

Gestión de la configuración

DÍA 3

Control de versiones

SVN

Tareas de la Práctica

- Hecho
 - Documento inicial de control de cambios.
 - DP_ControldeCambios_GrupoN-v1.doc
 - Versión asociada a la revisión por pares.
 - DP_ControldeCambios_GrupoN-v2.doc
 - Creada con las solicitudes v1_1 aceptadas o rechazadas en v1_2:
 - DP_ControldeCambios_GrupoN-v1_1.pdf. *Solicitudes de cambio*
 - DP_ControldeCambios_GrupoN-v1_2.pdf. *Aceptación/rechazo solicitudes*
- Pendiente
 - Versión mejorada del proceso.
 - DP_ControldeCambios_GrupoN-v3.doc
- Presentación SVN
 - Solo existirá: DP_ControldeCambios_GrupoN.doc

Control de versiones

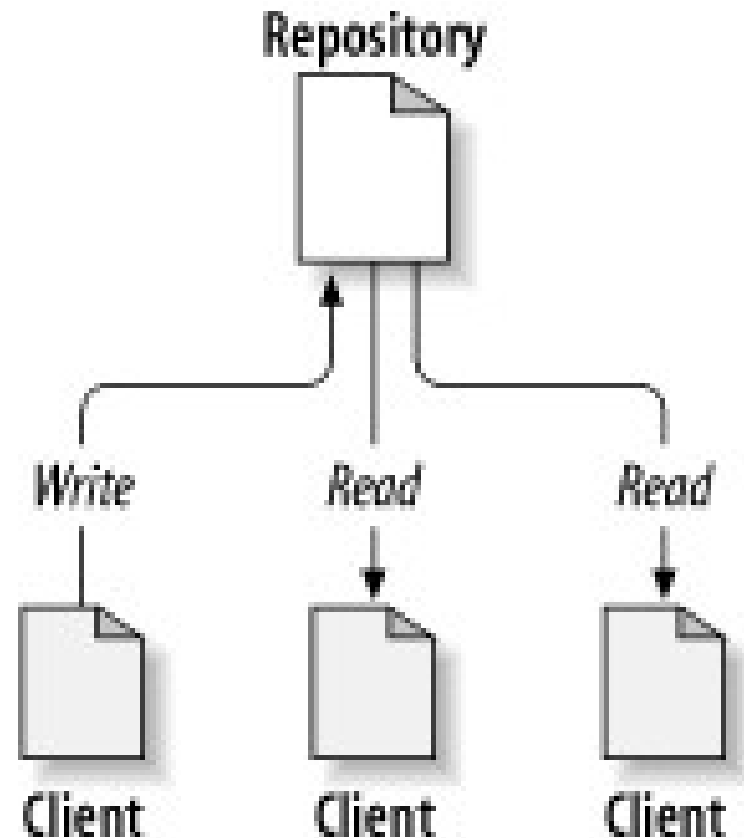
- Objetivo: Seguir y mantener todas las versiones y modificaciones significativas en un documento.
 - Aunque el control de versiones se aplica sobre todo a código, sus principios, y frecuentemente sus herramientas, permiten su aplicación sobre cualquier tipo de documento.
- Alcanza su máximo sentido en desarrollos en equipo, de donde surgió. Aún así, su uso debe plantearse para cualquier desarrollo.

Herramientas

- El control de versiones es imprescindible en cualquier proyecto. Puede realizarse manualmente.
- Herramientas
 - CVS: Sistema centralizado original. Basado en fichero tiene fuertes limitaciones para el seguimiento de proyectos.
 - **SVN**: Sistema centralizado desarrollado tratando de resolver problemas en el CVS.
 - Git: Sistema distribuido desarrollado por Linus Torvalds usado en el mantenimiento del núcleo de linux. Puede usar repositorios svn
 - Mercurial: Sistema distribuido implementado en Python originalmente para Linux.

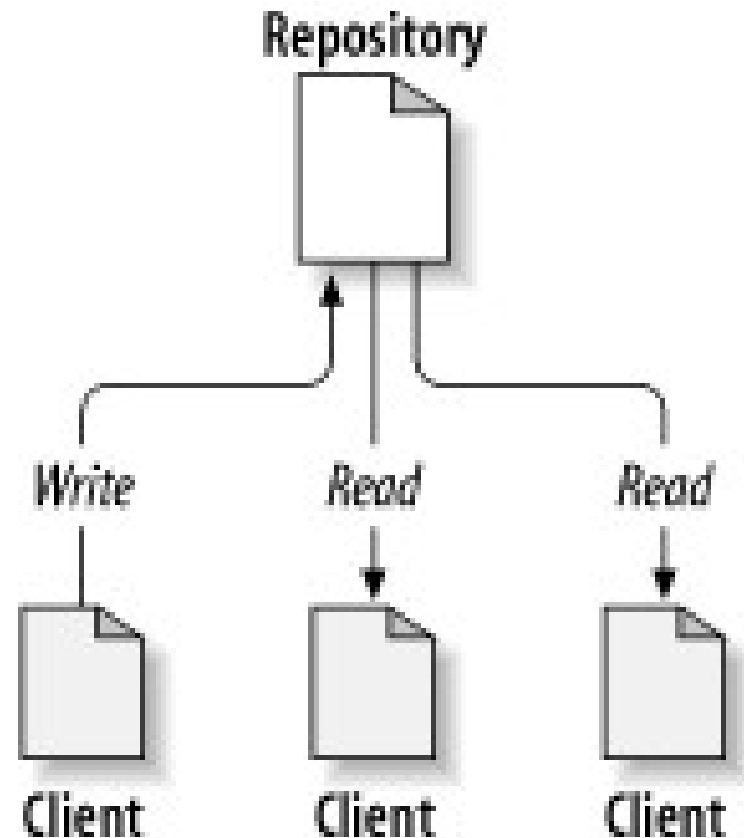
Subversión: Repositorio

- Almacén “centralizado” en el que compartir “toda” la información del proyecto
 - Permite el acceso de múltiples clientes
 - Información en árbol
 - Recuerda todos los cambios hechos sobre él. Incluidos los realizados sobre el propio árbol
 - El cliente puede ver estados previos del sistema



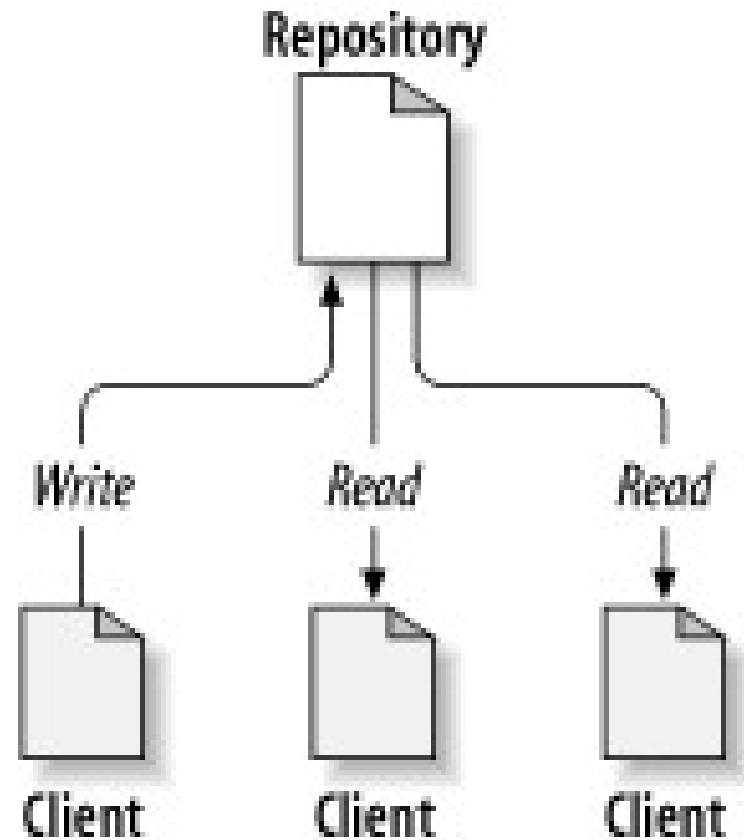
Subversión: Repositorio

- Semejante al Drop-Box
 - La información está en remoto (repositorio) y local.
 - El repositorio guarda versiones de cada documento y es posible volver a ellas.
 - Cuando se modifica la copia en local se modifica en remoto.

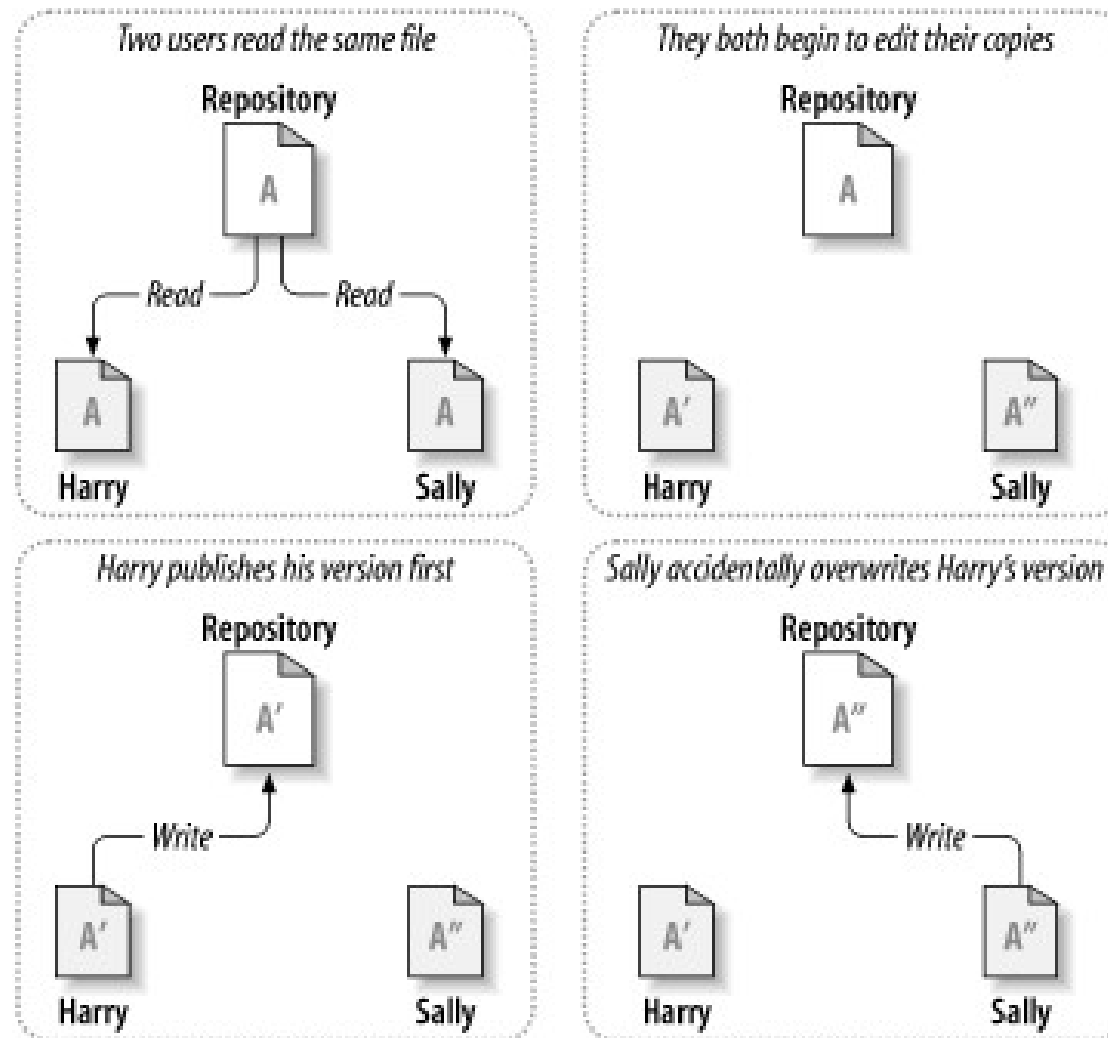


Subversión: Repositorio

- Problemas en Drop-Box
 - Se guardan versiones intermedias.
 - Los ficheros guardan versiones independientes.
 - ¿Cómo sabes que contenían otras versiones?. ¿En qué se diferencian de la actual.?
 - ¿Qué ocurre si varios usuario cambian el mismo fichero?



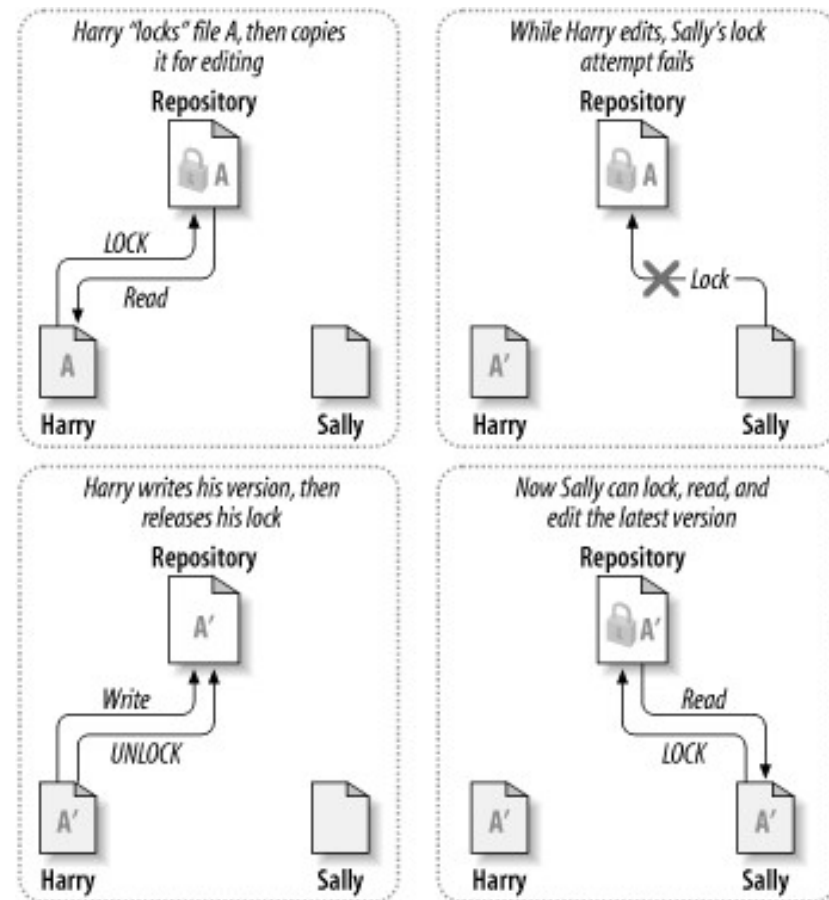
Subversión. El problema a evitar



Solución:

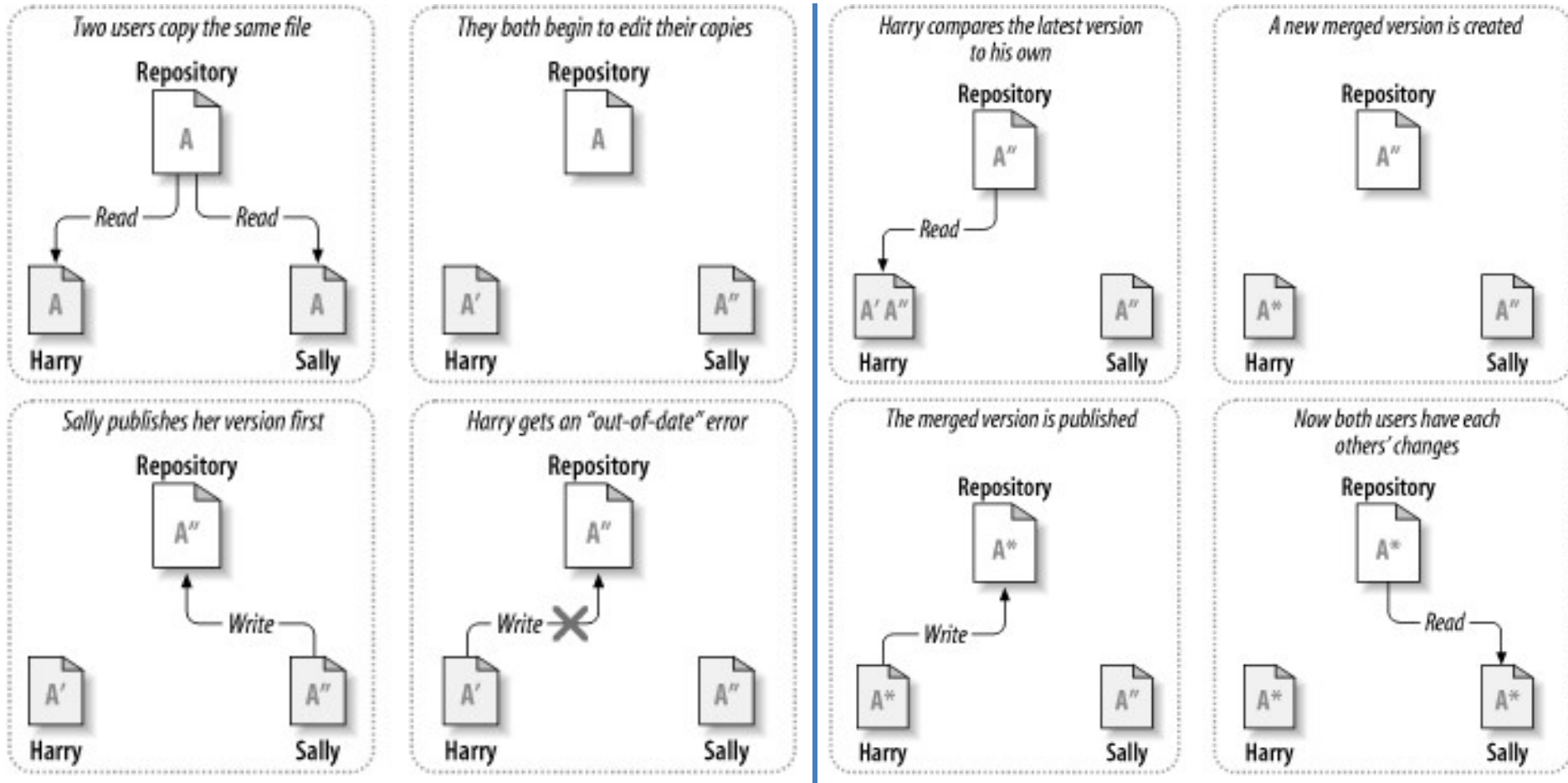
Bloqueo-modificación-desbloqueo

- **Bloqueos por tiempo indefinido.**
- **Serialización innecesaria.** Ej. No se pueden modificar dos métodos independientes en el mismo fichero.
- **Falsa sensación de seguridad.** Un código en varios ficheros A y B se puede modificar de forma incompatible



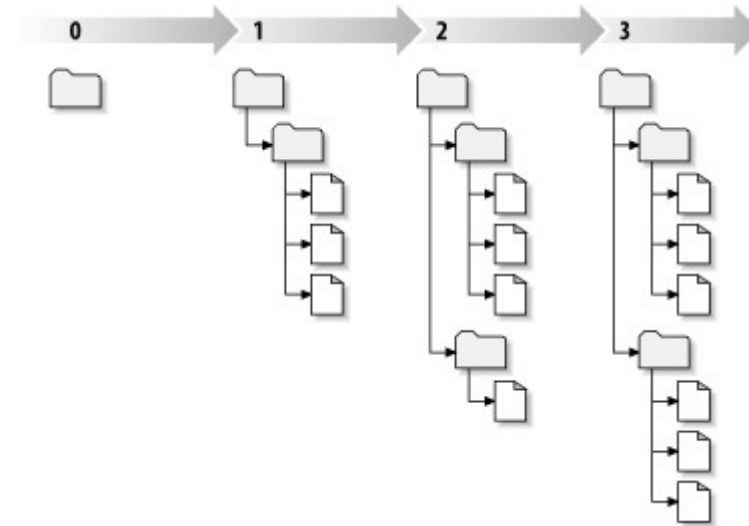
Solución SVN.

Copiar-modificar-mezclar



Versiones

- En subversión cada modificación en cualquier fichero implica un cambio en el **número de revisión** para todo el árbol de directorios
- Un fichero puede cambiar de revisión sin haber sido modificado.



¿Qué necesitamos?

- Servidor svn: Subversión
 - Herramienta de código abierto
 - Multiplataforma
 - <http://subversion.apache.org/>
- Cliente: Tortoise SVN
 - Herramienta de código abierto
 - Integrado con el explorador de Windows
 - Permite ejecutar todos los comandos SVN desde su menú contextual
 - *Puede actuar como servidor local de SVN*
 - <http://tortoisesvn.net/>

¿Y con eclipse?

- Servidor svn: Subversión
 - Herramienta de código abierto
 - Multiplataforma
 - <http://subversion.apache.org/>
- Cliente: Subversive Plug-In
 - Herramienta de código abierto
 - <http://www.eclipse.org/subversive/index.php>

Comenzar - Administrador

- Crear un repositorio
 - Usar un servidor SVN
 - `https://Host/repositorio`
 - Herramienta Repo-browser
 - Crear un repositorio local con Tortoise
 - `file:///c:/repositorio`
 - **Create Repository Here**
 - Importar estructura
 - Crear una estructura según la filosofía que se desee
 - **Import** estructura
 - Importa todo lo que haya dentro de la carpeta estructura al repositorio. La carpeta estructura no se incluye.
 - Añadir un proyecto al repositorio
 - Ídem que con la estructura incluyendo también los ficheros del proyecto.
- Estructuras del repositorio
 - Raíz
 - branches
 - tags
 - trunk
 - Proyecto 1
 - Proyecto 2
 - ...
 - Raíz
 - Proyecto1
 - branches
 - tags
 - Trunk
 - Proyecto2
 - ...
 - EL REPOSITORIO NO PUEDE MANIPULARSE DIRECTAMENTE

Ramas y Etiquetas

- Una rama o branch es una copia del repositorio en un momento dado del desarrollo
 - La copia seguirá un desarrollo paralelo al tronco o la rama principal
 - En el futuro la rama puede abandonarse o volver a fusionarse con el tronco (**merge**)
- Etiquetas o Tags: Se corresponden con una copia del repositorio igual que una rama.
 - Una copia de Tag se marca con una etiqueta de referencia para indicar una versión especial. Release, estable, ...
 - Una copia marcada **jamás se modifica.**

Comenzar - Cliente

- Crear un directorio de trabajo.
 - Se crea un directorio normal
 - Se hace un Checkout de un directorio del repositorio
 - Sólo se hace una vez
 - Convierte el directorio normal en una copia de trabajo
 - Crea directorios ocultos .svn
 - Podemos crear tantos como queramos
 - Borrarlos, si la información está actualizada en el repositorio, no tiene consecuencias
- Checkout vs export
 - En un directorio normal
 - Se hace export de un directorio del repositorio
 - Se crea una copia exacta de la información en el repositorio
 - No incluye información ni directorios .svn
 - No genera un directorio de trabajo
 - Es la estrategia normal para extraer la información para un entregable o para importarla hacia otro repositorio.

Ciclo de trabajo

- Actualizar su copia de trabajo local
 - update
- Hacer cambios (S.O.)
 - add
 - delete
 - copy
 - move
- Enviar sus cambios
 - commit
- Examinar sus cambios
 - cleanup
 - status
 - show log
 - diff
 - revert
- Fusionar los cambios de otros en su copia de trabajo
 - merge
 - resolve

TortoiseSVN Status



Ciclo de trabajo

- Ciclo de trabajo
 - Update
 - Commit
- Ciclo de trabajo con conflicto
 - Update
 - Merge
 - Resolved
 - Commit
- Estado de las copias de trabajo
 - Sin cambios y actualizado: commit y update no hacen nada
 - Modificado local y actualizado: commit opera y update nada
 - Sin cambios y desactualizado: commit nada y update refresca
 - Modificado local y desactualizado: commit falla y update mezcla

Crear Ramas y Etiquetas

- Crear una rama o branch del repositorio en un momento dado del desarrollo
 - Conflicto Harry-Sally
 - Refusionar rama con el tronco
- Crear una Etiqueta o Tag.
 - ~~Crear versiones~~
 - Una copia marcada **jamás se modifica.**

Trabajo en parejas

- Hacer la práctica: “TrabajoEnParejas.pdf”.
 - Se realiza en clase
 - Distinguir entre
 - Repositorio, Copia de trabajo, versión exportada
 - Entender la estructura de un repositorio y los conceptos
 - Rama, Trunk, Tag.
 - Acciones
 - Add, delete, copy, move. Ficheros en la copia de trabajo.
 - Cleanup, status, show log, diff, revert. Manejo de versiones.
 - Entender ciclo de trabajo y los procesos
 - Update, Merge, Resolved, Commit.
 - Entender la creación de Ramas y Tags y el fusionado de las primeras con el tronco de desarrollo.
- Prepararse para hacer una práctica Individual.