



Análisis y Diseño de Sistemas 2

2015

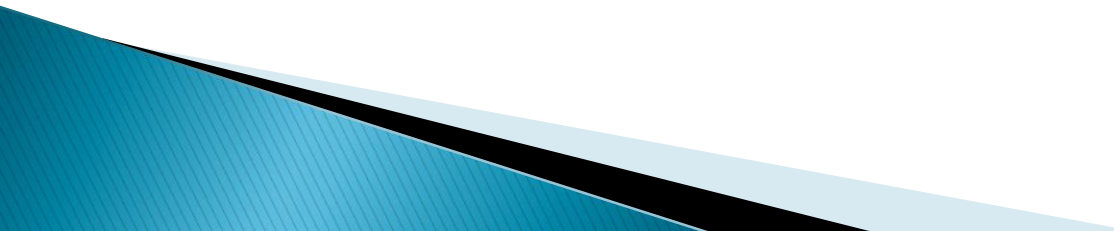
Ing. Luis Alberto Arias Solórzano

Unidad 1

Repositorio

- ▶ Conjunto de información gestionada por el sistema. Este repositorio contiene el historial de versiones de todos los elementos gestionados.

Repositorio en una herramienta de control de versiones

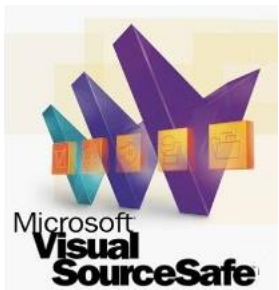
- ▶ Sitio centralizado donde se almacena y mantiene información habitualmente en bases de datos o archivos informáticos, debe contar con sistema de backup y mantenimiento preventivo.
 - ▶ Un cliente lee datos del repositorio, normalmente sólo ve la última versión del árbol de archivos. Pero el cliente también tiene la habilidad de ver estados previos del sistema de archivos
- 

Sistema de control de versiones

- ▶ Sistemas que están diseñados para registrar y seguir cambios en datos a través del tiempo

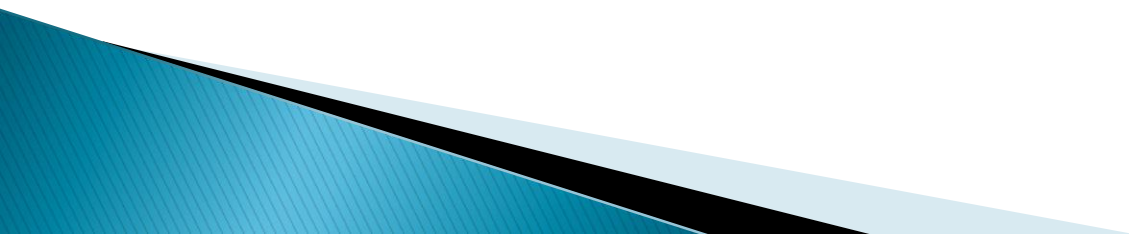
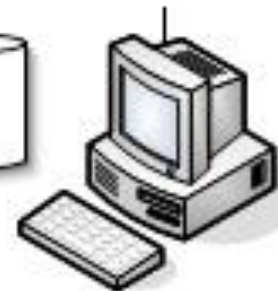
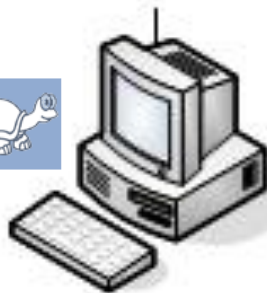
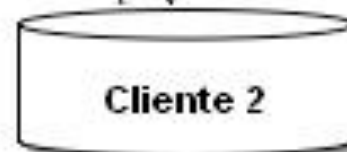
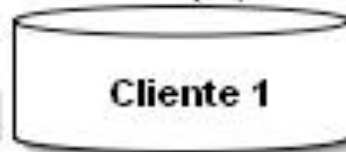


CVS



Cientes:





Modelos de versionado

El problema de compartir archivos

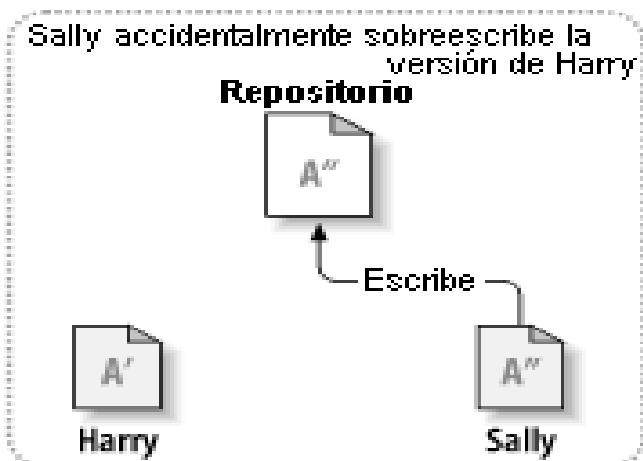
- Bloqueo-modificación-bloqueo
- Copiar-modificar-mezclar



El problema de compartir archivos

- ▶ 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, es posible que (unos momentos después) Sally pueda accidentalmente sobrescribirlos 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.

El problema de compartir archivos



Solución 1:

Bloqueo-modificación-bloqueo

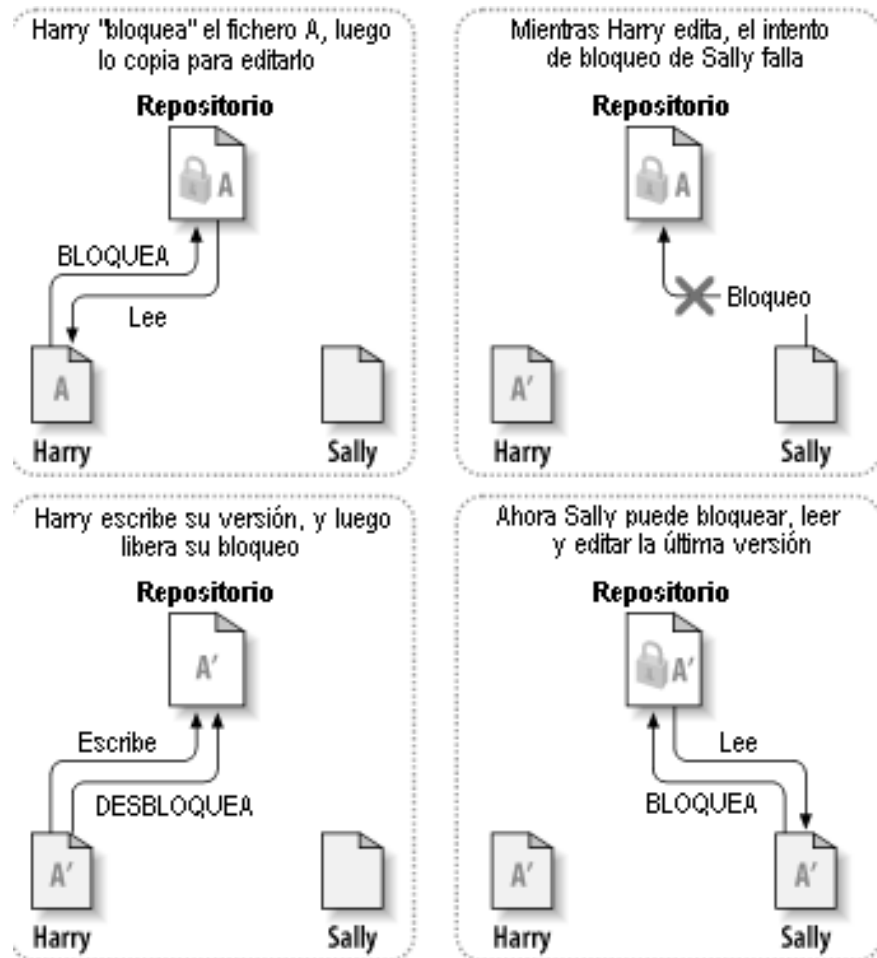
- ▶ El repositorio sólo permite que una persona cambie un fichero al mismo tiempo.

Situaciones de conflicto:

- Problemas administrativos: a un usuarios se le olvida desbloquear
- Serialización innecesaria: ¿Qué ocurre si Harry está editando el inicio de un fichero de texto, y Sally simplemente quiere cambiar la parte final del mismo fichero?
- 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.

Solución 1:

Bloqueo-modificación-bloqueo



Solución 2:

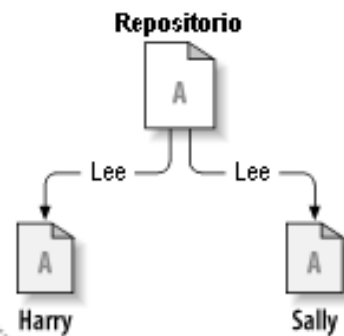
Copiar-modificar-mezclar

- ▶ 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.

Solución 2:

Copiar-modificar-mezclar

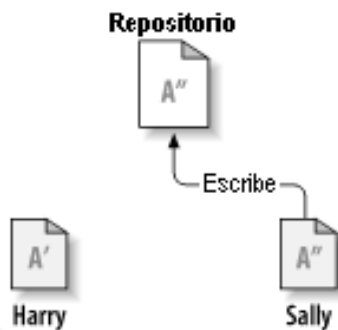
Dos usuarios copian el mismo fichero



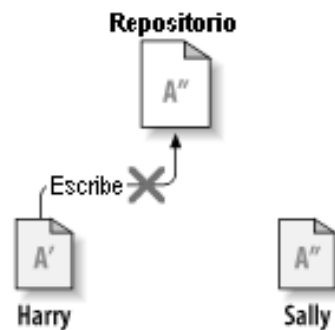
Ambos empiezan a editar sus copias



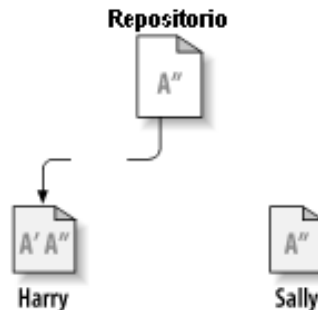
Sally publica su versión primero



Harry obtiene un error "no actualizado"



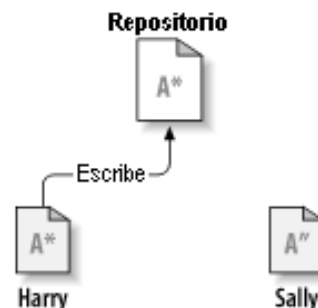
Harry compara la última versión con la suya



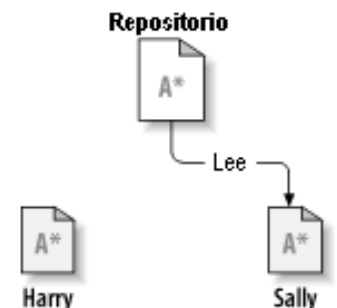
Se crea una nueva versión fusionada



Se publica la versión fusionada



Ahora ambos usuarios tienen los cambios del otro



Terminología

- ▶ Configuración (Setup básico)
 - Repositorio – repository
 - Servidor – server (donde esta almacenado el repositorio)
 - Cliente – client (la computadora que se conecta al repositorio)
 - Copia local – Working Set / Working copy
 - Trunk / Main – Locación primaria para el código en el repositorio

► Acciones básicas

- **Add** – Agregar un archivo a un proyecto por primera vez
- **Revision** – Versión en la que está el archivo (v1, v2)
- **Head** – La última versión en el repositorio, la más reciente
- **Check out** – Bajar un archivo del repositorio, generalmente se bloquea dicho archivo (solución 2)
- **Check in** – Subir el archivo al repositorio luego de modificarlo, el archivo obtiene un nuevo número de **revisión**
- **Checkin message** – mensaje corto describiendo lo que se cambió
- **Changelog / history** – Listado de todos los cambios en un archivo desde que fue creado
- **Update/Sync** – Sincronizar los archivos con el último en el repositorio, se toman las ultimas versiones de c/u de los archivos
- **Revert (Undo check out)**: descartar los cambios locales y recargar la última versión de un archivo

► Acciones avanzadas:

- **Branch** – Crear una copia separada de un archivo o folder para uso privado (arreglar un bug, pruebas, nueva version experimental, etc). “¿En qué branch esta?”
- **Diff/Change/Delta** – Encontrar las diferencias entre dos archivos.
- Merge / patch: Aplicar los cambios de un archivo a otro para actualizarlo
- **Conflicto (conflict)** – Cuando hay cambios pendientes que afectan a un archivo
- **Resolve** – Componer los cambios que contradicen archivos y hacer check in (commit) los cambios en el repositorio
- **Locking** – Tomar el control sobre un archivo para que nadie más pueda editarlo hasta que se quite el bloqueo (unlock)
- **Breaking the lock** – forzar un desbloqueo para poder editar un archivo
- **Check out for edit** – Dar check out haciendo editable una versión de un archivo

Ejemplos

Alice agrega un archivo (list.txt) al **repositorio**. Le da **check out**, hace un cambio (pone “milk” en la lista), y le da un **check in** con un **checkin message** (“Added required item.”).

La siguiente mañana, Bob actualiza su **working copy** y visualiza la última **revision** de list.txt, la cual contiene “milk”. Puede ver el **changelog** o usar **diff** para ver que Alice puso “milk” el día anterior.

Checkins

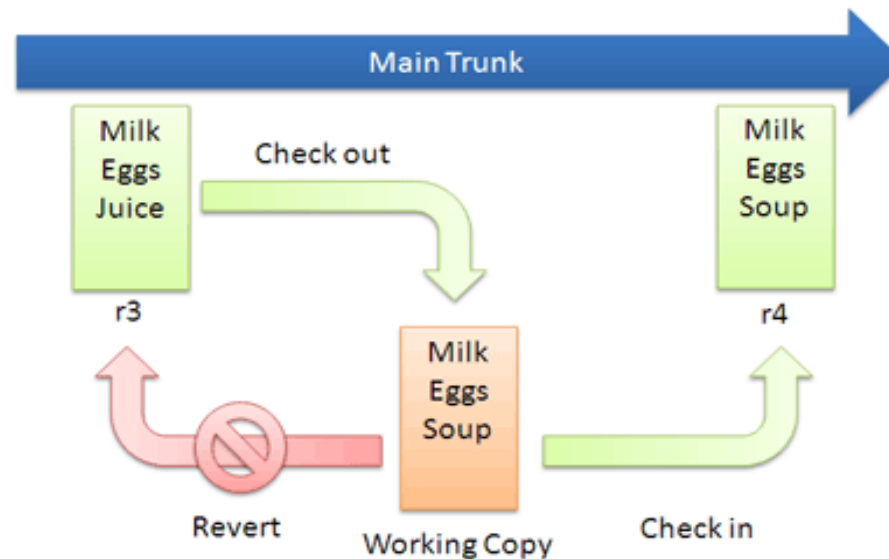
Basic Checkins



- ▶ Cada vez que se da un **check in** se obtiene un nuevo numero de revisión (r1, r2, r3)

Checkouts y modificaciones

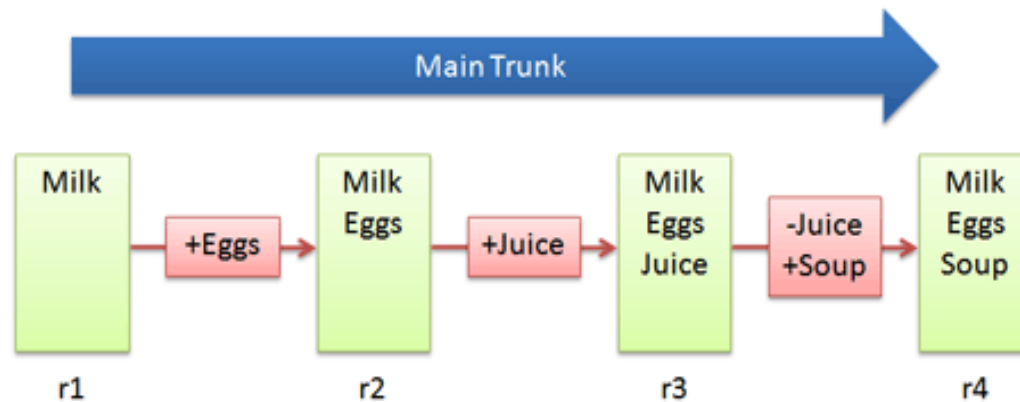
Checkout and Edit



- ▶ Check out – MODIFICAR – Check in y si no le gustan sus cambios puede REVERTIRLOS y trabajar con la revisión original

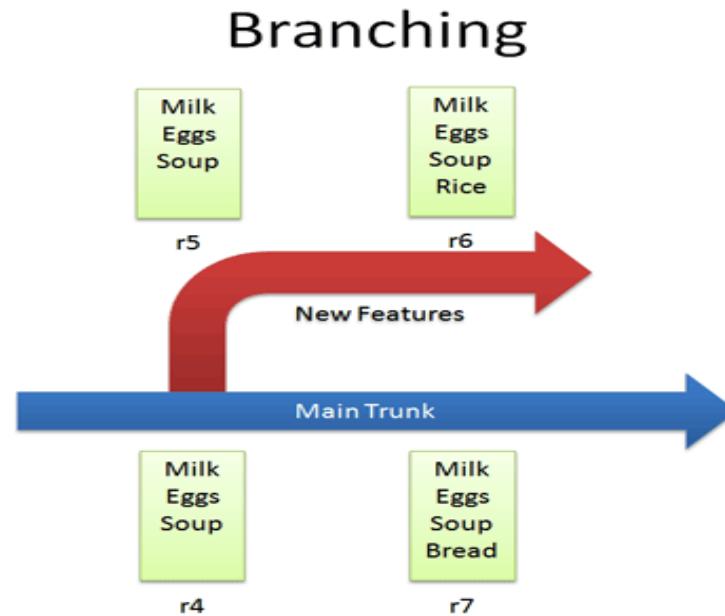
Diffs

Basic Diffs



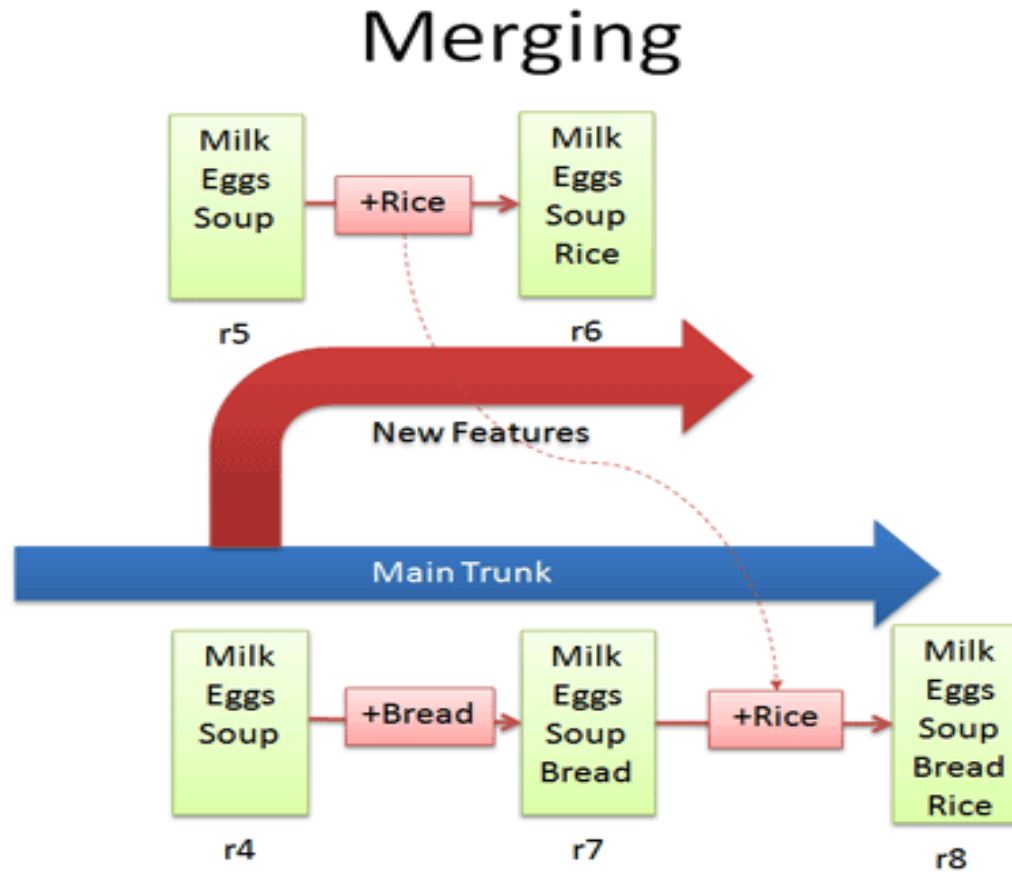
- ▶ Diffs son los cambios hechos cuando se esta modificando, muchos de los sistemas de control de versiones guardan diffs en lugar de copias completas, para guardar espacio en disco

Branching

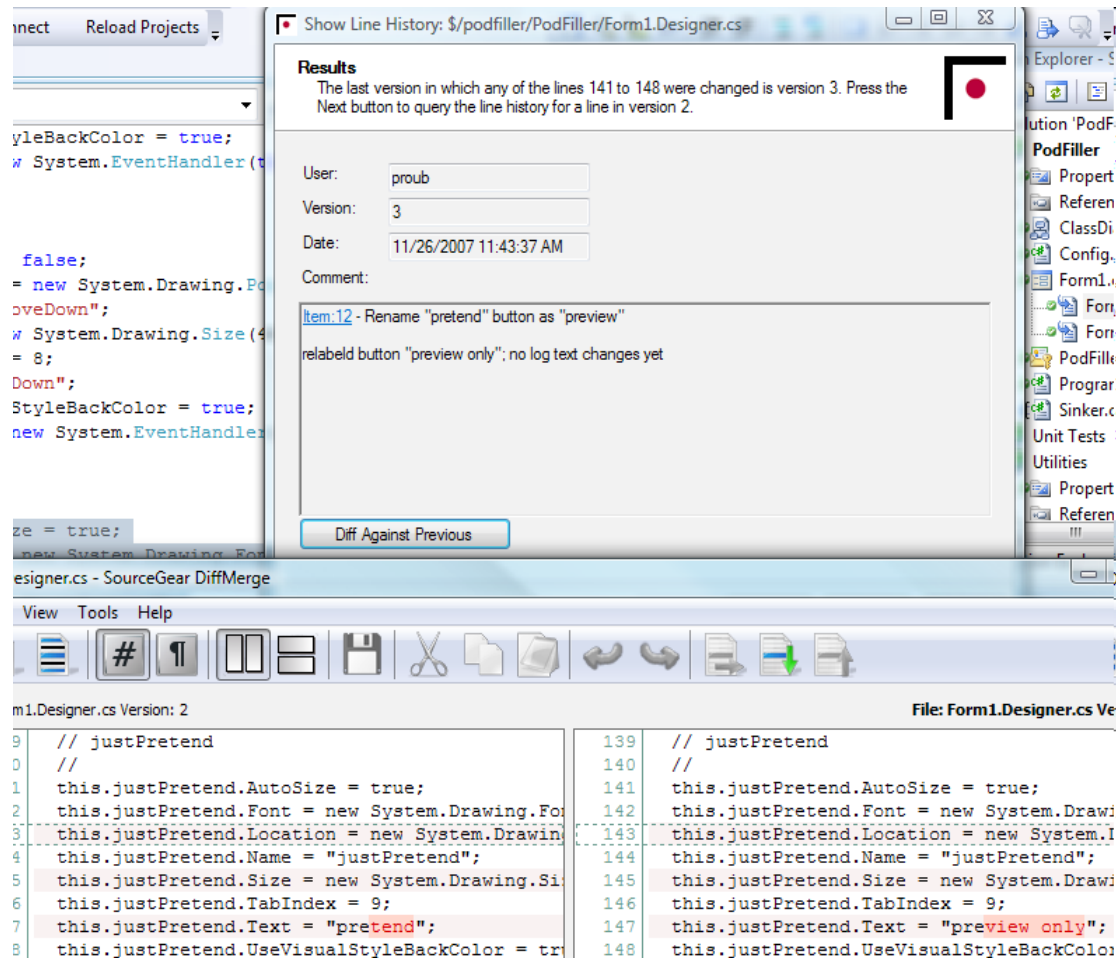


- ▶ Nos permite copiar código en folders separados es una copia exacta donde podemos tener diferentes revisiones que no afectan el proyecto actual

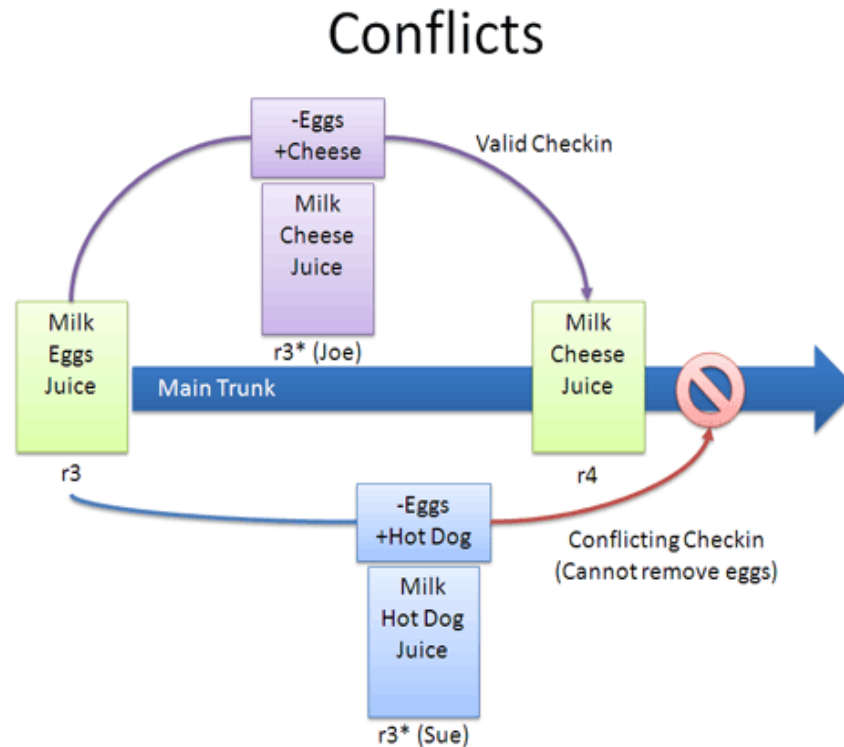
Merging



Merge con subversion

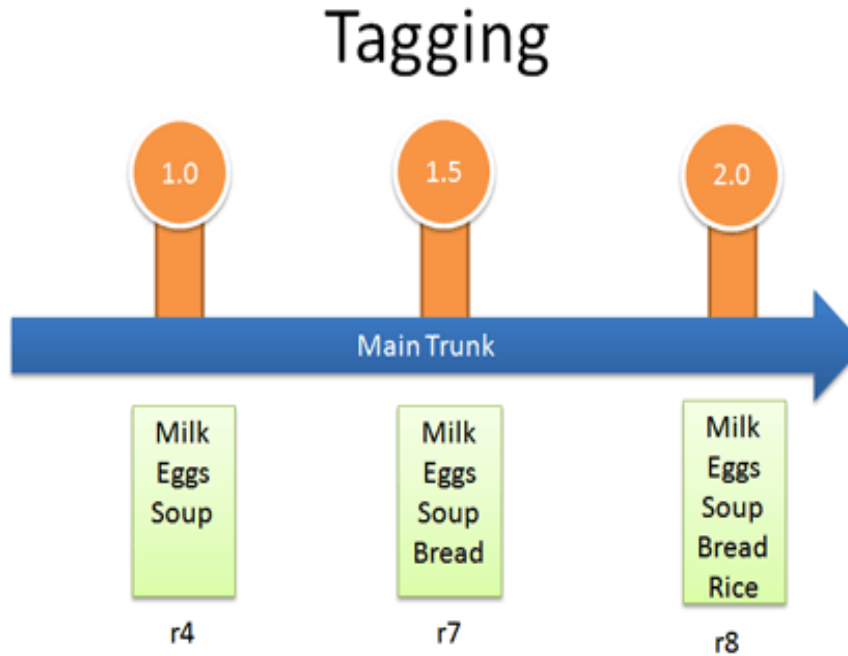


Conflicts



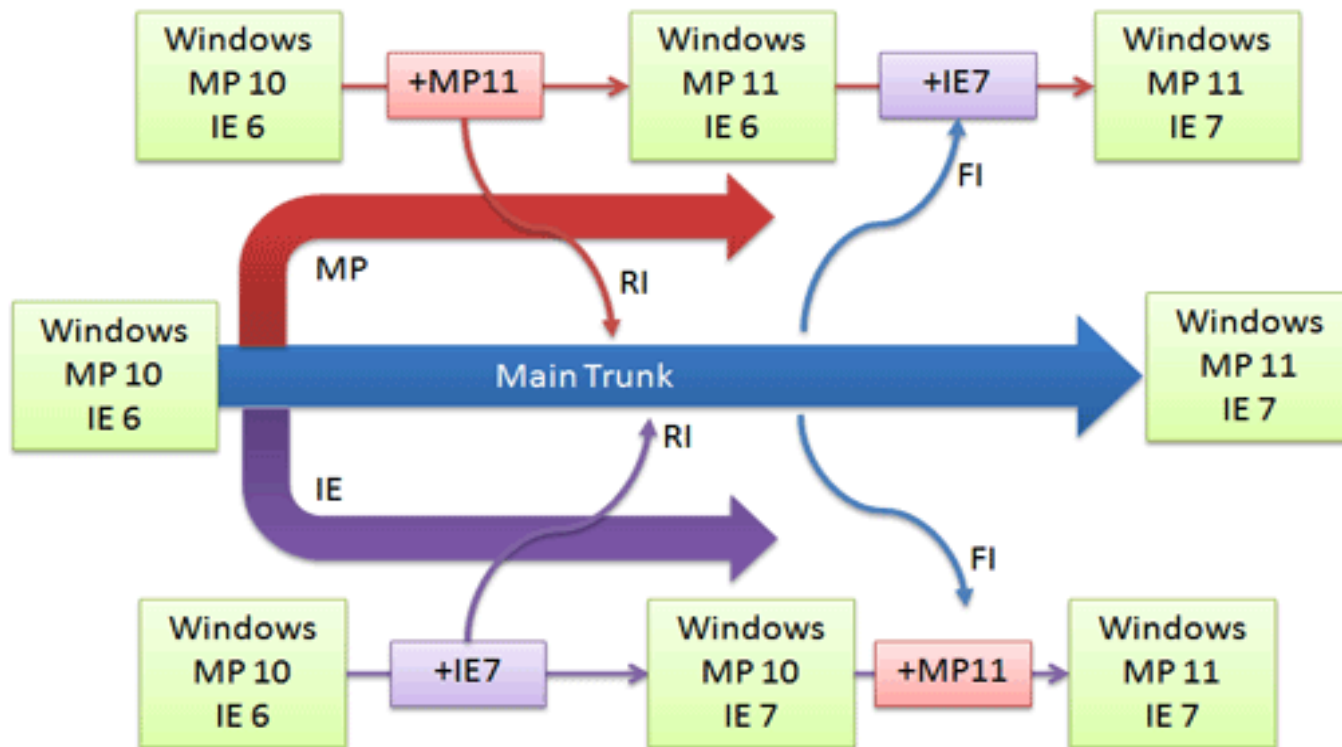
- ▶ Reaplicar cambios a la versión del repositorio: **Sync** (Sincronizar) y reaplicar los cambios

Tagging



- ▶ Se etiquetan las revisiones de manera que no se tenga que referir al número de build, sino a una version por ejemplo: “Release 1” o “Windows XP Service Pack 1” o “Mozilla Firefox 3.0”

Managing Windows



Bibliografia

Beyond Software Architecture: Creating and Sustaining Winning Solutions

[Luke Hohmann, Guy Kawasaki](#)

Chapter 15: Release Management

Modelos de versionado

http://tortoisesvn.net/docs/release/TortoiseSVN_es/tsvn-basics-versioning.html

A Visual Guide to Version Control

<http://betterexplained.com/articles/a-visual-guide-to-version-control>

Gracias