un comando automáticamente sin utilizar la configuración permanente, puede pasar el parámetro `/closeonend`.

- `/closeonend:0` no cierra el diálogo automáticamente
- `/closeonend:1` auto cerrar si no hay errores
- `/closeonend:2` cierra automáticamente si no hay errores ni conflictos
- `/closeonend:3` cierra automáticamente si no hay errores, conflictos ni fusiones
- `/closeonend:4` cierra automáticamente si no hay errores, conflictos ni fusiones para operaciones locales

La siguiente tabla lista todos los comandos a los que se puede acceder utilizando la línea de comando de TortoiseProc.exe. Como se describe más arriba, se deben utilizar en la forma `/command:abcd`. En la tabla, el prefijo `/command` se omite para ahorrar espacio.

Comando	Descripción
:about	Muestra el diálogo Acerca de. También se muestra si no se pasa ningún comando.
:log	Abre el diálogo de registro. `/path` especifica el fichero o la carpeta para el que se debe mostrar el registro. Se pueden establecer tres opciones adicionales: `/startrev:xxx`, `/endrev:xxx` y `/strict`
:checkout	Abre el diálogo de obtener. `/path` especifica el directorio de destino y `/url` especifica la URL desde la que obtener.
:import	Abre el diálogo de importar. `/path` especifica el directorio con los datos a importar.
:update	Actualiza la copia de trabajo en `/path` hasta HEAD. Si se especifica la opción `/rev` se muestra un diálogo para preguntar al usuario a qué revisión debe actualizarse. Para evitar el diálogo especifique un número de revisión `/rev:1234`. Otras opciones son `/nonrecursive` y `/ignoreexternals`.
:commit	Abre el diálogo de confirmación. `/path` especifica el directorio de destino o la lista de ficheros a confirmar. También puede especificar la opción `/logmsg` para pasar un mensaje de registro predefinido al diálogo de confirmación. O, si no desea pasar el mensaje de registro en la línea de comandos, utilice `/logmsgfile:ruta`, donde `ruta` apunta a un fichero que contiene el mensaje de registro. Para rellenar la caja de texto del ID del error (en caso de que haya preparado adecuadamente la integración con el sistema de control de errores), puede utilizar `/bugid:"el ID del error aquí"`.
:add	Añade los ficheros en `/path` al control de versiones.
:revert	Revierte las modificaciones locales de una copia de trabajo. La `/path` dice qué items deben revertirse.
:cleanup	Hace limpieza tras operaciones interrumpidas o abortadas, y desbloquea la copia de trabajo en `/path`.
:resolve	Marca un fichero en conflicto especificado en `/path` como resuelto. Si se especifica `/noquestion`, la resolución se realiza sin preguntar al usuario primero si realmente debe hacerse.
:repocreate	Crea un repositorio en `/path`
:switch	Abre el diálogo cambiar. `/path` especifica el directorio de destino.
:export	Exporta la copia de trabajo en `/path` a otro directorio. Si `/path` apunta a un directorio sin versionar, aparecerá un diálogo pidiendo una URL para exportar al directorio en `/path`.
:merge	Abre el diálogo de fusión. `/path` especifica el directorio de destino. Para fusionar un rango de revisiones, están disponibles las siguientes opciones: `/fromurl:URL`, `/revrange:cadena`. Para fusionar dos árboles de repositorio, están disponibles las siguientes opciones: `/fromurl:URL`, `/tourl:URL`, `/fromrev:xxx` y `/torev:xxx`. Estas opciones pre-rellenan los campos relevantes en el diálogo de fusión.
:mergeall	Abre el diálogo fusionar todo. `/path` especifica el directorio de destino.
:copy	Abre el diálogo de rama / etiqueta. `/path` especifica la copia de trabajo desde la que crear la rama o etiqueta. Y `/url` es la URL de destino. También puede especificar la opción `/logmsg` para pasar un mensaje de registro predefinido al diálogo de rama / etiqueta. O, si no desea pasar el mensaje de registro en la línea de comandos, utilice `/logmsgfile:ruta`, donde `ruta` apunta a un fichero que contiene el mensaje de registro.

Comando	Descripción
:settings	Abre el diálogo de configuración.
:remove	Elimina el/los fichero/s en `/path` del control de versiones.
:rename	Renombra el fichero en `/path`. El nuevo nombre del fichero se pregunta con un diálogo. Para evitar la pregunta sobre renombrar ficheros similares en un paso, utilice `/noquestion`.
:diff	Inicia el programa externo de diferencias especificado en la configuración de TortoiseSVN. `/path` especifica el primer fichero. Si se establece la opción `/path2`, se inicia el programa de diferencias con esos dos ficheros. Si se omite `/path2`, se hace la comparación entre el fichero de `/path` y su BASE. Para establecer explícitamente los números de revisión utilice `/startrev:xxx` y `/endrev:xxx`. Si `/blame` se establece y `/path2` no se establece, entonces la comparación se efectúa tras realizar la autoría de los ficheros entre las revisiones dadas.
:showcompare	Dependiendo de las URLs y las revisiones a comparar, esto o bien muestra un diff unificado (si la opción `unified` está establecida), o un diálogo con una lista de ficheros que han cambiado o, en el caso de que las URLs apunten a ficheros, inicia el visor de diferencias para esos dos ficheros. Las opciones `url1`, `url2`, `revision1` y `revision2` deben especificarse. Las opciones `pegrevision`, `ignoreancestry`, `blame` y `unified` son opcionales.
:conflicteditor	Inicia el editor de conflictos especificado en la configuración de TortoiseSVN con los ficheros correctos para el fichero en conflicto de `/path`.
:relocate	Abre el diálogo relocalizar. `/path` especifica la ruta de la copia de trabajo a relocalizar.
:help	Abre el fichero de ayuda.
:repostatus	Abre el diálogo Comprobar modificaciones. La `/path` especifica el directorio de la copia de trabajo.
:repobrowser	Inicia el diálogo del navegador de repositorios, apuntando a la URL de la copia de trabajo dada en `/path` o `/path` apunta directamente a una URL. Puede utilizarse una opción adicional `/rev:xxx` para especificar la revisión que el navegador de repositorios debe mostrar. Si se omite `/rev:xxx`, se supone HEAD. Si `/path` apunta a una URL, el `/projectpropertiespath:ruta/a/copia/trabajo` especifica la ruta desde donde hay que leer y utilizar las propiedades del proyecto.
:ignore	Añade todos los objetivos en `/path` a la lista de ignorados, es decir, añade la propiedad `svn:ignore` a esos ficheros.
:blame	Abre el diálogo de autoría para el fichero especificado en `/path`. Si las opciones `/startrev` y `/endrev` se establecen, entonces el diálogo preguntando por el rango de la autoría no se muestra y se utilizan los valores de revisión especificados. Si la opción `/line:nnn` se establece, TortoiseBlame se abrirá mostrando el número de línea especificado. Las opciones `/ignoreeol`, `/ignorespaces` y `/ignoreallspaces` también se soportan.

Comando	Descripción
:cat	Graba un fichero de una URL o una copia de trabajo dada en `/path` al lugar dado en `/savepath:ruta`. La revisión se da en `/revision:xxx`. Esto puede utilizarse para obtener un fichero cno una revisión concreta.
:createpatch	Crea un fichero de parte para la ruta dada en `/path`.
:revisiongraph	Muestra el gráfico de revisiones para la ruta dada en `/path`.
:lock	Bloquea un fichero o todos los ficheros en un directorio dado en `/path`. Se muestra el diálogo Bloquear para que el usuario pueda introducir un comentario para el bloqueo.
:unlock	Desbloquea un fichero o todos los ficheros en un directorio dado en `/path`.
:rebuildiconcache	Reconstruye la caché de iconos de Windows. Utilice esto sólo en el caso de que los iconos de Windows se hayan corrompido. Un efecto secundario de esto (y no puede evitarse) es que los iconos del escritorio se recolocan. Para eliminar el mensaje de advertencia, pase `/noquestion`.
:properties	Muestra el diálogo de propiedades para la ruta dada en `/path`.

Tabla D.1. Lista de comandos y opciones disponibles

Ejemplos (que deben introducirse en una única línea):

```
TortoiseProc.exe /command:commit
                 /path:"c:\svn_wc\file1.txt*c:\svn_wc\file2.txt"
                 /logmsg:"mensaje de registro de prueba" /closeonend

TortoiseProc.exe /command:update /path:"c:\svn_wc\" /closeonend

TortoiseProc.exe /command:log /path:"c:\svn_wc\file1.txt"
                 /startrev:50 /endrev:60 /closeonend
```

D.2. Comandos de TortoiseIDiff

La herramienta de diferencia de imágenes tiene algunas pocas opciones de comando que puede utilizar para controlar cómo se inicia la herramienta. El programa se llama `TortoiseIDiff.exe`.

La siguiente tabla contiene todas las opciones que pueden pasarse a la herramienta de diferencia de imágenes por la línea de comandos.

Opción	Descripción
:left	Ruta al fichero mostrado a la izquierda.
:lefttitle	Una cadena de título. Esta cadena se utiliza en el título de la vista de la imagen en lugar de la ruta completa al fichero de la imagen.
:right	Ruta al fichero mostrado a la derecha.
:righttitle	Una cadena de título. Esta cadena se utiliza en el título de la vista de la imagen en lugar de la ruta completa al fichero de la imagen.
:overlay	Si se especifica, la herramienta de diferencias de imágenes cambia al modo de sobreimpresión (fundido alpha).
:fit	Si se especifica, la herramienta de diferencias de imágenes ajusta ambas imágenes juntas.
:showinfo	Muestra la caja de información de imágen.

Tabla D.2. Lista de las opciones disponibles

Ejemplo (que debería introducirse como una única línea):

```
TortoiseIDiff.exe /left:"c:\images\img1.jpg" /lefttitle:"imágen 1"
                  /right:"c:\images\img2.jpg" /righttitle:"imágen 2"
                  /fit /overlay
```

Apéndice E. Referencia cruzada del interface de línea de comandos

A veces este manual hace referencia a la documentación principal de Subversion, que describe Subversion en términos del Interfaz de Línea de Comandos (CLI). Para ayudarle a entener lo que TortoiseSVN hace tras el escenario, hemos compilado una lista mostrando los comandos CLI equivalentes a cada operación en la Interfaz de Usuario de TortoiseSVN.

> **Nota**
>
> Incluso aunque hay equivalentes en CLI para lo que TortoiseSVN hace, recuerde que TortoiseSVN *no* llama al CLI sino que utiliza la librería de Subversion directamente.

Si cree que ha encontrado un error en TortoiseSVN, puede que le pidamos que intente reproducirlo utilizando el CLI, para que podamos distinguir los problemas de TortoiseSVN de los problemas de Subversion. Esta referencia le dice qué comando debe probar.

E.1. Convenciones y reglas básicas

En las descripciones que aparecen a continuación, la URL con el lugar de un repositorio se muestra simplemente como `URL`, que podría corresponder por ejemplo a `http://tortoisesvn.googlecode.com/svn/trunk`. La ruta de la copia de trabajo se muestra simplemente como `RUTA`, que podría corresponder por ejemplo a `C:\TortoiseSVN\trunk`.

> **Importante**
>
> Dado que TortoiseSVN es una Extensión del Shell de Windows, no es capaz de utilizar la noción de directorio de trabajo actua. Todas las rutas de las copias de trabajo se deben especificar con rutas absolutas, no relativas.

Algunos ítems son opcionales, y a menudo se controlan con casillas o botones de opciones en TortoiseSVN. Estas opciones se muestran [entre corchetes] en las definiciones de la línea de comandos.

E.2. Comandos de TortoiseSVN

E.2.1. Obtener

```
svn checkout [-N] [--ignore-externals] [-r rev] URL RUTA
```

Si se marca Sólo obtener la carpeta superior, utilice la opción `-N`.

Si se marca Omitir externos, utilice la opción `--ignore-externals`.

Si está obteniendo una revisión en concreto, especifíquela detrás de la URL utilizando la opción `-r`.

E.2.2. Actualizar

```
svn info URL_de_la_copia_de_trabajo
svn update [-r rev] RUTA
```

A día de hoy, la actualización no es una operación atómica en Subversion. Por tanto, TortoiseSVN primero encuentra la revisión HEAD del repositorio, y luego actualiza todos los ítems a ese número de revisión en concreto, evitando crear así una copia de trabajo mezclada.

Si sólo se selecciona un único ítem para actualizar, o si los ítems seleccionados no son todos del mismo repositorio, TortoiseSVN simplemente actualiza hasta HEAD.

Aquí no se utilizan opciones en la línea de comandos. Actualizar a la Revisión también implementa el comando actualizar, pero ofrece más opciones.

E.2.3. Actualizar a la revisión

```
svn info URL_de_la_copia_de_trabajo
svn update [-r rev] [-N] [--ignore-externals] RUTA
```

Si se marca Sólo obtener la carpeta superior, utilice la opción `-N`.

Si se marca Omitir externos, utilice la opción `--ignore-externals`.

E.2.4. Confirmar

En TortoiseSVN, el diálogo de confirmación utiliza varios comandos Subversion. La primera fase es una comprobación del estado que determina los ítems de su copia de trabajo que potencialmente pueden ser confirmados. Puede revisar la lista, comparar los ficheros contra BASE y seleccionar los ítems que desea incluir en la confirmación.

```
svn status -v RUTA
```

Si se marca Mostrar archivos no versionados, TortoiseSVN le mostrará también todos los ficheros y carpetas no versionados en su jerarquía de la copia de trabajo, teniendo en cuenta las reglas de ignorar. Esta característica en concreto no tiene equivalente directo en Subversion, ya que el comando `svn status` no entra en carpetas no versionadas.

Si selecciona cualquier fichero o carpeta no versionado, esos ítems primero se añadirán a su copia de trabajo.

```
svn add RUTA...
```

Cuando pulsa Aceptar, tiene lugar la confirmación en Subversion. Si ha dejado todas las casillas de selección de ficheros sin tocar tal y como aparecieron, TortoiseSVN utilizará una única confirmación recursiva en la copia de trabajo. Si ha deseleccionado algún fichero, entonces se debe utilizar una confirmación no-recursiva (`-N`), y se debe especificar individualmente cada ruta en la línea de comandos de la confirmación.

```
svn commit -m "Mensaje de registro" [-N] [--no-unlock] RUTA...
```

`Mensaje de registro` aquí representa los contenidos del cuadro de texto Mensaje de Registro. Puede estar vacío.

Si se marca Mantener Bloqueos, utilice la opción `--no-unlock`.

E.2.5. Diff

```
svn diff RUTA
```

Si utiliza Diff desde el menú contextual principal, está comparando un fichero modificado contra su revisión BASE. La salida del comando CLI anterior también hace ésto y produce la salida en formato diff unificado. Sin embargo, esto no es lo que TortoiseSVN utiliza. TortoiseSVN utiliza TortoiseMerge

(o un programa de comparación de su elección) para mostrar las diferencias visualmente entre ficheros de texto completos, por lo que no hay equivalente directo en el CLI.

También puede comparar 2 ficheros cualquiera utilizando TortoiseSVN, estén o no bajo el control de versiones. TortoiseSVN simplemente alimenta con los dos ficheros al programa de diferencias y le deja que él encuentre dónde están las diferencias.

E.2.6. Mostrar registro

```
svn log -v -r 0:N --limit 100 [--stop-on-copy] RUTA
  o
svn log -v -r M:N [--stop-on-copy] RUTA
```

Por defecto, TortoiseSVN intenta recoger 100 mensajes de registro utilizando el método --limit. Si la configuración establece que utilice las APIs antiguas, se usa la segunda forma para obtener los mensajes de registro de 100 revisiones del repositorio.

Si se marca Parar en copia/renombrado, utilice la opción `--stop-on-copy`.

E.2.7. Comprobar modificaciones

```
svn status -v RUTA
  o
svn status -u -v RUTA
```

La comprobación de estado inicialmente sólo muestra su copia de trabajo. Si pulsa en Comprobar repositorio, se comprueba también el repositorio para ver qué ficheros cambiarían tras una actualización, lo que necesita la opción `-u`.

Si se marca Mostrar archivos no versionados, TortoiseSVN le mostrará también todos los ficheros y carpetas no versionados en su jerarquía de la copia de trabajo, teniendo en cuenta las reglas de ignorar. Esta característica en concreto no tiene equivalente directo en Subversion, ya que el comando `svn status` no entra en carpetas no versionadas.

E.2.8. Gráfico de revisión

El gráfico de revisiones es una característica única de TortoiseSVN. No hay equivalente en el cliente de línea de comandos.

Lo que hace TortoiseSVN es

```
svn info URL_de_la_copia_de_trabajo
svn log -v URL
```

donde URL es la *raíz* del repositorio, y luego analiza los datos devueltos.

E.2.9. Navegador de repositorios

```
svn info URL_de_la_copia_de_trabajo
svn list [-r rev] -v URL
```

Puede utilizar `svn info` para determinar la raíz del repositorio, que es el nivel superior mostrado en el navegador de repositorios. No puede navegar `más arriba` de ese nivel. Además, este comando devuelve toda la información sobre bloqueos que se muestra en el navegador de repositorios.

La llamada `svn list` le mostrará los contenidos de un directorio, cuando se proporciona una URL y una revisión.

E.2.10. Editar conflictos

Este comando no tiene equivalente CLI. Invoca TortoiseMerge o una herramienta externa de fusión/comparación en 3 paneles para mirar los ficheros en conflictos y ver qué líneas hay que utilizar.

E.2.11. Resuelto

```
svn resolved RUTA
```

E.2.12. Renombrar

```
svn rename RUTA_ACTUAL NUEVA_RUTA
```

E.2.13. Eliminar

```
svn delete RUTA
```

E.2.14. Revertir

```
svn status -v RUTA
```

La primera fase es una comprobación del estado que determina los ítems en su copia de trabajo que pueden ser revertidos potencialmente. Puede revisar la lista, comparar los ficheros contra BASE y seleccionar los ítems que desea incluir en la operación de revertir.

Cuando pulsa Aceptar, tiene lugar la operación de revertir en Subversion. Si ha dejado todas las casillas de selección de ficheros sin tocar tal y como aparecieron, TortoiseSVN utilizará una única operación revertir recursivamente (`-R`) en la copia de trabajo. Si ha deseleccionado algún fichero, entonces se debe especificar individualmente cada ruta en la línea de comandos de la operación de revertir.

```
svn revert [-R] RUTA...
```

E.2.15. Limpieza

```
svn cleanup RUTA
```

E.2.16. Obtener bloqueo

```
svn status -v RUTA
```

La primera fase es una comprobación del estado que determina los ficheros en su copia de trabajo que pueden ser potencialmente bloqueados. Puede seleccionar los ítems que desea bloquear.

```
svn lock -m "Mensaje de bloqueo" [--force] RUTA...
```

`Mensaje de bloqueo` aquí representa los contenidos del cuadro de texto Mensaje de bloqueo. Puede estar vacío.

Si se marca Robar los bloqueos, utilice la opción `--force`.

E.2.17. Quitar bloqueo

```
svn unlock RUTA
```

E.2.18. Ramas / Etiqueta

```
svn copy -m "Mensaje de registro" URL URL
  o
svn copy -m "Mensaje de registro" URL@rev URL@rev
  o
svn copy -m "Mensaje de registro" RUTA URL
```

El diálogo Rama/Etiqueta hace una copia en el repositorio. Hay 3 botones de opciones:

- Revisión HEAD en el repositorio
- Revisión específica en el repositorio
- Copia de trabajo

que corresponden a las 3 variantes de línea de comandos anteriores.

`Mensaje de registro` aquí representa los contenidos del cuadro de texto Mensaje de Registro. Puede estar vacío.

E.2.19. Cambiar

```
svn info URL_de_la_COPIA_DE_TRABAJO
svn switch [-r rev] URL RUTA
```

E.2.20. Fusionar

```
svn merge [--dry-run] --force URL_Origen@revN URL_Destino@revM RUTA
```

Probar fusión realiza la misma fusión con la opción `--dry-run`.

```
svn diff URL_Origen@revN URL_Destino@revM
```

Diff unificado muestra la operación de diferenciación que se utilizará para hacer la fusión.

E.2.21. Exportar

```
svn export [-r rev] [--ignore-externals] URL RUTA_Exportación
```

Esta forma se utiliza cuando se accede desde una carpeta sin versionar, y la carpeta se utiliza como el destino.

La exportación de una copia de trabajo a otro lugar se hace sin utilizar las bibliotecas de Subversion, por lo que no hay línea de comandos equivalente.

Lo que TortoiseSVN hace es copiar todos los ficheros al nuevo destino mientras le muestra el progreso de la operación. Opcionalmente, pueden exportarse también los ficheros y/o carpetas sin versionar.

En ambos casos, si se marca Omitir externos, utilice la opción `--ignore-externals`.

E.2.22. Relocalizar

```
svn switch --relocate URL_Origen URL_Destino
```

E.2.23. Crear repositorio aquí

```
svnadmin create --fs-type fsfs RUTA
```

E.2.24. Añadir

```
svn add RUTA...
```

Si había seleccionado una carpeta, en primer lugar TortoiseSVN la escanea recursivamente en busca de ítems que puedan ser añadidos.

E.2.25. Importar

```
svn import -m "Mensaje de registro" RUTA URL
```

`Mensaje de registro` aquí representa los contenidos del cuadro de texto Mensaje de Registro. Puede estar vacío.

E.2.26. Autoría

```
svn blame -r N:M -v RUTA
svn log -r N:M RUTA
```

Si utiliza TortoiseBlame para ver la información de autoría, se necesita también el fichero de registro para mostrar los mensajes de registro en un texto de ayuda. Si visualiza la autoría como un fichero de texto, no se necesita esta información.

E.2.27. Añadir a la lista de ignorados

```
svn propget svn:ignore RUTA > ficherotemporal
{editar los nuevos ítems a ignorar en el fichero ficherotemporal}
svn propset svn:ignore -F ficherotemporal RUTA
```

Dado que `svn:ignore` a menudo tiene un valor con varias líneas, aquí se muestra cómo se cambiaría utilizando un fichero de texto mejor que hacerlo directamente en la línea de comandos.

E.2.28. Crear parche

```
svn diff RUTA > fichero-parche
```

TortoiseSVN crea un fichero de parche en formato diff unificado comparando la copia de trabajo con su versión BASE.

E.2.29. Aplicar parche

Aplicar parches es un trabajo complicado a no ser que el parche y la copia de trabajo estén en la misma revisión. Afortunadamente, puede utilizar TortoiseMerge, que no tiene equivalente directo en Subversion.

Apéndice F. Detalles de implementacion

Este apéndice contiene una discusión más detallada sobre la implementación de algunas de las características de TortoiseSVN.

F.1. Iconos sobreimpresionados

Cada fichero y carpeta tiene un valor del estado de Subversion tal y como lo devuelve la librería de Subversion. En el cliente de línea de comandos, estos valores se representan por códigos de una única letra, pero en TortoiseSVN se muestran gráficamente utilizando los iconos sobreimpresionados. Dado que el número de sobreimpresiones es muy limitado, cada sobreimpresión puede representar uno de varios valores de estado.

La sobreimpresión *En conflicto* se utiliza para representar el estado `en conflicto`, donde una actualización o un cambio han resultado en conflictos entre los cambios locales y los cambios descargados desde el repositorio. También se utiliza para indicar el estado `obstruído`, que puede ocurrir cuando una operación no puede terminar.

La sobreimpresión *Modificado* representa el estado `modificado`, donde ha hecho modificaciones locales; el estado `fusionado`, donde los cambios del repositorio se han fusionado con sus cambios locales; y el estado `reemplazado`, donde un fichero ha sido eliminado y reemplazado por otro diferente con el mismo nombre.

La sobreimpresión *Eliminado* representa el estado `eliminado`, donde un ítem está marcado para su eliminación; o el estado `faltante`, donde un ítem no está presente. Naturalmente un ítem que está faltante no puede tener una sobreimpresión sobre si mismo, pero la carpeta padre puede marcarse si uno de sus ítems hijo falta.

La sobreimpresión *Añadido* se utiliza simplemente para representar el estado `añadido` cuando un ítem ha sido añadido al control de versiones.

La sobreimpresión *En Subversion* se utiliza para representar un ítem que está en el estado `normal`, o un ítem versionado cuyo estado aún se desconoce. dado que TortoiseSVN utiliza un proceso de caché en segundo plano para obtener los estados, puede llevar unos pocos segundos antes deque se actualice la sobreimpresión.

La sobreimpresión *Necesita bloqueo* se utiliza para indicar cuándo un fichero tiene la establecida la propiedad `svn:needs-lock`. Para las copias de trabajo que fueron creadas utilizando Subversion 1.4.0 y posteriores, el estado `svn:needs-lock` se guarda en la caché local por Subversion y se utiliza para determinar cuándo mostrar esta sobreimpresión. Para copias de trabajo que están en formato pre-1.4.x,

TortoiseSVN muestra esta sobreimpresión cuando el fichero tiene el estado de sólo-lectura. Tenga en cuenta que Subversion automáticamente actualiza las copias de trabajo cuando las actualiza, aunque el almacenamiento en caché de la propiedad `svn:needs-lock` puede no realizarse hasta que el propio fichero se actualice.

La sobreimpresión *Bloqueado* se utiliza cuando la copia de trabajo local contiene un bloqueo para ese fichero.

La sobreimpresión *Ignorado* se utiliza para representar un ítem que está en el estado `ignorado`, bien por un patrón global de ignorado, o por la propiedad `svn:ignore` de la carpeta padre. Esta sobreimpresión es opcional.

La sobreimpresión *No versionado* se utiliza para representar a un ítem que está en el estado `no versionado`. Este es un ítem dentro de una carpeta versionada, pero que él mismo no está bajo el control de versiones. Esta sobreimpresión es opcional.

Si un ítem tiene el estado de Subversion `ninguno` (el ítem no está dentro de una copia de trabajo), entonces no se mostrará ninguna sobreimpresión. Si ha escogido deshabilitar las sobreimpresiones *Ignorado* y *Sin versionar* entonces tampoco se mostrarán las sobreimpresiones para estos ficheros.

Un ítem sólo puede tener un valor de estado de Subversion. Por ejemplo un fichero puede estar modificado localmente y puede estar marcado para su eliminación al mismo tiempo. Subversion devuelve un único valor de estado - en este caso `eliminado`. Estas prioridades están definidas en el propio Subversion.

Cuando TortoiseSVN muestra el estado recursivamente (la configuración por defecto), cada carpeta muestra una sobreimpresión reflejando su propio estado y el estado de todos sus hijos. Para mostrar una única sobreimpresión *resumen*, utilizamos el orden de prioridad mostrado arriba para determinar qué sobreimpresión utilizar, tomando la sobreimpresión *En conflicto* como la que tiene mayor prioridad.

De hecho, puede que se encuentre con que no todos estos iconos se utilizan en su sistema. Esto se debe a que el número de sobreimpresiones permitidas por Windows está limitado a 15. Windows utiliza 4 de ellas, y las 11 restantes pueden ser utilizados por otras aplicaciones. Si no hay suficientes espacios para sobreimpresiones disponibles, TortoiseSVN intenta ser un “Buen Ciudadano (TM)” y limita su uso de sobreimpresiones para darles una oportunidad al resto de aplicaciones.

- *Normal*, *Modificado* y *En conflicto* siempre se cargan y están visibles.
- *Borrado* se carga si es posible, pero se cambia por *Modificado* si no hay suficientes huecos.
- *Sólo-lectura* se carga si es posible, pero se cambia por *Normal* si no hay suficientes huecos.
- *Bloqueado* se carga sólo si hay menos de 13 sobreimpresiones ya cargadas. Se cambia por *Normal* si no hay suficientes huecos.
- *Añadido* se carga sólo si hay menos de 14 sobreimpresiones ya cargadas. Se cambia por *Modificado* si no hay suficientes huecos.

Apéndice G. Asegurando Svnserve utilizando SSH

Esta sección proporciona una guía paso-a-paso para configurar Subversion y TortoiseSVN para utilizar el protocolo `svn+ssh`. Si ya está utilizando conexiones SSH autenticadas para identificarse en su servidor, entonces esto ya lo tiene hecho y puede encontrar más detalles en el libro de Subversion. Si no está utilizando SSH pero le gustaría hacerlo para proteger su instalación de Subversion, esta guía le proporciona un método simple con el que no necesita crear una cuenta de usuario SSH diferente en el servidor para cada usuario de Subversion.

En esta implementación creamos una única cuenta de usuario SSH para todos los usuarios de Subversion, y utilizamos diferentes claves de autenticación para diferenciar entre los usuarios reales de Subversion.

En este apéndice asumimos que ya tiene las herramientas de Subversion instaladas, y que ha creado un repositorio tal y como se describe en alguna sección de este manual. Tenga en cuenta que *no* debería iniciar svnserve como un servicio o demonio cuando se utilice con SSH.

Mucha de esta información viene de un tutorial proporcionado por Marc Logemann, que puede encontrarse en *www.logemann.org* [http://www.logemann.org/2007/03/13/subversion-tortoisesvn-ssh-howto/] Thorsten Müller ha proporcionado información adicional sobre cómo configurar un servidor Windows. ¡Gracias tíos!

G.1. Preparando un servidor Linux

Necesita tener SSH habilitado en el servidor, y aquí asumimos que utilizará OpenSSH. En la mayoría de distribuciones ya lo tendrá instalado. Para averiguarlo, escriba:

```
ps xa | grep sshd
```

y busque trabajos de ssh.

Un punto a tener en cuenta es que si compila Subversion desde el código fuente y no proporciona ningún argumento a `./configure`, Subversion crea un directorio `bin` bajo `/usr/local` y pone sus binarios ahí. Si desea utilizar el modo túnel con SSH, debe asegurarse de que el usuario que haga login via SSH necesita ejecutar el programa svnserve y algunos otros binarios. Por esta razón, o bien ponga `/usr/local/bin` en la variable `PATH`, cree enlaces simbólicos de sus binarios en el directorio `/usr/sbin` o en cualquier otro directorio que esté incluido en el `PATH`.

Para comprobar que todo está bien, haga login como el usuario de destino con SSH y escriba:

```
which svnserve
```

Este comando le dirá si se puede alcanzar svnserve.

Cree un nuevo usuario que utilizaremos para acceder al repositorio svn:

```
useradd -m svnuser
```

Asegúrese de otorgar a este usuario todos los derechos de acceso al repositorio.

G.2. Preparando un servidor Windows

Instale el demonio Cygwin SSH como se describe aquí: *http://pigtail.net/LRP/printsrv/cygwin-sshd.html*

Cree una nueva cuenta de usuario de Windows `svnuser` que utilizaremos para acceder al repositorio. Asegúrese de otorgar a este usuario todos los derechos de acceso al repositorio.

Si aún no hay fichero de contraseñas, cree uno desde la consola Cygwin utilizando:

```
mkpasswd -l > /etc/passwd
```

G.3. Herramientas de cliente SSH para utilizar con TortoiseSVN

Obtenga las herramientas que necesitamos para utilizar SSH desde el cliente Windows desde este sitio: *http://www.chiark.greenend.org.uk/~sgtatham/putty/* Simplemente vaya a la sección de descargas y obtenga `Putty`, `Plink`, `Pageant` y `Puttygen`.

G.4. Creando certificados OpenSSH

El siguiente paso es crear un par de claves para la autenticación. Hay dos formas posibles para crear las claves. La primera es crear las claves con PuTTYgen en el cliente, subir la clave pública a su servidor y utilizar la clave privada con PuTTY. La otra es crear el par de claves con la herramienta de OpenSSH ssh-keygen, descargar la clave privada a su cliente y convertir esa clave privada en una clave privada al estilo de PuTTY.

G.4.1. Crear claves utilizando ssh-keygen

Haga login en el servidor como `root` o `svnuser`y escriba:

```
ssh-keygen -b 1024 -t dsa -N contraseña -f ficherodeclave
```

utilizando una frase-clave real (y que nadie más conozca) y un nombre de fichero de clave. Acabamos de crear una clave SSH2 DSA con una frase-clave de 1024 bit. Si escribe

```
ls -l ficherodeclave*
```

verá dos ficheros, `ficherodeclave` y `ficherodeclave.pub`. Como puede suponer, el fichero `.pub` es el fichero de la clave pública, y el otro es el de la clave privada.

Añada la clave pública a las que ya están en la carpeta `.ssh` dentro del directorio home del usuario `svnuser`:

```
cat ficherodeclave.pub >> /home/svnuser/.ssh/authorized_keys
```

Para utilizar la clave privada que hemos generado, tenemos que convertirla a un formato PuTTY, ya que el formato del fichero de clave privada no está especificado por un cuerpo estándar. Tras descargar el fichero de clave privada a su PC cliente, inicie PuTTYgen y utilice Conversiones → Importar clave. Navegue a su fichero `ficherodeclave` que ha obtenido desde el servidor e introduzca la frase-clave que utilizó cuando creó la clave. Finalmente pulse en Guardar clave privada y guarde el fichero como `ficherodeclave.PPK`.

G.4.2. Crear claves utilizando PuTTYgen

Utilice PuTTYgen para generar un par de clave pública/clave privada y guardarla. Copie la clave pública al servidor y añádala a las que ya están en la carpeta `.ssh` del directorio home del usuario `svnuser`:

```
cat ficherodeclave.pub >> /home/svnuser/.ssh/authorized_keys
```

G.5. Comprobación utilizando PuTTY

Para comprobar la conexión utilizaremos PuTTY. Inicie el programa y en la pestaña Session establezca el nombre del host al nombre o dirección IP de su servidor, el protocolo a SSH y guardar sesión como

a ConexionSvn o con el nombre que prefiera. En la pestaña SSH establezca la versión preferida del protocolo SSH a 2 y en Auth establezca la ruta completa al fichero de clave privada .PKK que convirtió antes. Vuelva a la pestaña Session y pulse el botón Save. Ahora verá ConexionSvn en la lista de sesiones almacenadas.

Pulse en Abrir y debería ver un prompt de login al estilo telnet. Utilice svnuser como el nombre de usuario y si todo va bien debería conectarse directamente sin que se le pregunte por una contraseña.

Puede que necesite editar /etc/sshd_config en el servidor. Cambie las lineas como sigue y después reinicie el servicio SSH.

```
PubkeyAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication no
```

G.6. Comprobando SSH con TortoiseSVN

Por ahora sólo hemos comprobado que puede hacer login utilizando SSH. Ahora debemos asegurarnos de que la conexión SSH realmente puede ejecutar svnserve. En el servidor modifique /home/svnuser/.ssh/authorized_keys para permitir que varios autores de Subversion utilicen la misma cuenta del sistema, svnuser. Tenga en cuenta que cada autor de Subversion utiliza el mismo login pero una clave de autenticación diferente, por lo tanto deberá añadir una línea por cada autor.

Nota: Todo esto es una única línea muy larga.

```
command="svnserve -t -r <RutaRaizRepositorios> --tunnel-user=<autor>",
            no-port-forwarding,no-agent-forwarding,no-X11-forwarding,
            no-pty ssh-rsa <ClavePublica> <Comentario>
```

Hay varios valores que necesitará establecer de acuerdo a su configuración.

<RutaRaizRepositorios> debería reemplazarse con la ruta al directorio que contiene sus repositorios. Esto evita que tenga que especificar las rutas completas del servidor dentro de las URLs. Tenga en cuenta que debe utilizar barras de dividir incluso en servidores Windows, por ejemplo, c:/svn/raizrepo. En los ejemplos siguientes asumiremos que tiene una carpeta de repositorio dentro de la raíz de repositorios llamada repos.

<autor> debería reemplazarse con el autor SVN que desea que se almacene en la confirmación. Esto también permite que svnserve utilice sus derechos de acceso dentro de svnserve.conf.

<ClavePublica> debería reemplazarse con la clave pública que generó anteriormente.

<Comentario> puede ser cualquier comentario que desee, aunque es útil para asociar un nombre de autor SVN con el nombre real de la persona.

Haga click con el botón derecho sobre cualquier carpeta en el Explorador de Windows y seleccione TortoiseSVN → Navegador de repositorios. Se le pedirá introducir una URL, así que introduzca una de esta forma:

```
svn+ssh://svnuser@ConexionSvn/repos
```

¿Qué significa esa URL? El nombre de esquema es svn+ssh lo que indica a TortoiseSVN cómo manejar las peticiones al servidor. Después de la doble barra, especifica el usuario con el que conectarse al servidor, en nuestro caso svnuser. Después de la @ proporcionamos nuestro nombre de sesión PuTTY. Este nombre de sesión contiene todos los detalles como dónde encontrar la clave privada y la dirección

IP o DNS del servidor. Por último debemos proporcionar la ruta al repositorio, relativa a la raíz del repositorio en el servidor, como se especifica en el fichero `authorized_keys`.

Pulse en Aceptar y debería ser capaz de navegar por el contenido del repositorio. Si es así, ahora tiene ejecutándose un túnel SSH con TortoiseSVN.

Tenga en cuenta que por defecto TortoiseSVN utiliza su propia versión de Plink para conectarse. Esto evita que aparezca una ventana de consla para cada intento de autenticación, pero también significa que no hay ningún lugar donde aparezcan los mensajes de error. Si recibe el error “No es posible escribir en la salida estándar”, puede intentar especificar Plink como el cliente en la configuración de red de TortoiseSVN. Esto le permitirá ver el mensajes de error real generado por Plink.

G.7. Variantes de configuración SSH

Una forma de simplificar la URL en TortoiseSVN es almacenar el usuario dentro de la sesión PuTTY. Para esto deberá cargar su sesión ya definida `ConexionSvn` en PuTTY y en la pestaña Conexión establezca Usuario para auto login con el nombre de usuario, por ejemplo `svnuser`. Guarde su sesión PuTTY como antes e intente la siguiente URL dentro de TortoiseSVN:

```
svn+ssh://ConexionSvn/repos
```

Esta vez sólo proporcionamos el nombre de la sesión PuTTY `ConexionSvn` al cliente SSH que utiliza TortoiseSVN (TortoisePlink.exe). Este cliente buscará en la sesión todos los detalles necesarios.

En el momento de escribir este documento PuTTY no comprueba todas las configuraciones almacenadas, por lo que si tiene múltiples configuraciones con el mismo nombre de servidor, elegirá la primera que concuerde. Además, si edita la configuración por defecto y la guarda, el nombre de usuario para el login automático *no* se almacena.

A mucha gente le gusta utilizar Pageant para almacenar todas sus claves. Dado que una sesión PuTTY puede almacenar una clave, no siempre necesitará Pageant. Pero imagine que quiere almacenar claves para diferentes servidores; en este caso necesitaría editar todas las sesiones PuTTY una y otra vez, dependiendo del servidor al que se intente conectar. En esta situación utilizar Pageant tiene sentido, porque cuando PuTTY, Plink, TortoisePlink o cualquier otra herramienta basada en PuTTY intente conectarse a un servidor SSH, comprueba todas las claves privadas que Pageant mantiene para iniciar la conxión.

Para esta tarea, simplemente ejecute Pageant y añada la clave privada. Debe ser la misma clave privada que definió en la sesión PuTTY anteriormente. Si utiliza Pageant para el almacenamiento de claves privadas, puede eliminar la referencia al fichero de clave privada en su sesión PuTTY almacenada. Por supuesto que puede añadir más claves para otros servidores, o para otros usuarios.

Si no quiere repetir este procedimiento tras cada reinicio de su cliente, entonces debería colocar Pageant en el grupo de auto-inicio de su instalación de Windows. Puede añadir las claves con las rutas completas como argumentos de línea de comandos de Pageant.exe

La última forma de conectarse con un servidor SSH es simplemente utilizando esta URL dentro de TortoiseSVN:

```
svn+ssh://svnuser@100.101.102.103/repos
svn+ssh://svnuser@midominio.com/repos
```

Como puede ver, no utilizamos una sesión almacenada PuTTY sino una dirección IP (o nombre de dominio) como objetivo de la conexión. También proporcionamos el usuario, pero puede que se pregunte cómo se encontrará el fichero de clave privada. Dado que TortoisePlink.exe es simplemente una versión modificada de la herramienta estándar Plink dentro del conjunto de utilidades de PuTTY, TortoiseSVN también intentará utilizar todas las claves almacenadas en Pageant.

Si utiliza este último método, asegúrese de no tener establecido un usuario por defecto en PuTTY. Hemos tenido informes de errores en PuTTY provocando que las conexiones se cierren en este caso. Para eliminar el usuario por defecto, simplemente elimine `HKEY_CURRENT_USER\Software\SimonTatham\Putty\Sessions\Default%20Settings\HostName`

Glosario

Actualizar	Este comando de Subversion incorpora los últimos cambios del repositorio a su copia de trabajo, fusionando cualquier cambio hecho por otros con los cambios locales en la copia de trabajo.
Añadir	Un comando de Subversion que se utiliza para añadir un fichero o un directorio a su copia de trabajo. Los nuevos ítems se añaden al repositorio cuando confirme.
Autoría	Este comando es sólo para ficheros de texto, y anota cada línea para mostrar la revisión del repositorio en la que se cambió por última vez, y el autor que hizo ese cambio. Nuestra implementación GUI se llama TortoiseBlame y también le muestra la fecha y hora de la confirmación y ell mensaje de registro cuando mueve el ratón por encima del número de revisión.
BDB	Base de datos Berkeley. Un soporte de base de datos muy probado para los repositorios, que no puede utilizarse en unidades compartidas de red. Por defecto para repositorios anteriores a la versión 1.2.
Bloqueo	Cuando obtiene un bloqueo de un ítem versionado, lo marca en el repositorio como no confirmable, excepto para la copia de trabajo desde la que se obtuvo el bloqueo.
Cambiar	De la misma forma que “Actualizar-a-la-revisión” cambia la ventana temporal de una copia de trabajo para que quede como estuvo en un punto diferente de su historia, “Cambiar” cambia la ventana espacial de una copia de trabajo para que apunte a una parte diferente del repositorio. Es particularmente útil cuando esté trabajando en troncos y ramas en los que sólo hay unos pocos ficheros diferentes. Puede cambiar su copia de trabajo entre los dos y sólo se transferirán los ficheros cambiados.
Confirmar	Este comando de Subversion se utiliza para pasar los cambios desde su copia de trabajo local al repositorio, creando una nueva revisión en el repositorio.
Conflicto	Cuando los cambios desde el repositorio se mezclan con los cambios locales, a veces esos cambios ocurren en las mismas líneas. En este caso Subversion no puede decidir automáticamente qué versión utilizar, y se dice que el fichero está en conflicto. Tiene que editar el fichero manualmente y resolver el conflicto antes de que pueda confirmar más cambios.
Copia de trabajo	Esta es su “caja de arena” local, el área donde trabaja con los ficheros versionados, y normalmente se encuentra en su disco duro local. Puede crear una copia de trabajo al “Obtener” desde un repositorio, y devolver sus cambios al repositorio utilizando “Confirmar”.
Copiar	En un repositorio de Subversion puede crear una copia de un único fichero o un árbol completo. Estos se implementan como “copias baratas” que actúan un poco como un enlace al original en el sentido de que casi no ocupan espacio. Haciendo una copia preserva la historia del ítem en la copia, por lo que puede rastrear los cambios que se hicieron antes de que ocurriera la copia.

Diff
: Abreviatura de “Mostrar diferencias”. Muy útil cuando desea ver exactamente qué cambios se han hecho.

Eliminar
: Cuando elimina un ítem versionado (y confirma el cambio), el ítem ya no existirá nunca más en el repositorio después de la revisión confirmada. Pero por supuesto aún existe en las revisiones previas en el repositorio, por lo que aún puede acceder a él. Si es necesario, puede copiar un ítem eliminado y “resucitarlo” por completo con su historia.

Exportar
: Este comando produce una copia de una carpeta versionada, igual que una copia de trabajo pero sin los directorios locales `.svn`.

FSFS
: Un sistema de ficheros propietario de Subversion que se utiliza como soporte de los repositorios. Se puede utilizar en unidades compartidas de red. Se utiliza por defecto para los repositorios creados a partir de la versión 1.2.

Fusionar
: El proceso por el cual los cambios del repositorio se añaden a su copia de trabajo sin interrumpir cualquier cambio que haya realizado ya localmente. A veces estos cambios no se pueden reconciliar automáticamente, y se dice que la copia de trabajo está en conflicto.

  Las fusiones ocurren automáticamente cuando actualiza su copia de trabajo. También puede fusionar cambios específicos desde otras ramas utilizando el comando Fusionar de TortoiseSVN.

GPO
: Objeto de política de grupo

Historial
: Muestra la historia de las revisiones de un fichero o carpeta. También se conoce como “Registro”.

Importar
: Comando de Subversion para importar una jerarquía de carpetas completa en el repositorio en una única revisión.

Limpieza
: Citando el libro de Subversion: “Limpia recursivamente la copia de trabajo, eliminando los bloqueos y continuando las operaciones sin terminar. Si alguna vez obtiene un error `copia de trabajo bloqueada`, ejecute este comando para eliminar los bloqueos y conseguir que su copia de trabajo vuelva a un estado usable de nuevo.” Tenga en cuenta que en este contexto “bloqueado” se refiere a un bloqueo del sistema de ficheros local, no a bloqueos del repositorio.

Obtener
: Un comando de Subversion que crea una copia local de trabajo en un directorio vacío al descargar los ficheros versionados desde el repositorio.

Parche
: Si una copia de trabajo tiene cambios únicamente en ficheros de texto, es posible utilizar el comando Diff de Subversion para generar un único fichero con el resúmen de esos cambios en el formato de Diff unificado. Un fichero de este tipo a menudo se denomina “Parche”, y puede ser enviado por email a otro (o a una lista de correo) y apllicado en otra copia de trabajo. Alguien sin acceso de confirmación puede hacer cambios y enviar un fichero de parche para que lo aplique un confirmador autorizado. O si no está seguro de un cambio, puede enviar el parche a otros para que lo revisen.

Propiedad
: Además de versionar sus directorios y ficheros, Subversion le permite añadir metainformación versionada - llamadas “propiedades” para cada uno de sus directorios y ficheros

versionados. Cada propiedad tiene un nombre y un valor, como un entrada del registro. Subversion tiene algunas propiedades especiales que utiliza internamente, como `svn:eol-style`. `TortoiseSVN tambitsvn:logminsize`. Puede a

Propiedad de revisión (revprop)

Al igual que los ficheros pueden tener propiedades, las revisiones del repositorio también. Algunas revprops especiales se añaden automáticamente cuando se crea la revisión; son: `svn:date` `svn:author` `svn:log` que representan la fecha/hora de la confirmación, el confirmador y el mensaje de registro, respectivamente. Estas propiedades pueden editarse, pero no están versionadas, por lo que cualquier cambio es permanente y no puede deshacerse.

Rama

Un término de uso frecuente en los sistemas de control de versiones para describir lo que ocurre cuando el desarrollo se parte en un punto en concreto y sigue dos caminos separados. Puede crear una rama desde la línea principal de desarrollo para desarrollar una nueva funcionalidad sin hacer que la línea principal quede inestable. O puede hacer una rama de una versión estable en la que sólo hará correcciones de errores, mientras los nuevos desarrollos tienen lugar en el inestable tronco. En Subversion una rama se implementa como una “copia barata”.

Registro

Muestra la historia de las revisiones de un fichero o carpeta. También se conoce como “Historial”.

Relocalizar

Si su repositorio se mueve, quizás porque lo ha movido a un directorio diferente en su servidor, o el nombre de dominio del servidor ha cambiado, necesitará “relocalizar” su copia de trabajo para que sus URLs de repositorio apunten al nuevo lugar.

Nota: sólo debería utilizar este comando si su copia de trabajo apunta al mismo lugar en el mismo repositorio, pero es el propio repositorio el que se ha movido. En cualquier otra circunstancia probablemente necesita el comando “Cambiar” en su lugar.

repositorio

Un repositorio es un lugar central donde se almacenan y mantienen los datos. Un repositorio puede ser un lugar donde se encuentran múltiples bases de datos o ficheros para distribuirlos en una red, o un repositorio puede ser un lugar directamente accesible por el usuario sin tener que viajar por una red.

Resolver

Cuando los ficheros en una copia de trabajo se quedan en un estado de conflicto tras una fusión, estos conflictos deben arreglarse por un humano utilizando un editor (o quizás TortoiseMerge). Este proceso se denomina “Resolviendo conflictos”. Cuando finalice, marque los ficheros con conflictos como resueltos, y eso le permitirá confirmarlos.

Revertir

Subversion maniene una copia local “prístina” de cada fichero tal y como estaban cuando actualizó su copia de trabajo por última vez. Si ha hecho cambios y decide que quiere deshacelros, puede utilizar el comando “revertir” para volver a la copia prístina.

Revisión

Cada vez que confirma un conjunto de cambios, crea una nueva “revisión” en el repositorio. Cada revisión representa el estado del árbol del repositorio en un cierto punto de su historia. Si desea volver en el tiempo para examinar el repositorio tal y como estaba en la revisión N.

	En otro sentido, una revisión puede referirse al conjunto de cambios que se hicieron cuando esa revisión se creó.
revisión BASE	La revisión base actual de un fichero o una carpeta en su *copia de trabajo*. Esta es la revisión en la que estaba el fichero o la carpeta cuando se hizo la última operación de obtener, actualizar o confirmar. La revisión BASE normalmente no equivale a la revisión HEAD.
revisión HEAD	La última revisión de un fichero o una carpeta en el *repositorio*.
SVN	Una abreviatura muy usada para Subversion.
	El nombre del protocolo personalizado de Subversion que utiliza el servidor de repositorios “svnserve”.

Índice

Símbolos

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W