Eusko Jaurlaritzaren Informatika Elkartea
Sociedad Informática del Gobierno Vasco

Subversion:

Manual de usuario

Fecha:

Referencia:

EJIE S.A.
Mediterráneo, 3
Tel. 945 01 73 00*
Fax. 945 01 73 01
01010 Vitoria-Gasteiz
Posta-kutxatila / Apartado: 809
01080 Vitoria-Gasteiz
www.ejie.es

Control de documentación

Título de documento: Subversión. Manual de Usuario

Histórico de versiones

Código:

Versión: 2.8

Fecha: Diciembre 2014

Nueva estructura (2.2), Acceso http e identificación contra ldap y gestión de hooks.

Versión 2.4: Se elimina el acceso http. Se añaden utilidades de SVN en AHP. Se añade estructura para desarrollo para movilidad

Versión 2.5: Se añaden comentarios la estructura de un repositorio en página 13.

Versión 2.6: Añadida la carpeta pcl a la estructura SVN del backend.

Versión 2.7: Añadida la estructura de SVN para la configuración de backend

Versión 2.8: Listado de hooks existentes

Cambios producidos desde la última versión

Listado de hooks disponibles para activar por repositorio

Control de difusión

Responsable:

Aprobado por:

Firma:	Fecha:

Distribución:

Referencias de archivo

Autor:	Consultoría de áreas de conocimiento
Nombre archivo:	Subversion. Manual de Usuario v2.8.docx

Localización:

Contenido

1 Introducción

El presente documento describe cuáles son los aspectos necesarios que se deben conocer para la correcta administración a nivel de Repositorio de Subversion.

2 Conceptos

2.1 Conceptos básicos

Subversion es un sistema open-source escalable de control de versiones, muy potente, usable y flexible, que ha sido diseñado para sustituir a CVS. Para ello trata de proporcionar funcionalidades similares al CVS preservando su filosofía de desarrollo y de dar solución a los principales defectos del CVS.

A continuación se detallan las principales ventajas de Subversion:

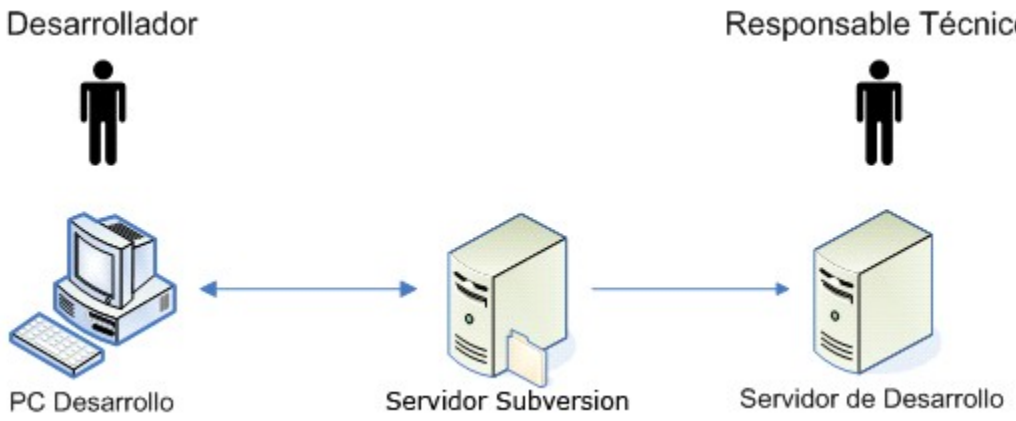
- Se mantiene el histórico de los archivos y directorios tras realizar copias y renombrados.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente; tiene costo de complejidad constante (O(1)) y no lineal (O(n)) como en CVS.
- Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).
- Se puede usar, mediante Apache, sobre WebDAV/DeltaV. Esto permite que clientes WebDAV utilicen Subversion de forma transparente.
- Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto).
- Permite el bloqueo de archivos selectivamente. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.
- Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor proporciona a la hora de autentificar archivos (SQL, LDAP, PAM, etc.).

Para la utilización de Subversion se han de tener en cuenta las siguientes restricciones:

- Se necesita el servidor web Apache 2.0 si se desea acceder al repositorio vía HTTP
- El control por directorios, no es directo con los grupos de LDAP, aún. Sólo pueden ser los grupos virtuales que estén definidos en la entrada AuthzSVNAccessFile del fichero dav_svn.conf.
- El contenido de los mensajes de error no es muy claro.

2.2 Arquitectura de desarrollo

Tal y como se ha comentado anteriormente, Subversion es un servicio que va a existir únicamente en el entorno de Desarrollo. A continuación se muestra la arquitectura a seguir al utilizar el producto.



Los elementos detectados en el diagrama son:

- Servidor Subversión: gestiona los repositorios de código que albergarán el código fuente de las aplicaciones. La política definida establece que cada aplicación disponga de un repositorio.
- PC Desarrollo: los programadores trabajarán en una copia local del código realizando actualizaciones en el repositorio remoto según se vayan cerrando implementaciones parciales que permitan suministrar dicha funcionalidad a otros integrantes del proyecto sin lastrarles con posibles errores de compilación de módulos no modificados por ellos. Al acabar la fase de construcción se marcará una versión sobre el código de la aplicación disponible en el repositorio de código.
- Servidor de Desarrollo: será usado por el responsable técnico de la aplicación para obtener una versión estable del código y poder hacer pruebas funcionales tras lanzar las pertinentes instrucciones de compilación y despliegue.

2.3 Fases

- **Fase -1**

  Esta fase no tiene repercusión respecto al producto.

- **Fase 0 – Puesta en Desarrollo**

  El único aspecto a tener en cuenta al pedir la Fase 0 es indicar que se requiere hacer uso del servicio Subversion para que Implantación incluya este aspecto en la petición de servicio.

  Por otro lado, la única diferencia de cara al Servidor de Desarrollo es que el código fuente de la aplicación en vez ser copiado vía ftp es obtenido directamente del Subversion..

  Una vez que el código fuente de la aplicación es recepcionado en el Servidor de Desarrollo el procedimiento de implantación sigue el procedimiento convencional establecido..

- **Fase 1 – Puesta en Pruebas**

  La puesta en el entorno de Preproducción sigue el procedimiento convencional establecido.

- **Fase 2 – Puesta en Producción**

La puesta en el entorno de Preproducción sigue el procedimiento convencional establecido.

3 Administración

El **responsable de proyecto** (repositorio de Subversion) tendrá a su disposición el usuario **codAplicación/Dpto** para la realización de las tareas relativas a la gestión de usuarios para dicho repositorio (a través de los scripts de gestión de usuarios disponibles). También se podrán dar de alta usuarios mediante la interfaz de Ant Hill Pro (ver Anexo II).

Además, otras tareas, a nivel de repositorio, que podrá realizar el responsable de proyecto serán:

- Definición de la estructura de un repositorio.
- Creación de etiquetas.

Estas dos últimas tareas, definidas a nivel de repositorio, se deben llevar a cabo a través de un cliente de Subversion (línea de comandos, plug-in Subversive para el entorno J2EE o TortoiseSVN para cualquier entorno). Esto implica, darse de alta en el repositorio (a través de Utilidades de usuario en Ant Hill Pro o a través de los scripts de gestión de usuarios en servidor de SVN) para llevar a cabo la realización de las mismas. Por tanto, cualquier usuario del repositorio podría realizar estas tareas pero se consideran funciones adecuadas únicamente para el responsable de proyecto.

3.1 Gestión de Usuarios

En primer lugar se expone la configuración que se ha adoptado para administrar la autentificación y autorización de Subversion con **svnserve**.

A continuación se muestran los scripts implementados para administrar los usuarios de un repositorio, usuarios introducidos en el archivo *passwd* (base de datos de contraseñas) de cada repositorio.

Nota: Todos los usuarios, autenticados dentro del archivo de contraseñas de cada repositorio, estarán autorizados a realizar tanto operaciones de lectura como de escritura sobre dicho repositorio.

3.1.1. Autentificación con *svnserve*

La configuración por defecto de **svnserve** proporciona acceso anónimo de sólo-lectura. Esto significa que se puede utilizar una URL de tipo *svn://* para ver el contenido del repositorio y para obtener del mismo pero no se podrá confirmar ningún cambio.

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

Se podrá saber quién ha hecho cambios en el repositorio y también se podrá controlar quién puede hacer cambios en el repositorio. La forma establecida para conseguir esto es crear una base de datos de contraseñas:

```
[general]
anon-access = none
auth-access = write
```

```
password-db = passwd
```

Donde *passwd* es un fichero que existe en el mismo directorio que *svnserve.conf*. y debería tener una estructura como ésta:

```
[users]
usuario = contraseña
```

Este ejemplo denegaría cualquier acceso a los usuarios no autentificados (anónimo), y daría acceso de lectura/escritura a los usuarios listados en *passwd*.

Esta es la manera que se ha establecido para trabajar con los repositorios de Subversion.

3.1.2. Scripts para la gestión de usuarios

Existen cinco scripts destinados a realizar la administración de usuarios de los repositorios Subversion:

- Menú de opciones.
- Creación de Usuarios.
- Eliminación de Usuarios.
- Listado de Usuarios.
- Actualización de Usuarios.

Se encargarán del mantenimiento del archivo de contraseñas *passwd* de cada uno de los repositorios disponibles.

Nota: La administración de los usuarios de los repositorios (ejecución de estos scripts) deberá realizarla cada **responsable de proyecto** con el usuario creado para tal fin **codAplicacion/Dpto**(usuario/grupo).

Menú de opciones

menu_SVN.sh: Script relativo al menú de opciones para la administración de usuarios de los repositorios Subversion disponibles.

La instrucción adecuada para lanzar los scripts encargados de la gestión de los usuarios es la siguiente:

```
$ sh /repositorios_svn/bin/menu_SVN.sh
```

Este script nos presentará en pantalla el siguiente menú de opciones con el que se podrá acceder a la tarea deseada. Será el script principal o punto de partida de todas las operaciones relativas a la administración de los usuarios de repositorios Subversion.

```
Administracion de Usuarios de SVN (centos-4.3-vm)
-------------------------------------------------

1. Alta de usuarios de un repositorio

2. Baja de usuarios de un repositorio

3. Listado de usuarios de un repositorio

4. Cambio de password de usuarios de un repositorio

-------------------------------------------------
s. Salir

Elije una opcion:
```

Los siguientes apartados muestran el funcionamiento de cada una de las tareas presentadas en el menú de opciones:

- Alta de usuarios de un repositorio.
- Baja de usuarios de un repositorio.
- Listado de usuarios de un repositorio.
- Cambio de password de usuarios de un repositorio.

Creación de Usuarios

alta_user_SVN.sh: Script relativo al alta de usuarios de los repositorios Subversion disponibles.

En caso de elegir la primera opción del menú se ejecutará el script *alta_user_SN.sh* que nos permitirá autorizar el acceso de lectura y escritura a un repositorio para un usuario concreto:

Eliminación de Usuarios

baja_user_SVN.sh: Script relativo a la baja de usuarios de los repositorios Subversion disponibles.

Seleccionando la segunda opción del menú se ejecutará el script *baja_user_SVN.sh*. Este script nos proporciona la posibilidad de eliminar la autorización de acceso (escritura/lectura) a un repositorio concreto para un determinado usuario:

Listado de Usuarios

listado_user_SVN.sh: Script relativo al listado de usuarios de Subversion.

La opción “Listado de usuarios de un repositorio”, evidentemente, mostrará el listado de usuarios del repositorio seleccionado (script *listado_user_SVN.sh*):

Cambio de Password de Usuarios

modif_user_SVN.sh: Script relativo al cambio de password de usuarios de los repositorios Subversion disponibles.

En caso de haber elegido la opción del menú relativa al cambio de password se ejecutará el script *modif_user_SN.sh* que nos permitirá cambiar la contraseña de un usuario para un determinado repositorio:

3.2 Definición de la Estructura de un Repositorio

Será labor del responsable del proyecto la creación de la estructura del repositorio.
La versión antigua de subversión para proyectos Weblogic 8 (compatible con las tareas ant de svn) es:

/repositorios_svn
/proyectos # directorio raíz de todos los repositorios
/xxx # repositorio para el proyecto xxx
/conf # directorio de configuración del repositorio para el proyecto xxx
/svnserve.conf # archivo de configuración de *passwd* para svnserve
/passwd # archivo de contraseñas
/codigo….# carpeta contenedora del código del proyecto xxx
/… # resto de carpetas relativas al código
/script….# carpeta contenedora de los scripts del proyecto xxx
/…
/librerias # carpeta contenedora de las librerías del proyecto xxx
/…
/estatico # carpeta contenedora del contenido estatico del proyecto xxx
/…
/config # carpeta contenedora de los ficheros de configuración del proyecto xxx
/…
/datos # carpeta datos del proyecto xxx
/…

/… # resto de repositorios para diferentes proyectos

La nueva estructura mostrada a continuación deberá ser considerada como la estructura estándar:

ESTRUCTURA

/repositorios_svn
/proyectos # directorio raíz de todos los repositorios
/xxx # repositorio para el proyecto xxx

/codigo # carpeta contenedora del código del proyecto xxx
- **/trunk**
  - **/Nombre del proyecto** *[Solo para Weblogic8]*
- **/branches**
- **/tags**

/backend # código fuente relativo a los proyectos en backend.
- **/trunk**
  - **/java** #Proyectos de java
  - **/cade** #Cadenas de ejecución (.sh)
  - **/scripts** #Scripts de base de datos, PL/SQLs (.sql)
  - **/c** #Proyectos de ProC
  - **/cobol** #Proyectos de Cobol
  - **/pcl** #Ficheros de macros para impresión
- **/branches**
- **/tags**

/estatico # carpeta contenedora de los recursos estáticos del proyecto
- **/trunk**
- **/branches**
- **/tags**

/config # carpeta contenedora de la plantilla de configuración dependiente de entorno
- **/trunk**
  - **/local**# Contiene los proyectos de Eclipse config de local
    - **/serapp** # carpeta con la plantilla y XMLs dependientes de capa backend
      - **/ xxxConfig** # carpeta con la plantilla y XMLs dependientes de entorno
    - **/backend** # carpeta con la plantilla y XMLs dependientes de entorno
  - **/serapp** # carpeta con la plantilla y XMLs dependientes de entorno de capa aplicación
    - **/ xxxConfig** # carpeta con la plantilla y XMLs dependientes de entorno
  - **/backend** # carpeta con la configuración dependiente de entorno de capa backend
    - **/desarrollo**
      - [/ xxxConfig] # opcional
      - *.properties
    - **/pruebas**
      - [/ xxxConfig] # opcional
      - *.properties
    - **/produccion**
      - [/ xxxConfig] # opcional
      - *.properties

/datos # carpeta datos del proyecto xxx
- **/trunk**
  - **/serapp**
    - **/file**
    - **/lis**
  - **/backend**
    - **/file**
    - **/lis**
- **/branches**
- **/tags**

```
/documentacion
        /trunk
        /branches
        /tags

/mobile
        /trunk
             /hybrid
             /native
        /branches
        /tags

/tags # carpeta de uso exclusivo para AntHillPro
        /anthillpro
/...   # resto de repositorios para diferentes proyectos
```

Es importante recalcar que la disposición de un repositorio (en este caso repositorio **xxx**) es una elección inamovible, la estructura detallada anteriormente se ha de mantener siempre.

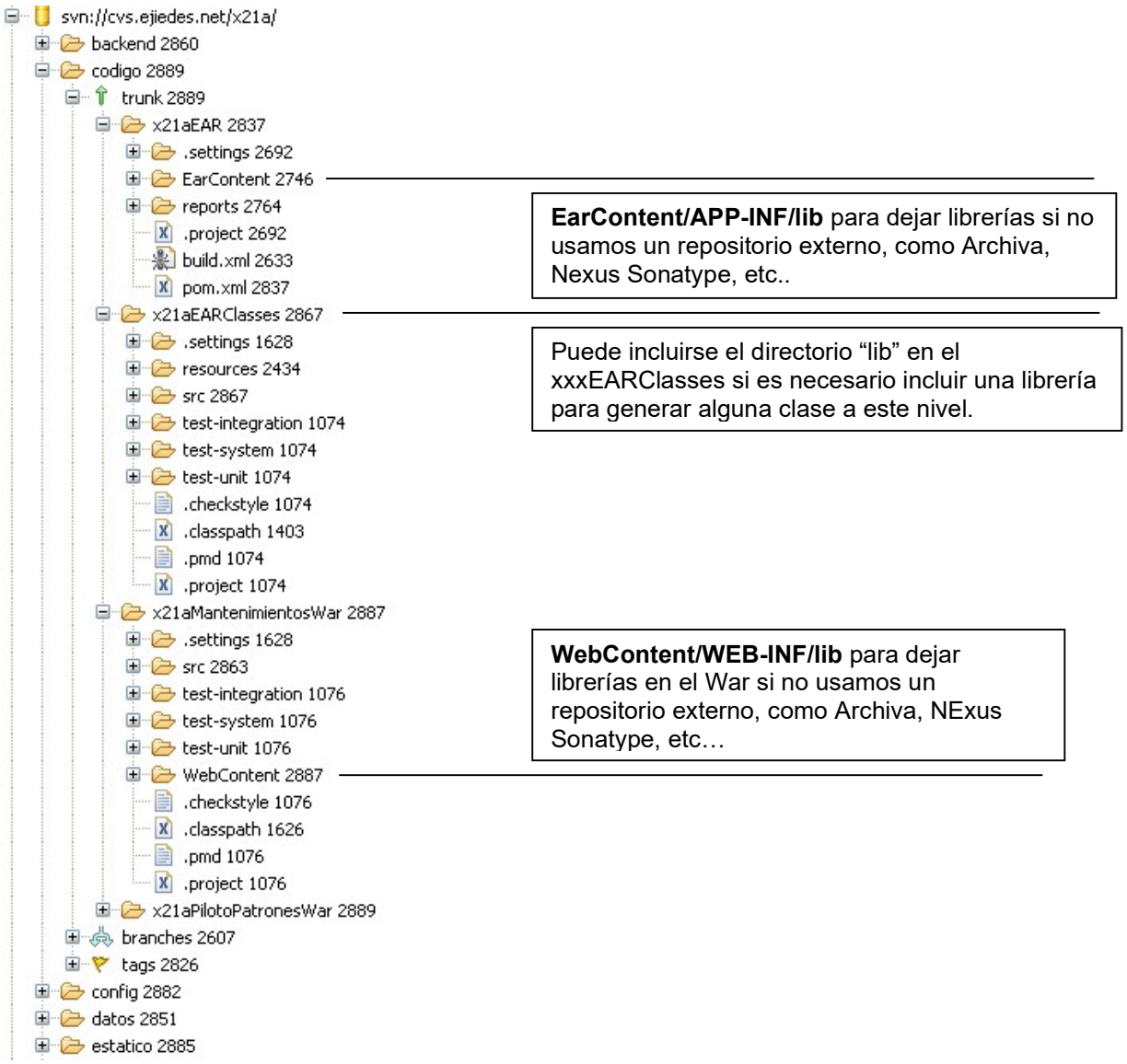
El contenido de las carpetas principales si se puede reorganizar en cualquier momento puesto que las ramas y las etiquetas son directorios normales, es decir, se pueden mover o renombrar como se desee.

Cambiar de una disposición a otra es sólo una cuestión de ejecutar una serie de movimientos en el lado del servidor; si se considera que la organización no es la adecuada sólo habrá que ir moviendo los directorios.

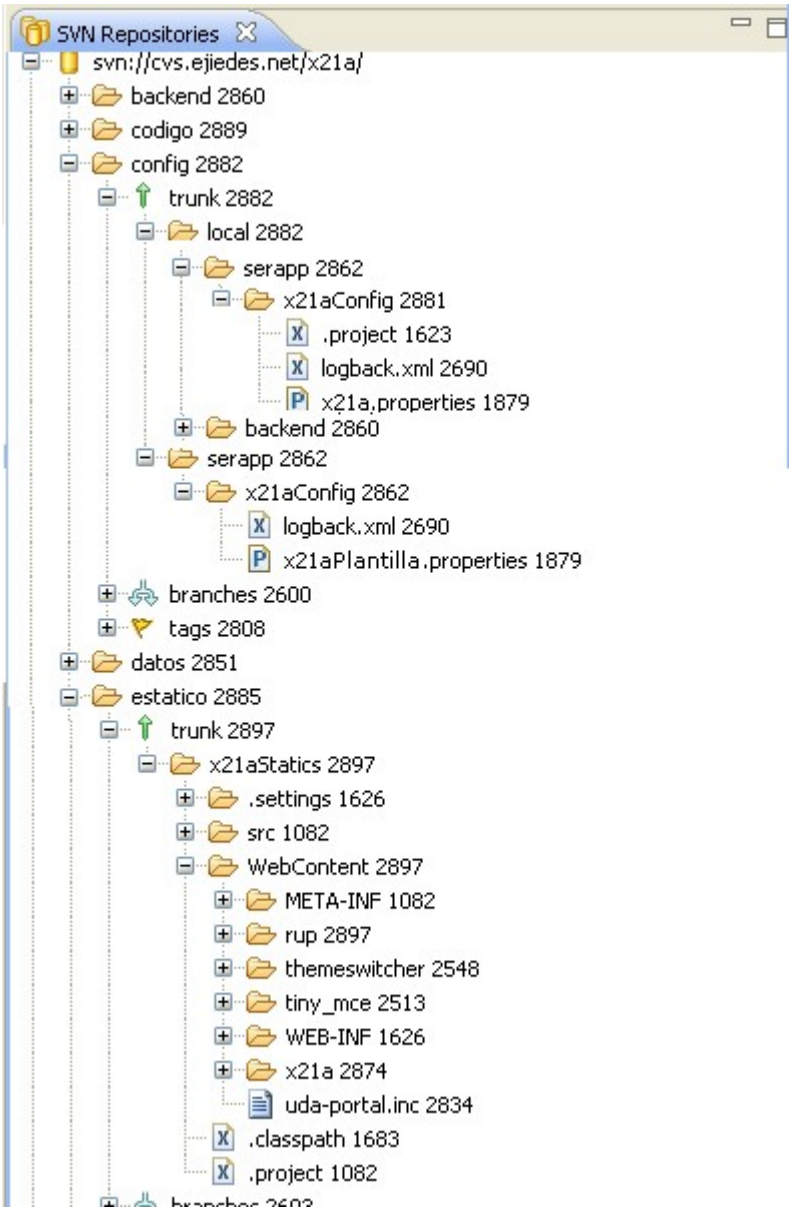
Existe la posibilidad de realizar estas tareas de cambio de disposición a través de los diferentes tipos de clientes de Subversion (cliente de línea de comandos de Subversion, plug-in Subversive de Eclipse para el entorno J2EE o bien TortoiseSVN para cualquier entorno).

Será necesario disponer de un usuario autorizado (introducido mediante AHP o gracias a los scripts que hemos visto en el apartado anterior; acceso lectura/escritura establecido para todos los usuarios) en el repositorio para poder realizar la modificación de su estructura así como para poder realizar el resto de tareas relativas al trabajo con un repositorio de Subversion.

A continuación se muestra la forma de realizar los cambios de distribución en la estructura de un repositorio Subversion con los clientes considerados para los distintos entornos.
Ejemplo codigo:



Ejemplo estatico y config:



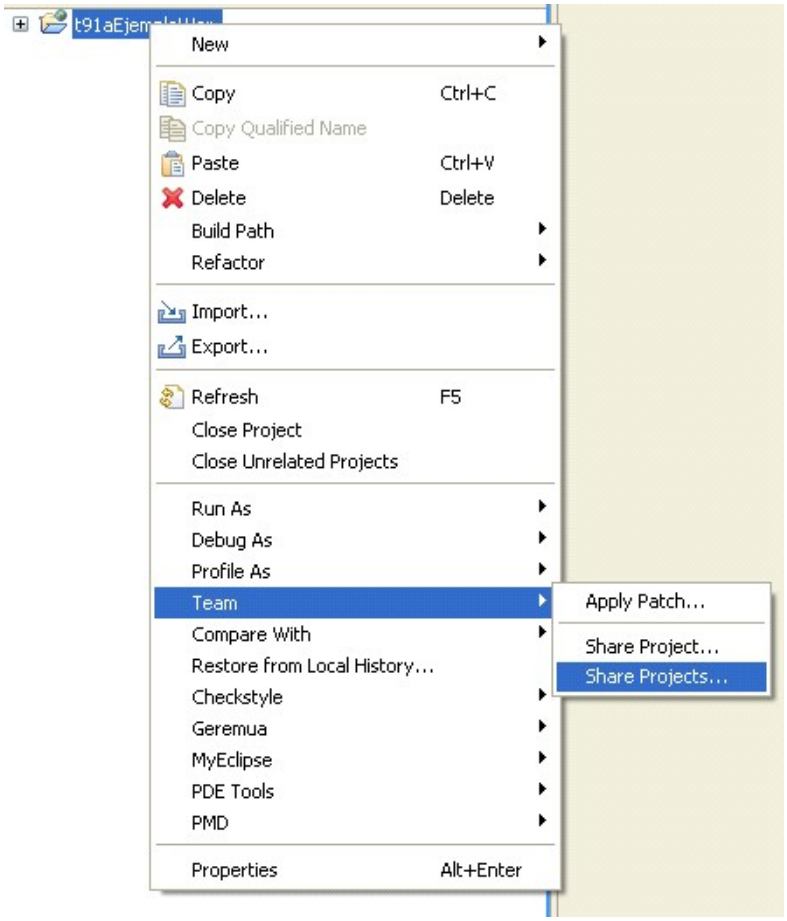
3.2.1. Entorno J2EE

Considerando el entorno de trabajo J2EE y utilizando como cliente de Subversion el plug-in Subversive para Eclipse se va a mostrar cómo alojar un proyecto en un repositorio bajo una estructura concreta y cómo poder modificar posteriormente dicha estructura.

Supuesto creado el repositorio **t91a** bajo la ubicación *svn://172.16.0.39* (que, en este caso, identifica al repositorio t91) y partiendo del (sub)proyecto de ejemplo **t91aEjemploWar**, proyecto localizado en el workspace del entorno de desarrollo Eclipse, se va alojar el dicho proyecto **t91aejemploWar** bajo la estructura:

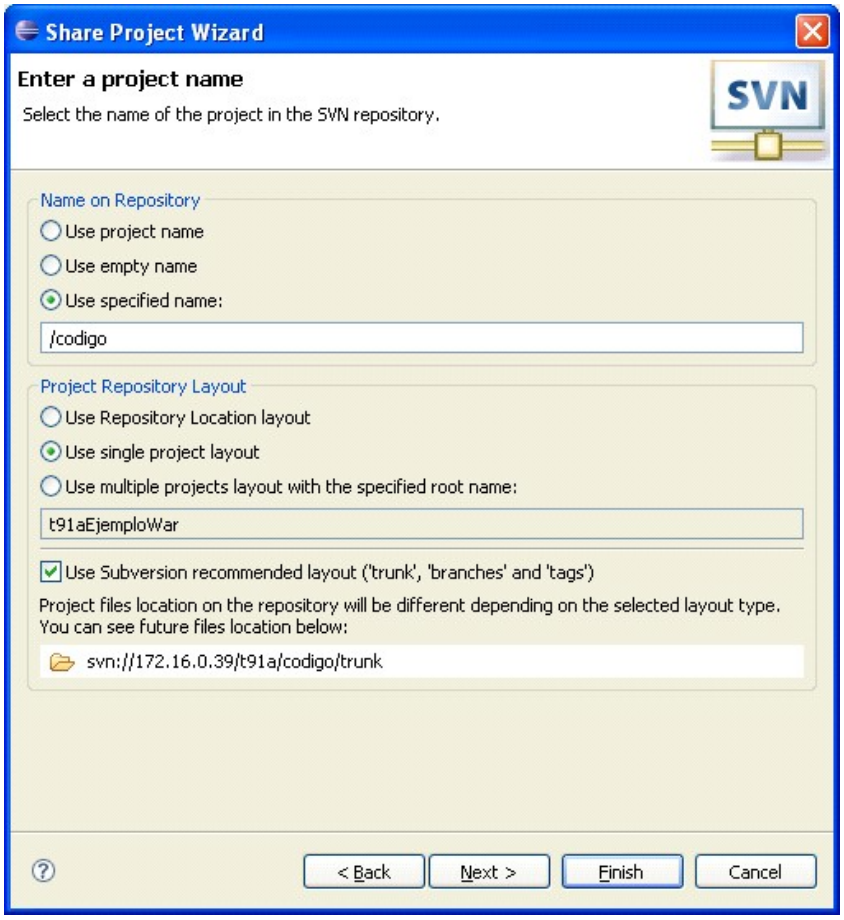
repositorio
/t91a
/codigo
/trunk
/t91aEjemploWar

Seleccionando el proyecto **t91a** y haciendo uso de la opción *Team* > *Share Projects* proporcionada por el plug-in Subversive alojaremos el proyecto en el repositorio **t91**:



En la ventana que nos aparece a continuación podremos crear la estructura inicial en la que alojar el proyecto. Seleccionaremos la opción “Use specified name” e introduciremos el valor **t91a/codigo**. Se seleccionará la distribución del repositorio “Use single project layout” y se podrá observar cómo la ubicación final, para el ejemplo señalado, será por tanto:

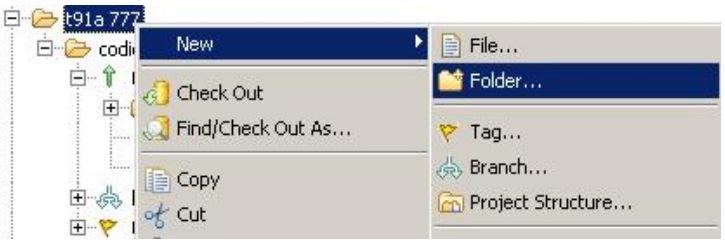
svn://172.16.0.39/t91a/codigo/trunk



El aspecto del proyecto alojado en el repositorio tendrá, utilizando la perspectiva SVN *Repository Exploring*, el siguiente aspecto:

Desde esta misma perspectiva y debido, por un lado, a la flexibilidad ofrecida por Subversion y, por otro, al buen funcionamiento del plug-in Subversive se podrá modificar la estructura definida para este repositorio desde la perspectiva *SVN Repository Exploring*.

Al situarnos encima del proyecto alojado en el repositorio (vista *SVN Repositories*) y desplegando el menú contextual podremos, entre otras posibilidades, crear nuevos directorios:



Nota: Hay que tener en cuenta que las copias locales de trabajo de cada desarrollador sincronizadas con el repositorio antes de realizar cambios de estructura en éste tendrán que apuntar al proyecto alojado al repositorio en base a la nueva estructura (sincronizarse con la nueva disposición del proyecto). Para ello se dispone de la opción *Switch…* proporcionada por el plug-in Subversive.

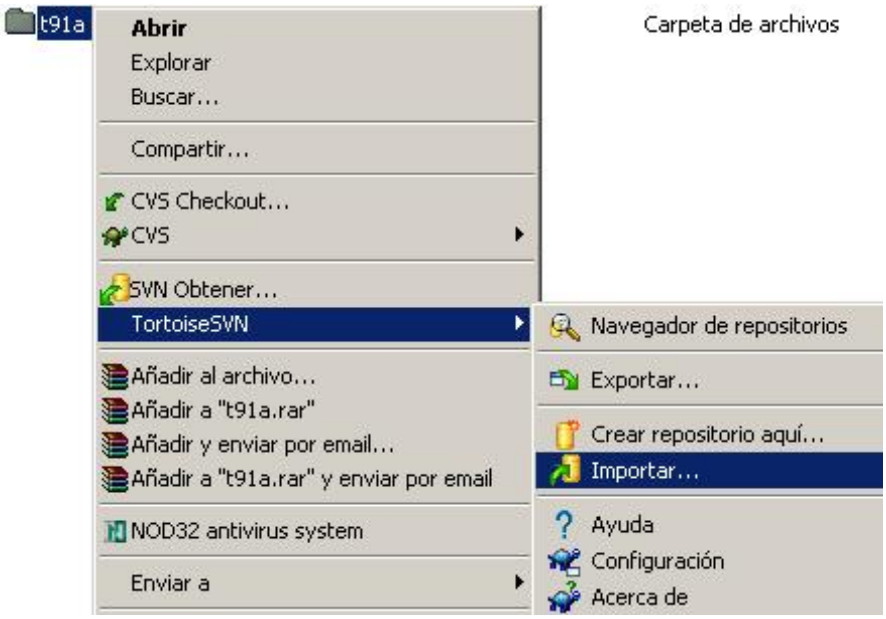
3.2.2. Independencia del entorno

Considerando cualquier entorno de trabajo y utilizando como cliente de Subversion la herramienta TortoiseSVN se va a mostrar, al igual que en el apartado anterior, cómo alojar un proyecto en un repositorio bajo una estructura concreta y cómo poder modificar posteriormente dicha estructura.

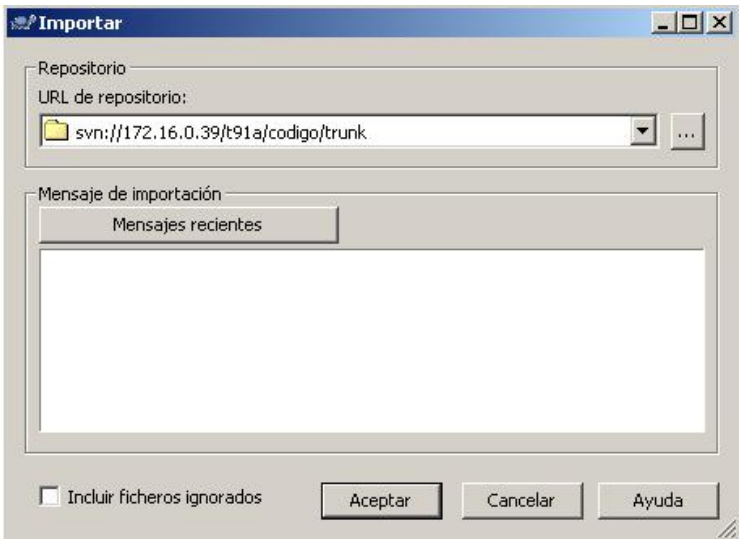
De igual manera, supuesto creado el repositorio **t91a** bajo la ubicación *svn://172.16.0.39* (que en este caso identifica al repositorio t91a) y partiendo del (sub)proyecto de ejemplo **t91aEjemploWar** y utilizando el explorador de Windows se va alojar el dicho proyecto **t91a** bajo la estructura:

/t91a
/codigo
/trunk
/t91aEjemploWar

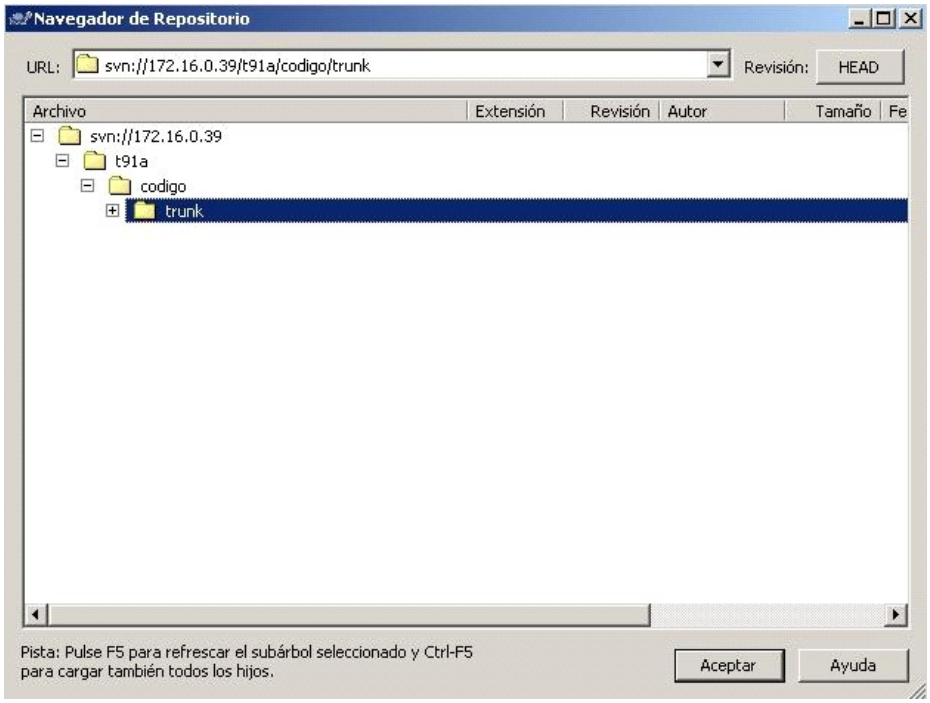
Seleccionando el proyecto **t91aEjemploWar** y haciendo uso de la opción *TortoiseSVN* > *Importar…* proporcionada por el menú contextual del explorador de Windows alojaremos el proyecto en el repositorio **t91a**:



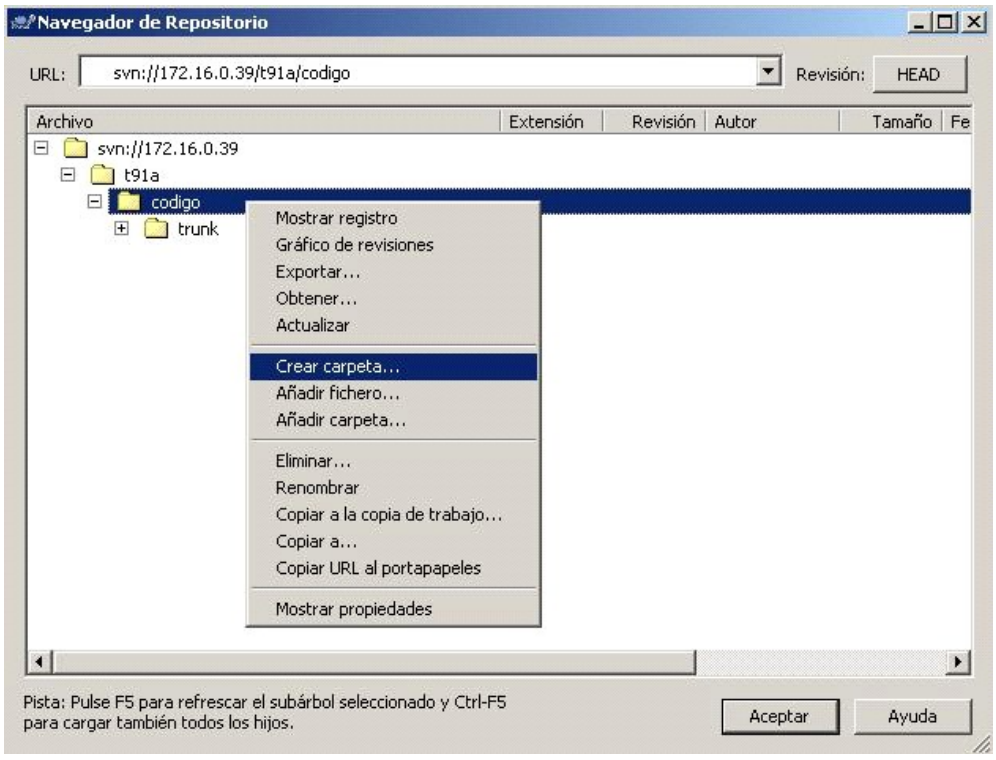
En la siguiente ventana se indicará la ubicación concreta donde se alojará el proyecto:



Con ayuda de la opción *TortoiseSVN* > *Navegador de repositorios* podremos observar la estructura del repositorio:

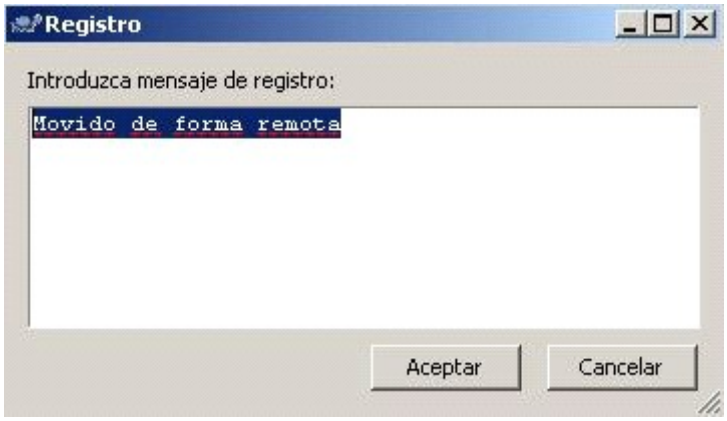


Desde esta ventana *Navegador de Repositorios* podremos realizar las operaciones necesarias de modificación de la estructura. En este caso crearemos una nueva carpeta *t91a* situada debajo de las carpeta *codigo*:

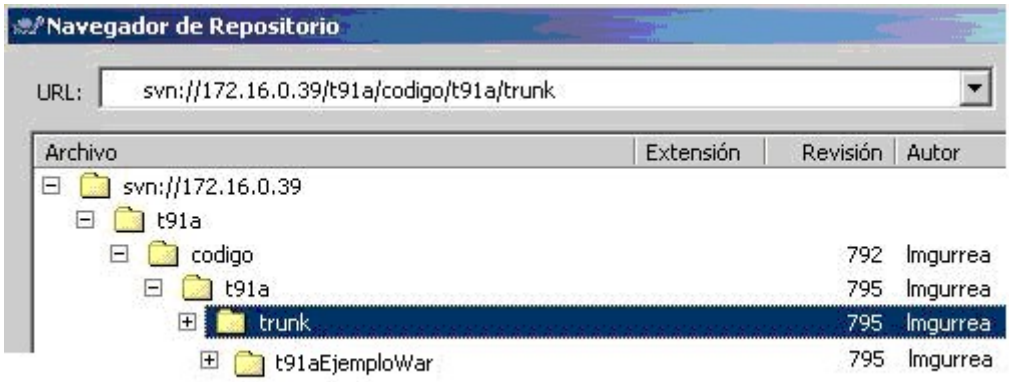


Y se introducirá en ella la carpeta *trunk* y todo su contenido (seleccionarla y arrastrarla):

Por defecto, se introduce el mensaje de registro “Movido de forma remota” que nos almacenará la información de la operación realizada:



La estructura final, por tanto, será la siguiente:



Nota: Al igual que en el apartado anterior, las copias locales de trabajo de cada desarrollador sincronizadas con el repositorio antes de realizar cambios de estructura en éste tendrán que apuntar al proyecto alojado al repositorio en base a la nueva estructura. Opción *TortoiseSVN > Cambiar….*

3.3 Creación de Etiquetas

La forma de establecer las versiones finales en el repositorio Subversion se realizará mediante *tags*. Las versiones de los diferentes módulos en el Subversion se distinguirán a través de su nombre. A continuación se muestra un ejemplo de nomenclatura:

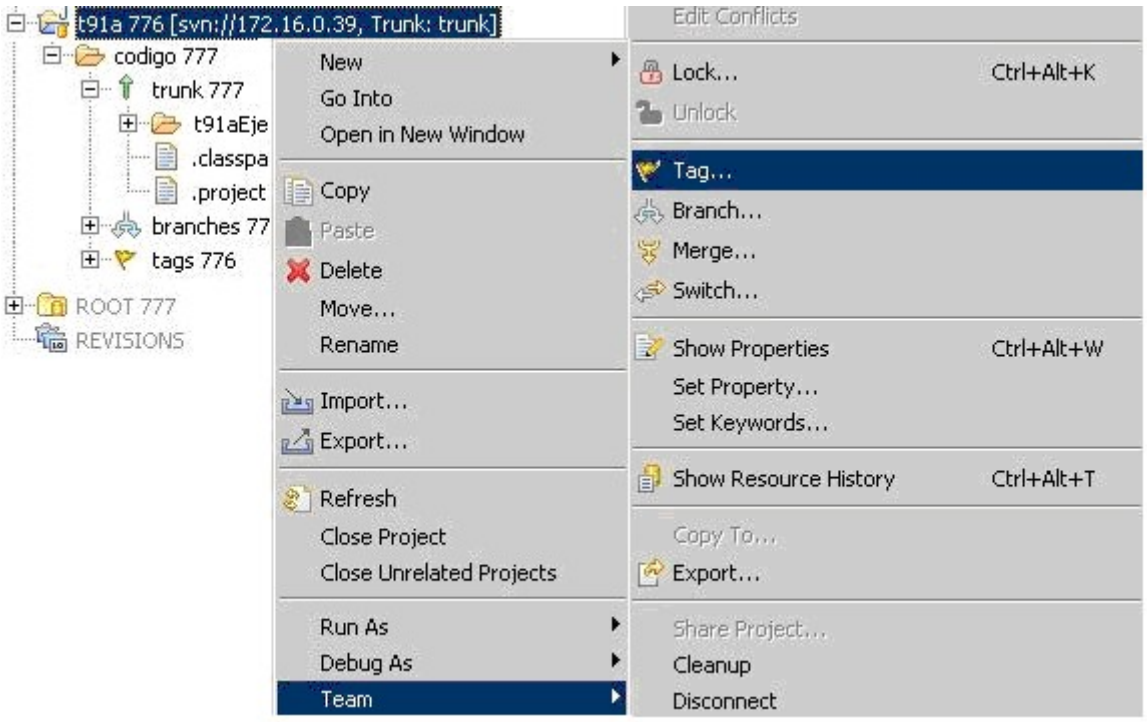
```
aaa_X.Y
```

donde “aaa” es código de aplicación, “X” indica el número de versión e “Y” indica el número de la sub-versión. Esta sería la forma más clara a la hora de denominar y así poder diferenciar cada una de las versiones

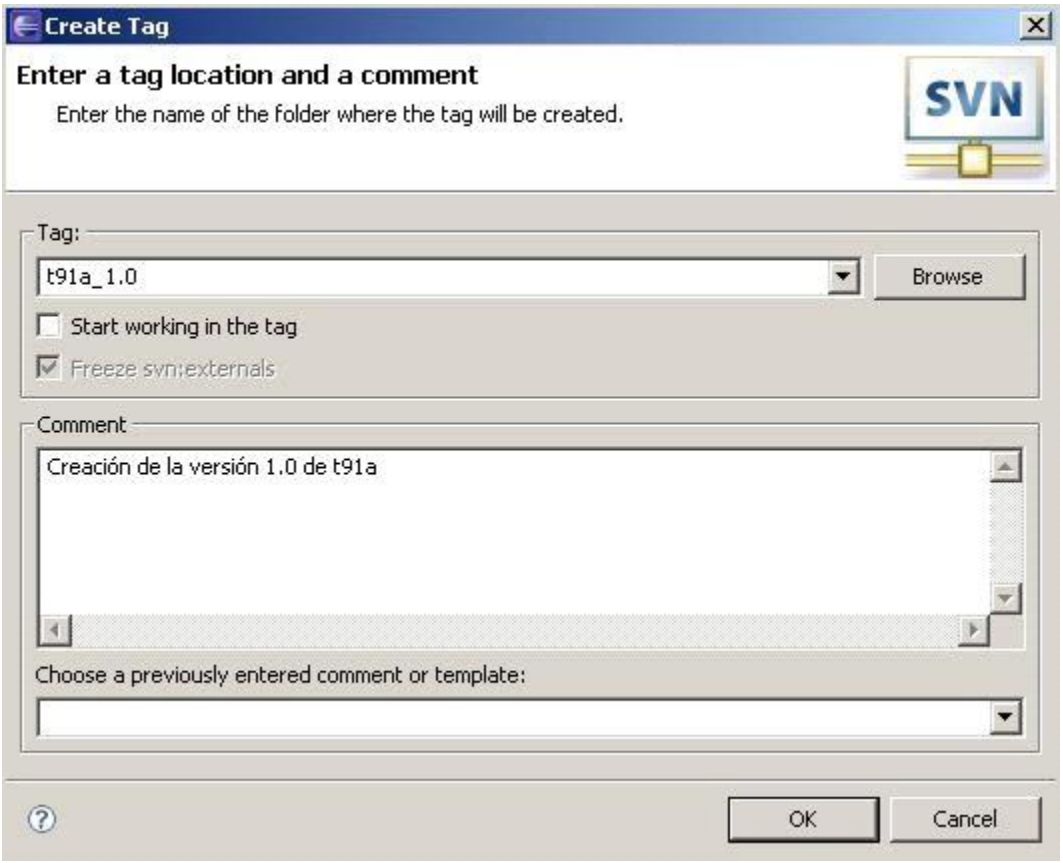
finales (tags). Pero debido a las diferentes necesidades de las distintas aplicaciones que van a alojarse en los repositorios Subversion podrá denominarse cada versión final (*tag*) con el nombre que se estime oportuno.

3.3.1. Entorno J2EE

Para crear una etiqueta a partir de una copia local de trabajo con ayuda del plug-in Subversive habrá que hacer uso de la opción *Team* > *Tag*….



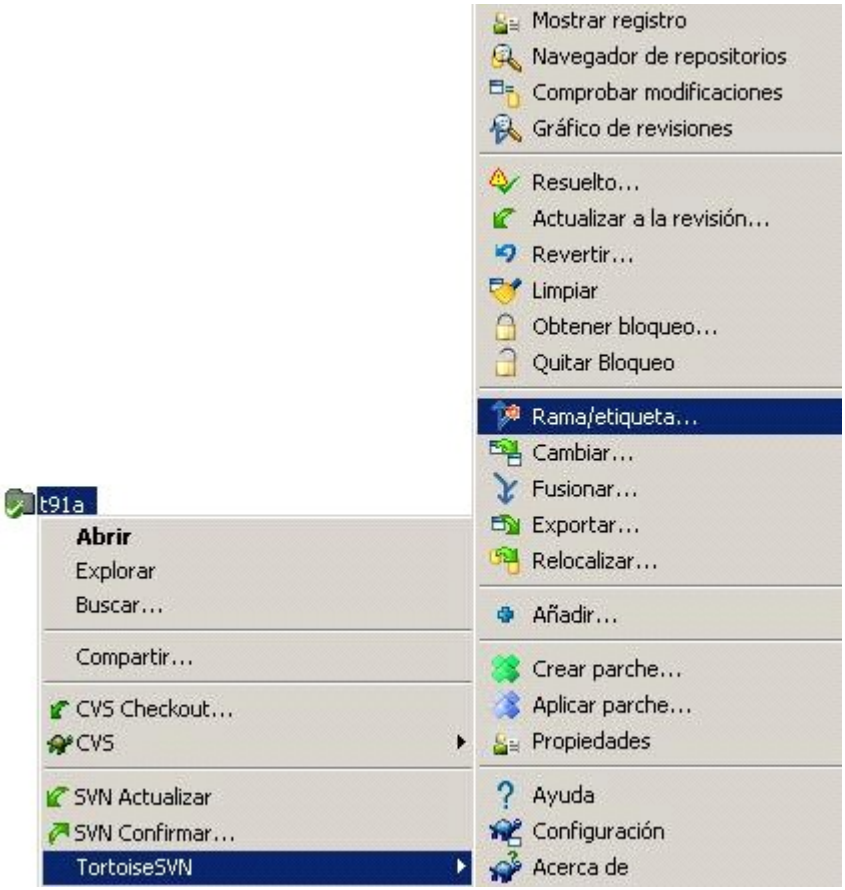
Sólo habrá que indicar el nombre de la etiqueta como se muestra en la siguiente figura:



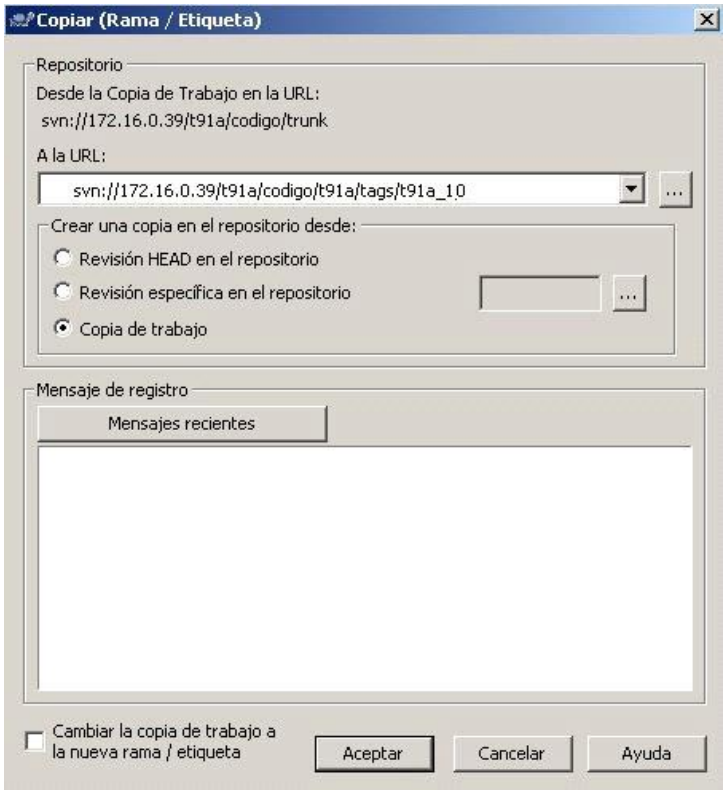
La estructura del repositorio con la nueva versión considerada será la siguiente:

3.3.2. Independencia del entorno

Para crear una etiqueta a partir de una copia local de trabajo con ayuda de la herramienta TortoiseSVN habrá que hacer uso de la opción *TortoiseSVN > Rama/Etiqueta…*.



E introducir correctamente el nombre de la versión considerada (svn://172.16.0.39/t91a/codigo/t91a/tags/t91a_1.0) en la ventana siguiente:



La estructura del repositorio, por tanto, quedará como se muestra a continuación:

4 Tarea Ant para Weblogic 8

Se trata de la tarea de ant que realiza el despliegue del proyecto, del que el usuario es responsable, desde el servidor de subversión al entorno de desarrollo. A continuación se detallan los parámetros que componen la tarea ant, y que deberán ser informados por el usuario

- **DrepoSVN ->** Nombre del Repositorio.
- **DuserSVN ->** Codigo usuario responsible aplicación.
- **DpassSVN ->** Password del usuario.
- **DtagSVN ->** Nombre de versión final (tag) a desplegar.
- **DtypeSVN->** Tipo de obtención (Siempre ha de tener el valor **Server**).
- **DcarpetaSVN ->** indicador de que es lo que se desea descargar.
  - **all ->** Descarga todo el repositorio.
  - **c ->** Descarga la carpeta código.
  - **s ->** Descarga la carpeta scripts.
  - **e** -> Descarga la carpeta estático.
  - **l ->** Descarga la carpeta librerías.
  - **d ->** Descarga la carpeta datos.
  - **f ->** Descarga la carpeta config.

En caso de que se quiera descargar todo e contenido del repositorio, de por ejemplo el proyecto t71, la invocación a realizar seria la siguiente:

```
ant    -DrepoSVN=t71a -DuserSVN=t71a -DpassSVN=t71a -DtagSVN=t71a_0.1 -
       DtypeSVN=server -DcarpetaSVN=all svn
```

Si lo que se desea en cambio es la obtención del código de la aplicación , la llamada seria

```
ant    -DrepoSVN=t71a -DuserSVN=t71a -DpassSVN=t71a -DtagSVN=t71a_0.1 -
       DtypeSVN=server -DcarpetaSVN=c svn
```

5 Tarea Ant para Weblogic 11

Se trata de la tarea de ant que realiza el despliegue del proyecto, del que el usuario es responsable, desde el servidor de subversión al entorno de desarrollo. A continuación se detallan los parámetros que componen la tarea ant, y que deberán ser informados por el usuario

- **Dsvn.app.url ->** Url del codigo que queremos mover al servidor de desarrollo
- **Dsvn.app.user ->** Codigo usuario responsible aplicación.
- **Dsvn.app.password ->** Password del usuario.
- **Dsvn.app.revision ->** Revision a mover al servidor de desarrollo.

En caso de que se quiera descargar todo e contenido del repositorio, de por ejemplo el proyecto w84b, la invocación a realizar seria la siguiente:

ant installApp -Dsvn.app.url=svn://cvs.ejiedes.net/w84b/codigo/trunk -Dsvn.app.user=w84b -Dsvn.app.password=w84b -Dsvn.app.revision=HEAD

Si lo que se desea en cambio es la obtención del código de la aplicación , la llamada seria

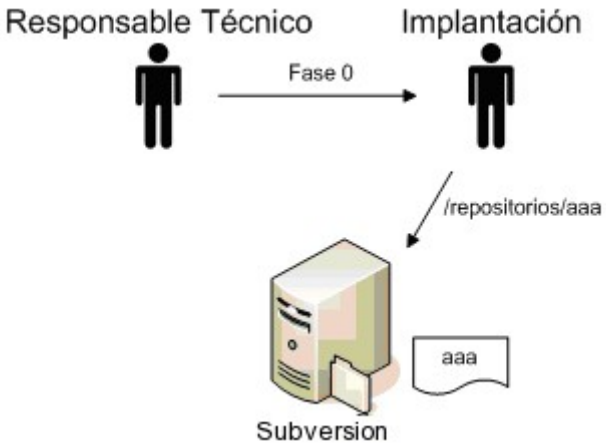
ant checkout -Dsvn.app.url=svn://cvs.ejiedes.net/w84b/codigo/trunk -Dsvn.app.user=w84b -Dsvn.app.password=w84b -Dsvn.app.revision=HEAD

Estas tareas ant se centran en la carpeta “código” de la estructura del svn. El resto de tareas ant serán informadas según se vayan creando.

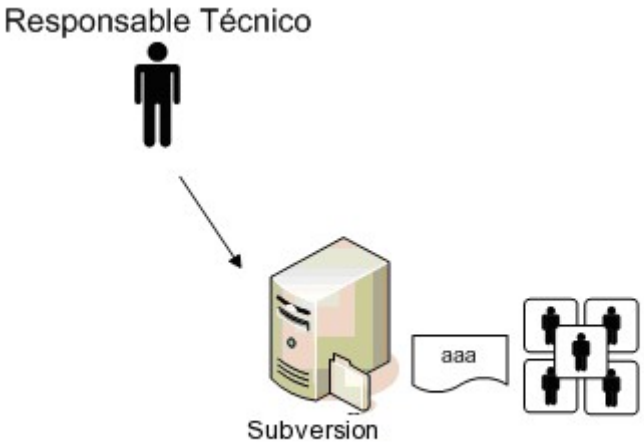
6 Ejemplo Practico

Los pasos generales a seguir durante la evolución de un proyecto respecto al Subversión serían:

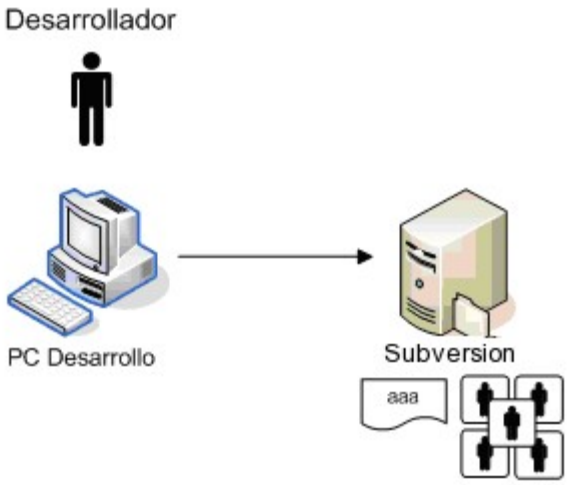
1. Solicitud del repositorio de código para la aplicación en la Fase 0.



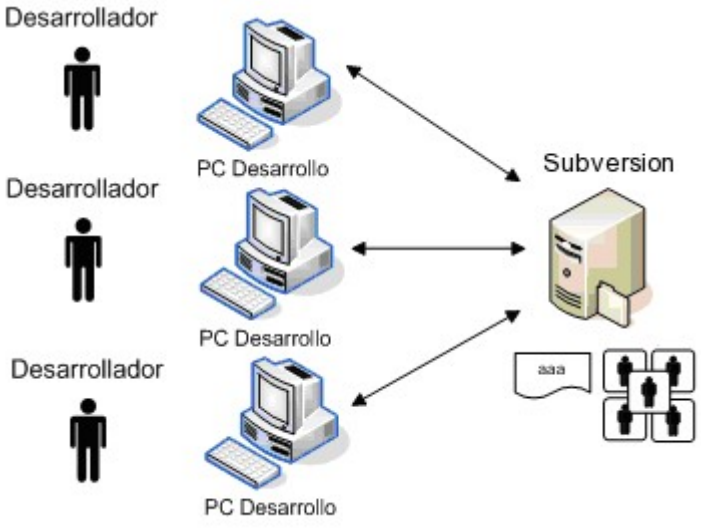
2. El Responsable Técnico crea un usuario en el repositorio para cada desarrollador



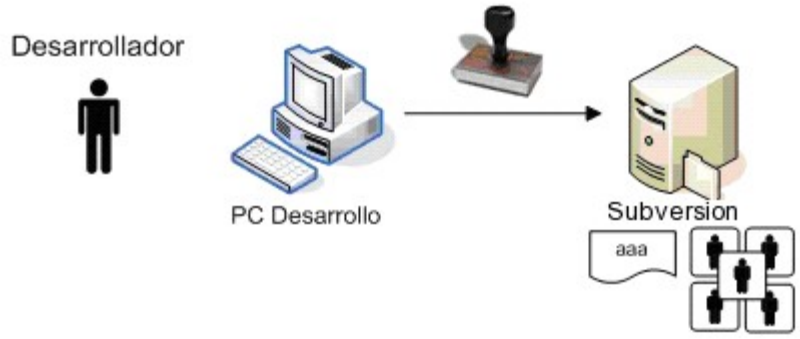
3. Con el usuario del repositorio suministrado por el Responsable Técnico, un desarrollador crea el proyecto inicialmente en el PC y lo publica en el repositorio de código.



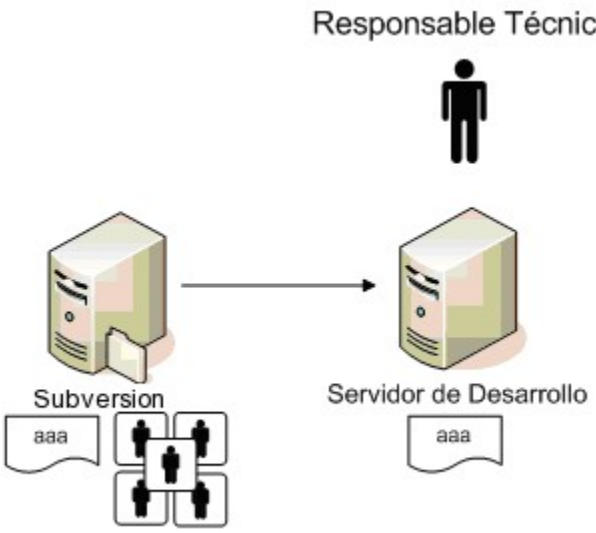
4. El resto de desarrolladores obtienen el código de la aplicación del Subversion y van sincronizando los cambios según se va avanzando en el desarrollo.



5. El desarrollo está listo para ser probado en el Servidor de Desarrollo. Hay que versionar la aplicación en el Subversion.



6. La versión marcada por el grupo de desarrollo es recepcionada en el Servidor de Desarrollo por el Responsable Técnico de la aplicación.



7. El responsable Técnico lanza tareas de compilación, despliegue y creación de recursos en el Servidor de Desarrollo.
8. Si las pruebas funcionales han sido satisfactorias se realiza el traspaso a Pruebas tras haber hecho la reunión de Fase 1 por el procedimiento habitual.

7 Anexo I. Solicitud de intervención Subversion

Con objeto de simplificar las tareas de administración, se utilizará un solo documento de solicitud que recoge todos los tipos de actuación posibles (altas), de este modo se incluirá una copia del formulario por cada tipo de actuación requerida.

La solicitud de intervención en Subversion se dirigirá al grupo de soporte software, teniendo en cuenta lo siguiente:

- Soporte recibirá la siguiente petición de alta de repositorio de un determinado proyecto
- Sólo se podrán recibir y atender peticiones referentes a la **creación de repositorios de proyectos**, sobre los proyectos de los que el **peticionario sea responsable.**

Como resultado se deberá informar de los siguientes datos:

- Ruta del repositorio
- Nombre de la máquina y puerto donde se ha creado el repositorio
- Usuario y Contraseña de acceso a la máquina del solicitante

SOLICITUD DE INTERVENCIÓN EN SUBVERSION

***Fecha de solicitud:**	
***Tipo de actuación:**	☒ **ALTA**
***Nombre Repositorio:**	
Solicitante *Nombre:	
***Departamento:**	
***Código Aplicación:**	

*-Los campos marcados con * son obligatorios*

<u>Para el Administrador</u>
- El campo "Nombre Repositorio" ser el nombre con el que se creara el repositorio. El numero máximo de caracteres es de 12.
- El campo "Código Aplicación" será con el que se cree el usuario de acceso a la maquina de Subversión
- El campo "Departamento" se corresponderá con el grupo al que pertenecerá el usuario.

8 Anexo II. Utilidades para SVN en AntHill Pro

En la herramienta AntHill Pro se han expuesto una serie de utilidades que nos van a permitir validar la estructura de SVN, dar de alta usuarios en SVN y activar y desactivar hooks.

8.1 Utilidades SVN – Validar estructura SVN

Utilidad creada para validar la estructura de SVN según los estándares definidos en la actualidad

Datos de entrada.

Aplicativo: Código del aplicativo. Requerido.
Tipo: [weblogic8|weblogic11]. Requerido.
Usa diccionario?: Indica si la configuración del aplicativo para la capa Servidor de Aplicaciones es generada por Diccionario.
Config en Dominio?: Indica si la configuración del aplicativo para la capa Servidor de Aplicaciones sigue la estructura en disco /config/$DOMINIO/$APP, siendo $APP el aplicativo y $DOMINIO el dominio de despliegue.
Dominios: Dominio de despliegue del aplicativo; en el caso de varios, separar por el caracter ','. Requerido.
Carpeta estaticos Web: Carpeta que contiene el contenido estático de la capa Web. Opcional.

- Ej:

```
Aplicativo: y98a
Tipo: weblogic11
Usa diccionario?: true
Config en Dominio?: true
Dominios: wl11_inter_apps_dpto
Carpeta estáticos Web: alias
```

Datos de salida.

- **Excepción**:
  Si existen varios proyectos AHP con nombre = Código Aplicativo.
  Si no encontrado el proyecto AHP con nombre = Código Aplicativo.
  Si el usuario intenta validar la estructura de un proyecto para el que no esta autorizado.
- **Salida sin error**:
  Listado de carpetas definidas por el estándar y que no existen el el repositorio SVN

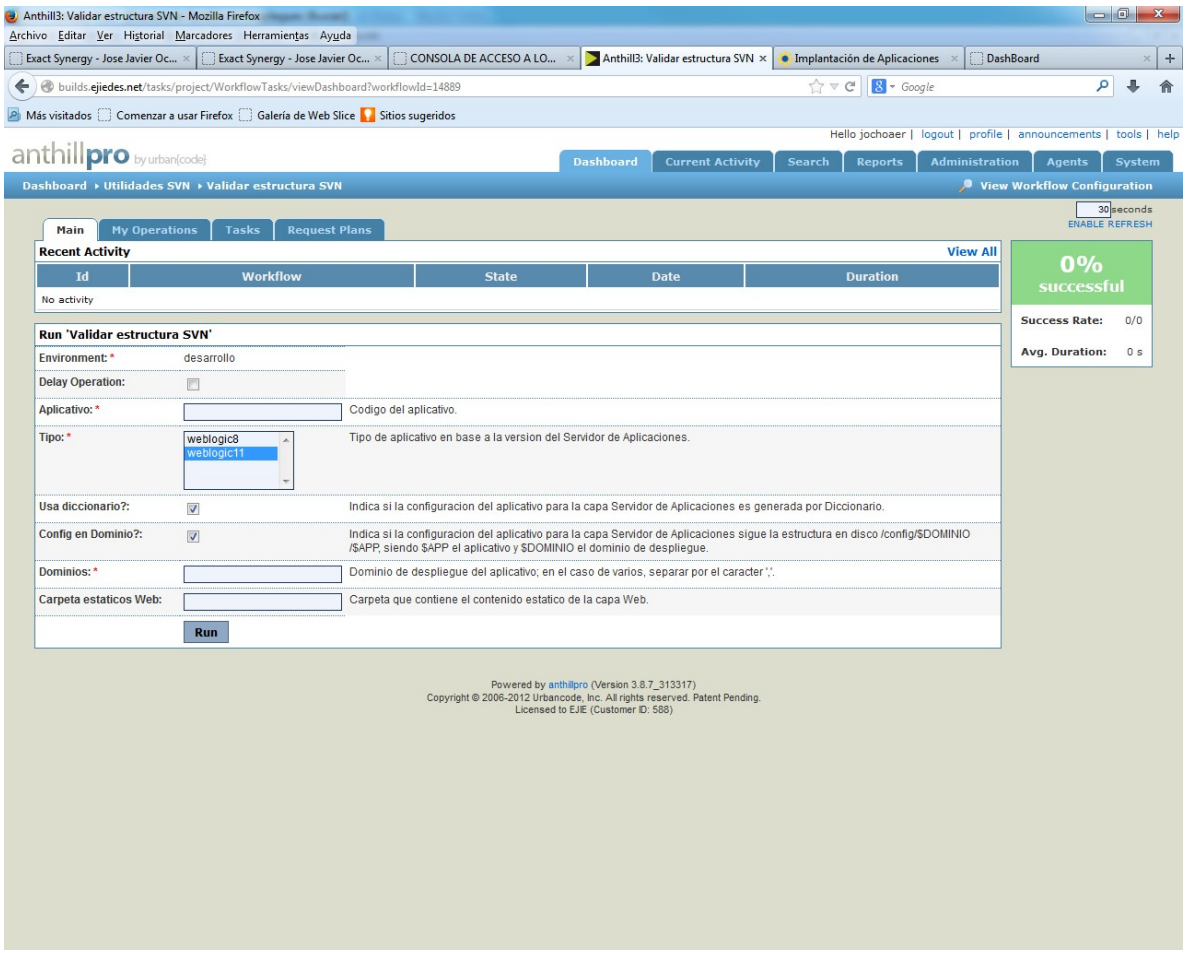
- Ej:

```
================================
command output:
Comando: /usr/ebin/soporte-unix/svn_check_weblogic_structure_ni.sh -a "y98a" -w "11" -
g "si" -c "si" -d "wl11_inter_apps_dpto" -f "alias"
y98a:x:9051:636:Registro de Centros Alimentarios:/web/repositorios_svn/bin/:/bin/bash

Aplicacion: y98a
Version de Weblogic: 11
Diccionario: si
Config en Dominio: si
Dominios: wl11_inter_apps_dpto
Alias: alias

Directorios que no existen en la estructura de SVN.....

estatico/trunk/alias

================================================================================
====
```

Notas:

- Asumimos que el usuario de SVN es el código de aplicativo.

Pantalla AHP:



8.2 Utilidades SVN - Alta usuario SVN

Utilidad que sirve dar de alta usuarios nuevos en SVN sin tener que utilizar el menú del servidor de SVN

Datos de entrada.

Aplicativo: Codigo del aplicativo. Requerido.

Usuario: Usuario a dar de alta en SVN. Requerido.

Password: Contraseña del usuario a dar de alta en SVN. Requerido.

- Ej:

Aplicativo: spaplic

Usuario: popeye

Password: olivia

Datos de salida.

- **Error**:

Si usuario logado en AHP no está autorizado para dar de alta usuario.

Si el repositorio SVN no existe.

Si el usuario ya está dado de alta.

- Salida sin error:

Mensaje de que el usuario ha sido dado de alta.

- Ej:

==========

command output:

Comando: /usr/ebin/soporte-unix/svn_create_user.sh -r "spaplic" -u "popeye" -p "****"

Usuario popeye creado en el repositorio spaplic

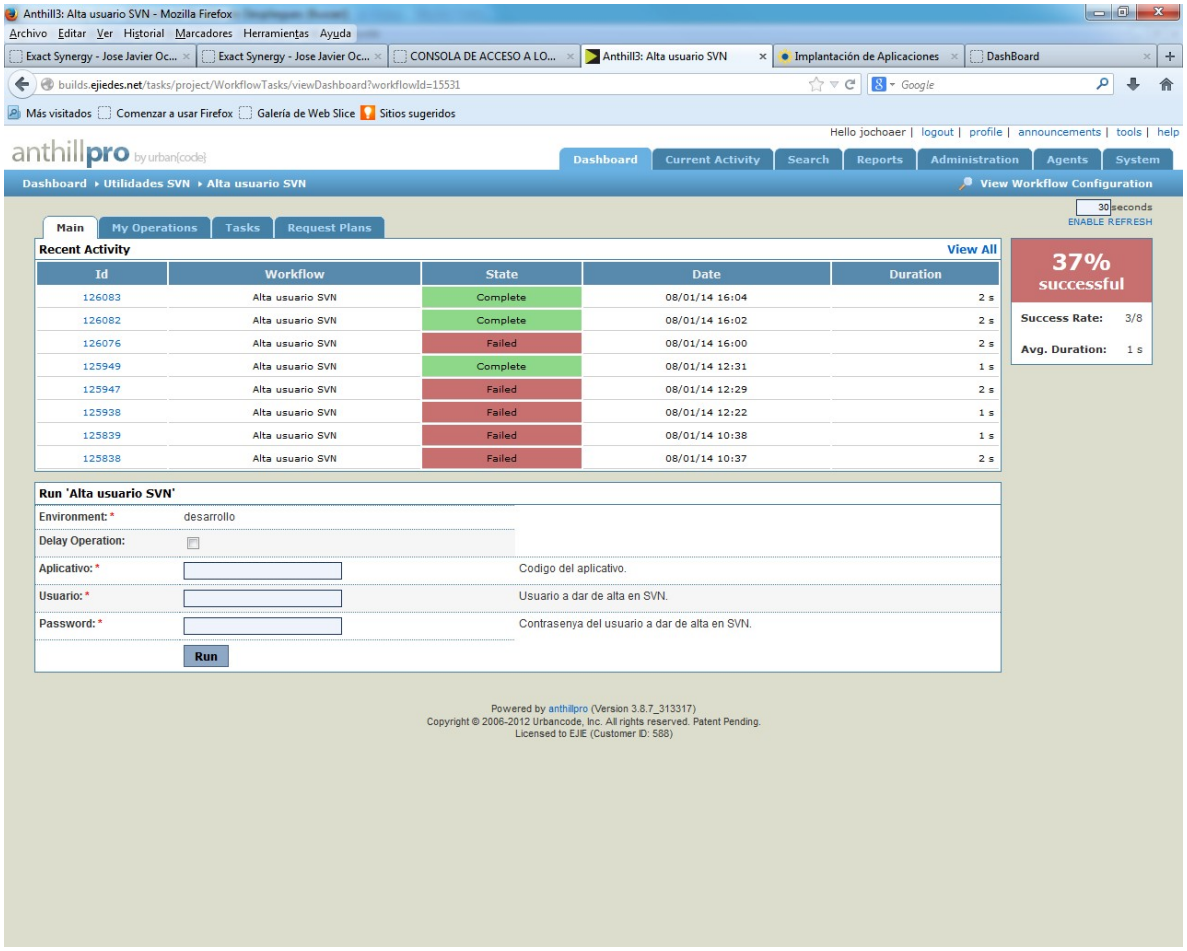
==========

Notas:

- Asumimos que el código de aplicativo coincide con el nombre del repositorio y nombre del proyecto AHP.

- La password la ocultamos en la salida de AHP.

Pantalla AHP:



8.3 Activación/Desactivación Hooks SVN

Utilidad que permite la administración (Activación/Desactivación) de los diferentes hooks que existen en el SVN de cada aplicación:

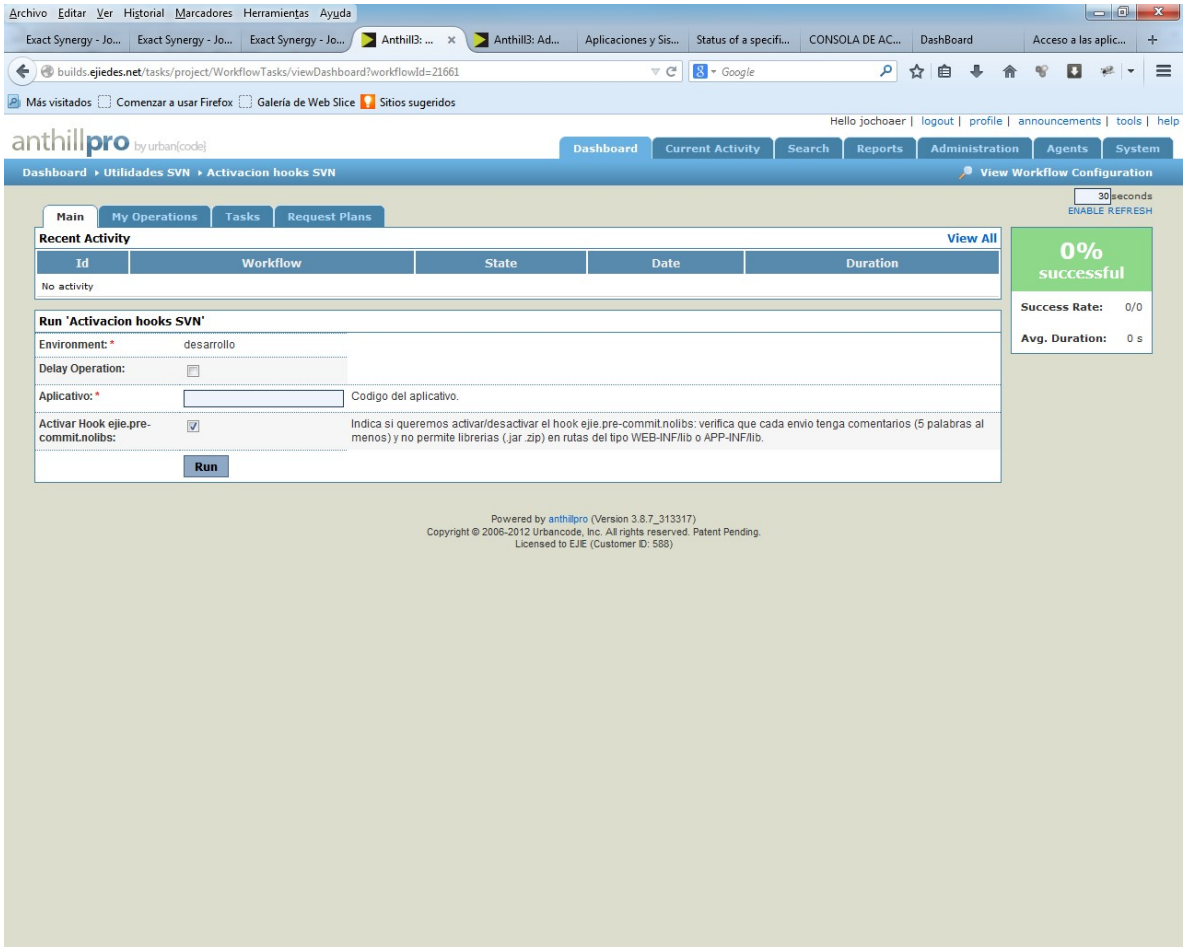
Datos de entrada

Aplicativo: Código de aplicación **Requerido**

Activar hook ejie.precommit.nolibs.: Acitvo/desactivo **Requerido**

Indica si queremos activar/desactivar el hook ejie.pre-commit.nolibs: verifica que cada envio tenga comentarios (5 palabras al menos) y no permite librerias (.jar .zip) en rutas del tipo WEB-INF/lib o APP-INF/lib.

Pantalla AHP



9 Anexo III. Gestión de hooks en el repositorio

Es posible asociar hooks a nuestro repositorio. Básicamente existen 3 tipos de hook asociados a estas **acciones**:

- Bloqueo/Desbloqueo de un recurso gestionado por el repositorio.
- Al realizar un cambio en el repositorio.
- Al realizar un cambio en una propiedad de un recurso gestionado por el repositorio.

Para cada una de estas acciones, podemos definir un **tipo** de hook, relacionado con el momento en que se realiza la acción :

- **pre-unlock** : Ejecutado antes de un desbloqueo.
- **post-unlock** : Ejecutado después de un desbloqueo.
- **pre-lock** : Ejecutado antes de un bloqueo.
- **post-lock** : Ejecutado después de un bloqueo.
- **start-commit** : Ejecutado antes de que la transacción sea creada dentro del proceso de realizar un cambio.
- **pre-commit** : Ejecutado antes de que la transacción sea ejecutada dentro del proceso de realizar un cambio.
- **post-commit** : Ejecutado después de que la transacción sea ejecutada.

- **post-revprop-change** : Ejecutado después de que una propiedad de revisión sea añadida, modificada o borrada.
- **pre-revprop-change** : Ejecutado antes de que una propiedad de revisión sea añadida, modificada o borrada.

Con el objetivo de facilitar su gestión y desarrollo, se han creado un par de scripts que permiten:

- Asociar de un catálogo de hooks existente uno de ellos a nuestro repositorio.
- Desasociar un hook de nuetro repositorio.

Lo que no evita que un responsable del repositorio pueda incluir sus propios hooks, pero consideramos que incluirlos en el catálogo permite su uso por otros usuarios y homogeniza el desarrollo de los mismos, que por defecto, y en este momento, estamos creándolos con scripting en shell.

Si queremos desarrollar un hook para el catáogo, sería necesario tener en cuenta:
Nombre del hook del tipo ORGANIZACIÓN.TIPO.DESCRIPCION

1. El hook debe ser un script de tipo “#!/bin/sh”
2. El hook debe tener este comentario: # DESCRIPCION: Descripción de lo que hace el hook..

Ejemplo: ejie.pre-commit.noop

```
#!/bin/sh
# PRE-COMMIT HOOK
# DESCRIPCION: Este hook no hace nada.

REPOS="$1"
TXN="$2"

SVNLOOK=/usr/local/bin/svnlook
GREP=/bin/grep

export LC_MESSAGES="es_ES"
# All checks passed, so allow the commit.
exit 0
```

Podemos solicitar la inclusión del hook en el catálogo vía petición a Soporte Software, incluyendo:

SOLICITUD DE INTERVENCIÓN EN SUBVERSION

***Fecha de solicitud:**	dd-mm-yyyy
***Tipo de actuación:**	☒ **ALTA HOOK EN CATALOGO**
***Nombre Hook:**	Máximo 8 caracteres a..z (Ej.- noop)
***Tipo de Hook:**	Eligir uno de estos: pre-unlock pre-revprop-change pre-lock pre-commit start-commit post-unlock post-revprop-change post-lock post-commit (Ej.- pre-commit)
***Breve descripción:**	Una línea que describa su funcionalidad (Ej.- Este hook no hace nada.)
***Organización:**	(Ej.- Ejie)

***Código del hook:** (Ej.- El código anterior)
*-Los campos marcados con * son obligatorios*

9.1.1. Scripts para la gestión de hooks

Existen dos scripts destinados a realizar la administración de hooks de los repositorios Subversion:

- Alta de un hook
- Baja de un hook

Estos scripts están enlazados desde el menu_ldap_SVN del punto anterior.

Nota: La administración de los hooks de los repositorios (ejecución de estos scripts) deberá realizarla cada responsable de proyecto con el usuario creado para tal fin codAplicacion/Dpto(usuario/grupo).

Alta de un hook

alta_hook_SVN.sh: Script relativo al menú de opciones para asociar un hook de los disponibles al repositorio.

La instrucción adecuada para lanzar este script es la siguiente:

$ sh /repositorios_svn/bin/ldap/alta_hook_SVN.sh

Este script permite asociar uno de los hooks disponibles a nuestro proyecto. Si es necesario crear un nuevo hook, será necesaria una petición al administrador del producto, que incluirá el hook en el catálogo de hooks disponibles una vez verificado su funcionamiento.

Baja de un hook

baja_hook_SVN.sh: Script relativo al menú de opciones para desasociar un hook de los disponibles en el repositorio.

La instrucción adecuada para lanzar este script es la siguiente:

$ sh /repositorios_svn/bin/ldap/baja_hook_SVN.sh

9.1.2. Listado de hooks existentes

Estos son los hooks existentes para dar de alta en un repositorio:
La sintaxis es: ORGANIZACION.TIPO.DESCRIPCION

NOMBRE:ejie.post-commit.svnci
DESCRIPCION: Este hook notifica a jenkins que hay cambios en el repositorio, para que se realice el Analisis estatico, por ejemplo.

NOMBRE:ejie.pre-commit.comentarios
DESCRIPCION: Este script verifica que se ha incluido un comentario de al menos 5 palabras
---

NOMBRE:ejie.pre-commit.nolibs
DESCRIPCION: Este hook verifica que cada envio tenga comentarios(5 palabras al menos) y no permite librerias (.jar .zip) en rutas del tipo WEB-INF/lib o APP-INF/lib
---
