﻿== CMD\_DESCRIPTION\_ACL ==

Establece permisos para un usuario, en un objeto dado.

== CMD\_USAGE\_ACL ==

Sintaxis:

cm acl --user=usr\_name | --group=group\_name <action=<+|-permission>+>+ spec\_objeto

--user Nombre de usuario.

--group Nombre de grupo.

action Acción a aplicar: permitir, denegar o sobrescribir.

-allowed para habilitar permisos

-denied para denegar

-overrideallowed

-overridedenied

permission Permiso a aplicar.

Usa comas para separar permisos.

(Usa 'cm showpermissions' para mostrar todos los

permisos disponibles.)

spec\_objeto Especificación del objeto sobre el que se van a fijar

los permisos.

Objetos válidos para este comando son:

repserver, repositorio, rama, changeset, etiqueta, item

y atributo.

(Usa 'cm help objectspec' para obtener más información

sobre especificaciones.)

La especificación puede ser también una ruta segura.

Consulta las Notas para obtener más información sobre

las rutas seguras.

Uso especial para rutas seguras:

cm acl [regular options] [--delete] [--branches=comma separated list] spec

(Opcionalmente, cada rama especificada en --branches se puede preceder con

los signos + o - para especificar si una rama se añadirá o se eliminará de

la lista cuando se edite.)

== CMD\_HELP\_ACL ==

La configuración de permisos requiere comprender cómo se gestiona la seguridad

en Plastic SCM. Lee la Guía de Seguridad para aprender cómo funcionan estos

permisos de seguridad:

https://www.plasticscm.com/documentation/security/plastic-scm-version-control-security-guide.shtml

Notas:

Este comando fija permisos de usuario o grupo a los objetos, repositorios,

ramas, etiquetas y/o rutas de servidor especificados.

Object specs:

Usa 'cm help objectspec' para aprender cómo especificar objetos.

El comando acl usa un tipo especial de especificación: rutas seguras.

- Especificación de rutas seguras (secured paths):

path:server\_path[#tag]

Ejemplos: path:/src/foo.c

path:/doc/pdf

path:/doc/pdf#documents

Acciones de permisos:

Usa -allowed y -denied para especificar qué permisos aplicar.

Usa los argumentos -overrideallowed y -overridedenied para especificar

qué permisos se sobrescriben.

Cada acción requiere una lista de permisos separados por comas.

Nombres de permisos:

Cada nombre de permiso debe ir precedido del símbolo + o -.

El símbolo + aplica el permiso y el símbolo - lo elimina.

Para ver los permisos de un objeto, usa el comando 'cm showacl'.

Permisos sobrescritos:

Sobrescribir un permiso usando --overrideallowed y --overridedenied te

permite evitar la herencia de permisos.

Esto es útil para evitar los permisos aplicados a nivel de repositorio o

de servidor.

Por ejemplo:

cm acl --user=vio -allowed=+ci -overrideallowed=+ci br:qa@test

(Permite a la usuaria 'vio' hacer checkin en la rama 'qa' del

repositorio 'test' incluso si ella tiene los permisos denegados a

nivel de repositorio.)

Permisos de rutas de servidor (rutas seguras):

- Es posible especificar permisos para una ruta de servidor dada.

- Estos permisos se comprueban durante la operación de check-in.

- Estos permisos también pueden ser comprobados durante la operación de

update y usados para prevenir ciertos directorios y ficheros sean

descargados al espacio de trabajo.

- Por cada ítem que se proteja (checkin), el servidor intenta concordar

la ruta del ítem con una ruta segura. Si se encuentra, entonces la

operación de checkin comprueba si el ítem tiene permisos para ser

protegido (checked in).

Los permisos que se pueden definir para una ruta segura son:

ci, change, add, move, rm, read

Si la comprobación de permisos no es satisfactoria para ninguno de los

ítems involucrados, entonces la operación de checkin se cancelará.

Para aplicar permisos de rutas seguras a un grupo de ramas, usa la

opción --branches.

Por ejemplo:

cm acl --user=jo -denied=+ci path:/src#rule0 --branches=main,main/rel0

Para editar el ACL asociado a una ruta segura, usa el símbolo "#".

Por ejemplo:

cm acl --user=jo -denied=+rm path:/src#rule0

(Sin "#", la lista de ramas tendría que ser especificada de nuevo).

Es posible también editar la lista de ramas de una ruta segura.

Por ejemplo:

cm acl path:/src#rule0 --branches=-main,+main/rel1

(Elimina main de la lista y añade main/rel1.)

Para eliminar una ruta segura, usa el argumento --delete.

Por ejemplo:

cm acl --user=jo --delete path:/src#rule0

Herencia:

Herencia es una opción que proviene de los inicios de Plastic SCM 3.0.

Es una opción avanzada, pero también está obsoleta.

Permite a un objeto heredar los permisos de otro objeto, sobrescribiendo

las relaciones de herencia por defecto.

Usa la opción -cut para cortar la cadena de herencia.

Usa la opción -cutncpy para cortar y copiar los permisos heredados.

(Estas características vienen inspiradas por los permisos de sistema de

Windows donde puedes cortar herencia pero mantener los permisos

actuales.)

La opción -inherit permiter heredar de una especificación de objeto.

Por ejemplo: '-inherit=object\_spec'

Ejemplos:

cm acl --user=danipen -denied=+ci rep:core

(Deniega el permiso de checkin al usuario danipen en el repositorio 'core'.)

cm acl --group=developers -allowed=+view,-read -denied=+chgperm br:main

(Otorga el permiso 'view', elimina el permiso 'read' y deniega el permiso

'chgperm' al grupo 'developers' en la rama 'main'.)

Ejemplos con rutas de servidor:

cm acl --group=devs -denied=+ci path:/server#rel --branches=main,main/2.0

(Deniega el permiso de checkin al grupo 'devs' para cualquier ruta que

concuerde con '/server' en las ramas 'main' y 'main/2.0'. La etiqueta #rel

se crea para poder referirse a ella más adelante.)

cm acl path:/server#rel --branches=-/main,+/main/Rel2.1

(Actualiza la ruta segura '/server', cuya etiqueta es 'rel', borrando la

rama 'main' y añadiendo la rama 'main/Rel2.1' al grupo de ramas al que la

ruta segura aplica. Teniendo en cuenta el ejemplo anterior, ahora la lista

de ramas contendrá main/Rel2.1 y main/2.0)

cm acl --user=vsanchezm -allowed=-read -overrideallowed=+read path:/doc

(Eliminar el permiso de lectura a vsanchezm, sobrescribiéndolo, en la ruta

/doc.)

== CMD\_DESCRIPTION\_ACTIVATEUSER ==

Activa un usuario de licencia desactivado previamente.

== CMD\_USAGE\_ACTIVATEUSER ==

Sintaxis:

cm activateuser nombre1 nombre2 ... [opciones]

nombre: Nombre del usuario que se desea activar.

Opciones:

--server: activa el usuario en el servidor especificado.

Usa 'cm help objectspec' para saber más sobre especificaciones.

== CMD\_HELP\_ACTIVATEUSER ==

Notas:

Este comando activa un usuario para que pueda utilizar Plastic SCM.

Para activar un usuario, éste ha de estar desactivado previamente.

Para activar un usuario por primera vez, basta con utilizar Plastic SCM

con las credenciales del usuario a activar, siempre que no se sobrepase

el número máximo de usuarios autorizados.

Ejemplos:

cm activate user juan

== CMD\_DESCRIPTION\_ADD ==

Añade un ítem al repositorio.

== CMD\_USAGE\_ADD ==

Sintaxis:

cm add [options]\* <ruta>+

ruta Rutas de los ítems a añadir, separados por espacios.

Usa comillas (") para especificar rutas que contengan

espacios.

Usa \* para añadir todo el contenido del directorio

actual.

Opciones:

-R|-r|--recursive Añade los ítems recursivamente.

--silent No muestra ninguna salida.

--ignorefailed Si un ítem no se puede añadir, la operación de add

continuará sin él. Importante: si un directorio no se

puede añadir, su contenido tampoco se añadirá.

--skipcontentcheck Cuando la extensión no es suficiente para determinar si

el fichero es de texto o binario, se asumirá que es

binario en lugar de usar el contenido para detectar el

tipo. Este funcionamiento se asume para aumentar el

rendimiento en grandes operaciones de checkin.

== CMD\_HELP\_ADD ==

Notas:

Requisitos para añadir items:

- El directorio padre del ítem a añadir debe existir en el repositorio.

Ejemplos:

cm add file1.txt file2.txt

(Añade los ítems file1.txt y file2.txt.)

cm add c:\workspace\file.txt

(Añade el ítem file.txt en la ruta c:\workspace.)

cm add -R c:\workspace\src

(Añade recursivamente el contenido de la ruta src.)

cm add -R \*

(Añade recursivamente todo el contenido del directorio actual.)

== CMD\_USAGE\_ADDIGNOREPATTERN ==

Uso:

cm addignorepattern pattern1 pattern2 ... patternN [--workspace=wkpath | --allworkspaces][--remove]

== CMD\_DESCRIPTION\_ADMIN ==

Ejecuta comandos de administración en el servidor.

== CMD\_USAGE\_ADMIN ==

Sintaxis:

cm admin comando [opciones]

Comandos:

readonly

Para obtener más información de cada uno de los comandos ejecute:

cm changeset command --usage

cm changeset command --help

== CMD\_HELP\_ADMIN ==

Notas:

Sólo el administrador del servidor puede ejecutar el comando admin.

Ejemplos:

cm admin readonly enter

cm admin readonly status

== CMD\_DESCRIPTION\_ADMIN\_READONLY ==

Permite habilitar/deshabilitar el modo solo-lectura en el servidor.

== CMD\_USAGE\_ADMIN\_READONLY ==

Sintaxis:

cm admin readonly acción [servidor]

Acciones:

enter: habilita el modo de solo-lectura en el servidor.

leave: deshabilita el modo de solo-lectura en el servidor.

status: muestra el estado del modo de solo-lectura del servidor.

Opciones:

servidor: El comando se ejecuta en el servidor especificado

(servidor:puerto; usa 'cm help objectspec' para más información.)

Si no se especifica un servidor, el comando se ejecuta

en servidor del espacio de trabajo actual.

Si la ruta actual no está en un espacio de trabajo,

se ejecuta en el servidor por defecto configurado en

client.conf.

== CMD\_HELP\_ADMIN\_READONLY ==

Notas:

Sólo el administrador del servidor puede ejecutar el comando readonly.

Ejemplos:

cm admin readonly enter diana:8086

cm admin readonly leave

== CMD\_DESCRIPTION\_ANNOTATE ==

Lista el contenido de un fichero o directorio, indicando para cada línea el propietario y la revisión en la cual fue introducida.

== CMD\_USAGE\_ANNOTATE ==

Sintaxis:

cm annotate spec [opciones]

spec Especificación del fichero que se va a anotar.

Usa 'cm help objectspec' para obtener más información

sobre especificaciones.

Opciones:

--format=cadena: Cadena con el formato de salida.

--dateformat=cadena: Cadena con el formato en que se escribirán las fechas.

--ignore=método\_de\_comparación: Cambia la forma de calcular las diferencias.

--repository=repspec: Especificación del repositorio para calcular las

anotaciones. Por defecto, este comando utiliza el

repositorio del workspace cuya revisión está cargada.

Usa 'cm help objectspec' para más información sobre las

especificaciones de repositorio.

método\_de\_comparación:

none (por defecto)

eol (ignora diferencias en retornos de carro)

whitespaces (ignora diferencias en espacios en blanco)

eol&whitespaces (las dos anteriores)

== CMD\_HELP\_ANNOTATE ==

Notas:

Los items de tipo binario no están soportados por este commando.

Este comando admite una cadena de formato para mostrar la salida.

Los parametros de salida de este comando son los siguientes:

{owner}: El propietario de la línea.

{rev}: La especificación de la revision donde apareció la línea.

{content}: El contenido de la línea.

{date}: La fecha de la revisión donde fue introducida la línea.

{comment}: El comentario de la revisión donde apareció la línea.

{changeset}: El changeset de la revisión donde apareció la línea.

{line}: El número de línea dentro del fichero.

{branch}: La rama de la revisión donde apareció la línea.

{ismergerev}: La revisión donde apareció la línea fue creada en una

operación de merge.

Puede usar la opción --dateformat para especificar el formato en el que se

escribirán las fechas en pantalla.

El commando annotate puede recuperar los datos de un repositorio remoto

usando la opción --repository. Esto es útil cuando se trabaja con

repositorios replicados.

Ejemplos:

cm ann c:\workspace\src

cm ann c:\workspace\file.txt

cm ann c:\workspace\file.txt --format"{owner} {date} {content}"

cm ann c:\workspace\file.txt --format"{owner, -7} {content} [{date}]" \

--dateformat=yyyyMMdd

cm ann c:\workspace\file.txt --repository=centralRep@myserver:8084

cm blame serverpath:/src/client/checkin/Checkin.cs#cs:73666

(Anota el fichero empezando en el changeset 73666 y usando un server path.)

== CMD\_DESCRIPTION\_APPLY\_LOCAL ==

Aplica los cambios locales (movidos, borrados y modificados localmente).

== CMD\_USAGE\_APPLY\_LOCAL ==

Sintaxis:

cm applylocal [opciones] [rutas]

rutas: rutas de los cambios locales a aplicar. Por defecto son los de todo

el workspace.

Opciones:

--dependencies: Incluir las dependencias de los cambios locales entre los

ítems a proteger.

== CMD\_HELP\_APPLY\_LOCAL ==

Notas:

- Si no se especifican [opciones] ni [rutas], la operación involucrará a

todos los cambios locales en el espacio de trabajo.

- La operación se aplica siempre recursivamente desde la ruta indicada.

Ejemplos:

cm applylocal fichero1.txt fichero2.txt

cm applylocal . (Aplica los cambios locales para el directorio actual y

todo su contenido)

cm applylocal (Aplica los cambios locales para todo el workspace)

== CMD\_DESCRIPTION\_ARCHIVE ==

Almacena o restaura datos en almacenamiento externo de un conjunto dado de revisiones especificadas.

== CMD\_USAGE\_ARCHIVE ==

Sintaxis:

cm archive [revspec]+ [-c=comentario] [-f=path y nombre de fichero base]

cm archive [revspec]+ --restore

[revspec]+ una o más especificaciones de revisión. Se pueden leer desde la

entrada estándar (STDIN) con el modificador de comando "-". Usa

'cm help objectspec' para más información sobre especificaciones.

[-c | --comment] especifica el comentario que se agregará a los datos

archivados que se generen.

[-f | --file] especifica el nombre y (opcionalmente) el path de los ficheros

que se crearán para almacenar los datos por archivar.

[--restore] restaura los datos previamente archivados utilizando los

ficheros de almacenamiento generados.

== CMD\_HELP\_ARCHIVE ==

Notas:

El comando permite extraer datos desde la base de datos de un repositorio

y volcarlos en almacenamiento externo, ahorrando espacio en la base de

datos.

Usa 'cm help objectspec' para saber más sobre cómo especificar una revisión.

El comando también puede usarse para restaurar las revisiones 'archivadas',

volviéndolas a introducir en la base de datos (opción --restore).

El usuario que ejecute este comando debe ser administrador del servidor

de Plastic SCM (propietario del servidor de repositorios) para que se le

permita completar la operación.

Cada segmento de datos de las revisiones especificadas se almacenará en

un fichero diferente, prefijado por el contenido del argumento --file.

Este argumento puede contener tanto un 'path' completo con un prefijo de

nombre de archivo como únicamente dicho prefijo.

Una vez archivados los datos de las revisiones, se podrá acceder a ellos

de dos maneras:

1) Desde el cliente: el cliente de Plastic SCM detectará que los datos

fueron archivados y pedirá al usuario que indique la ubicación en la que

se encuentran. El usuario puede configurar las ubicaciones de los datos

externos creando un fichero 'externaldata.conf' (en las ubicaciones

estándar para ficheros de configuración, siguiendo las mismas reglas que se

aplican para el fichero client.conf) que contendrá los paths en los que

se encuentren los datos archivados.

2) Desde el servidor: de esta forma los usuarios estarán totalmente

aislados del hecho de que los datos han sido archivados, ya que las

peticiones se resolverán de forma transparente por el servidor. Para

ello, el administrador creará un fichero 'externaldata.conf' en el

directorio del servidor con los paths de los datos externos.

Para restaurar (unarchive) los datos de una revisión (o conjunto de

revisiones) los datos archivados deberán estar accesibles desde el cliente.

Por tanto, no es posible restaurar datos que estén siendo resueltos por el

servidor -método 2)- porque el cliente no será capaz de identificarlos como

archivados, al resolverse de forma transparente por el servidor. Si se están

resolviendo los datos externos en servidor, para poder restaurar será

necesario que el administrador elimine los volúmenes por restaurar de las

ubicaciones accesibles por el servidor o que configure 'externaldata.conf'

apropiadamente.

Si existe una variable de entorno llamada PLASTICEDITOR apuntando al

ejecutable de un editor de texto, y no se especifica un comentario al

archivar, el editor se abrirá automáticamente para que especifique

un comentario.

Especificaciones de revisión:

rev:path[#(brSpec|csetSpec|lbSpec)], ej: rev:fichero.cs#cs:5

rev:fichero.cs#br:/main/task1

Ejemplos:

Archivar bigfile.zip:

$ cm archive bigfile.zip#br:/main#13

Archivar todos los ficheros mayores de 25Mb

$ cm find "revs where size > 26214400" --format="{item}#{branch}" --nototal | cm archive --comment="volume00" -f="volume00" -

Restaurar todos los ficheros archivados mayores de 25 Mb

$ cm find "revs where size > 26214400 and archived='true'" --format="{item}#{branch}" --nototal | cm archive --restore

== CMD\_DESCRIPTION\_ATTRIBUTE ==

Use este comando para administrar atributos.

== CMD\_USAGE\_ATTRIBUTE ==

Sintaxis:

cm attribute | att command [options]

Comandos:

create | mk

delete | rm

set

unset

rename

edit

Para obtener más información sobre cada comando:

cm attribute comando --usage

cm attribute comando --help

== CMD\_HELP\_ATTRIBUTE ==

Ejemplos:

cm attribute create estado

cm attribute set att:estado br:/main/SCM105 abierto

cm attribute unset att:estado br:/main/SCM105

cm attribute delete att:estado

cm attribute rename att:estado estadoBuild

cm attribute edit att:estado "Estado de la tarea en CI"

== CMD\_DESCRIPTION\_CHANGELIST ==

Permite organizar los cambios pendientes del espacio de trabajo en diferentes changelists.

== CMD\_USAGE\_CHANGELIST ==

Sintaxis:

a) Manejo de los objetos 'changelist':

cm changelist [list]

Muestra los changelists en el espacio de trabajo actual.

cm changelist add changelistName [descpription\_text] [--persistent | --notpersistent]

Crea un nuevo changelist con nombre = "changelistName":

"descpription\_text": descripción para el nuevo changelist.

"--persistent | --notpersistent": Especifica si el changelist ha de eliminarse o no si el

contenido del changelist se queda vacío (los cambios que contiene se protegen o se

deshacen).

(valor por defecto: "--notpersistent").

cm changelist rm changelistName

Borra un changelist con nombre = "changelistName".

cm changelist edit changelistName [editParameterAction newValue] [--persistent | --notpersistent]

Modifica un changelist con nombre = "changelistName:

"editParameterAction": Posibles valores: "rename" | "description"

"--persistent | --notpersistent": Especifica si el changelist ha de eliminarse o no si el

contenido del changelist se queda vacío (los cambios que contiene se deshacen o se les

ejecuta una operación de check-in.

(valor por defecto: "--notpersistent").

b) Manejo del contenido de un 'changelist' concreto:

cm changelist changelistName (add | rm) ruta(s)

Añade / elimina del changelist con nombre = "changelistName" el cambio que

coincide con la(s) ruta(s) especificada(s).

Opciones:

clist\_name El nombre un changelist.

clist\_desc La descripción un changelist.

action\_name Utiliza '^rename' o '^description' para modificar un

changelist.

action\_value Realiza la operación de renombrado o cambio de la descripción

cuando editas un changelist.

--persistent El changelist permacenerá en el workspace incluso si se ha ejecutado

una operación de check-in o de revert.

--notpersistent (Default) El changelist no permanecerá en el workspace.

--symlink Realizar la operacion de checkin sobre el fichero de link y no

al que apunta..

== CMD\_HELP\_CHANGELIST ==

Notas:

El comando 'changelist' permite administrar los changelists existentes en un

espacio de trabajo, así como los cambios que contiene cada uno de ellos.

Ejemplos del manejo de los changelists existentes en el espacio de trabajo:

$ cm changelist

(lista los changelists en el espacio de trabajo actual).

$ cm changelist add config\_changes "dotConf files" --persistent

(crea un changelist con nombre "config\_changes", descripción "dotConf files",

que además quedará persistente en el espacio de trabajo una vez que se

haga check-in o revert de dicho changelist).

$ cm changelist edit config\_changes rename "config\_files" --notpersistent

(edita el changelist con nombre "config\_changes" y lo renombra a

"config\_files". Además convierte el changelist en no persistente).

$ cm changelist edit config\_changes --notpersistent

(edita el changelist con nombre "config\_changes" y lo convierte en no persistente).

$ cm changelist rm "config\_files"

(elimina el changelist "congig\_files" del espacio de trabajo actual).

Ejemplos del manejo del contenido de un changelist:

(Nota, los ficheros que se manejan en los siguientes comandos han de

encontrarse en estado "Added" or "Checked-out".

$ cm changelist "config\_files" add "foo.conf"

(añade el fichero "foo.conf" al changelist "config\_files")

$ cm changelist "config\_files" rm "foo.conf"

(elimina el fichero "foo.conf" del changelist "config\_files" y

lo mueve al changelist por defecto del sistema).

== CMD\_DESCRIPTION\_CHANGESET ==

Ejecuta operaciones avanzadas sobre changesets.

== CMD\_USAGE\_CHANGESET ==

Sintaxis:

cm changeset comando [opciones]

Comandos:

move | mv

delete | rm

editcomment | edit

Para obtener más información de cada uno de los comandos ejecute:

cm changeset command --usage

cm changeset command --help

== CMD\_HELP\_CHANGESET ==

Ejemplos:

cm changeset move cs:15@myrepo br:/main/scm005@myrepo

cm changeset delete cs:2b55f8aa-0b29-410f-b99c-60e573a309ca@devData

== CMD\_DESCRIPTION\_CHANGESET\_EDIT\_COMMENT ==

Modifica el comentario de un changeset.

== CMD\_USAGE\_CHANGESET\_EDIT\_COMMENT ==

Sintaxis:

cm changeset editcomment | edit <cset\_spec> <nuevo\_comentario>

Opciones:

spec\_cset El changeset objetivo para editar su comentario.

Usa 'cm help objectspec' para más información.

nuevo\_comentario El nuevo comentario que será asignado al changeset

seleccionado.

== CMD\_HELP\_CHANGESET\_EDIT\_COMMENT ==

Notas:

- La especificación del changeset objetivo debe ser válida.

Ejemplos:

cm changeset editcomment cs:15@myrepo "Olvidé añadir detalles importantes."

cm changeset edit cs:cb11ecdb-1aa9-4f11-8698-dcab14e5885a \

"Este texto de comentario sustituirá al anterior."

== CMD\_DESCRIPTION\_CHANGESET\_MOVE ==

Mueve un changeset y todos sus descendientes en la misma rama a otra rama distinta.

== CMD\_USAGE\_CHANGESET\_MOVE ==

Sintaxis:

cm changeset move | mv <spec\_cset> <spec\_branch>

Opciones:

spec\_cset El primer changeset que debe moverse a una rama

diferente. Todos los changeset descendientes en la

misma rama se moverán también.

spec\_branch La rama destino a la que mover los changesets

seleccionados. Debe estar vacía o no existir. En caso

de no existir, el comando la creará.

Usa 'cm help objectspec' para obtener más información sobre cómo indicar

especificaciones de changeset y rama.

== CMD\_HELP\_CHANGESET\_MOVE ==

Notas:

- La especificación del changeset objetivo debe ser válida.

- La rama de destino debe estar vacía o no existir.

- Si la rama de destino no existe, el comando la creará.

- Los enlaces de merge no se verán afectados dado que las ramas no están

involucradas en ellos.

Ejemplos:

cm changeset move cs:15@myrepo br:/main/scm005@myrepo

cm changeset move cs:cb11ecdb-1aa9-4f11-8698-dcab14e5885a \

br:/hotfix/TL-352

== CMD\_DESCRIPTION\_CHANGESET\_DELETE ==

Elimina un changeset del repositorio.

== CMD\_USAGE\_CHANGESET\_DELETE ==

Sintaxis:

cm changeset delete | rm <spec\_cset>

Opciones:

spec\_cset El changeset seleccionado para eliminar. Debe cumplir

una serie de condiciones; ver ayuda completa para

más información. Usa 'cm help objectspec' para aprender

cómo construir las especificaciones.

== CMD\_HELP\_CHANGESET\_DELETE ==

Notas:

- El changeset objetivo debe ser el último en su rama.

- No debe haber ningún otro changeset que tenga al objetivo como padre.

- El changeset objetivo no debe ser origen de merge ni estar envuelto en

el intervalo origen de un merge de intervalo.

- No debe haber etiquetas aplicadas al changeset objetivo.

- El changeset objetivo no puede ser el changeset raíz (cs:0)

Ejemplos:

cm changeset rm cs:4525@myrepo@myserver

cm changeset delete cs:cb11ecdb-1aa9-4f11-8698-dcab14e5885a

== CMD\_DESCRIPTION\_CHANGEUSERPASSWORD ==

Cambia la contraseña del usuario actual cuando la configuración de seguridad es usuario/contraseña.

== CMD\_USAGE\_CHANGEUSERPASSWORD ==

Sintaxis:

cm changeuserpassword

== CMD\_HELP\_CHANGEUSERPASSWORD ==

Notas:

Este comando solamente está disponible cuando el sistema de seguridad y

usuarios está configurado en modo de funcionamiento de usuario/contraseña.

Ejemplo:

cm changeuserpassword

== CMD\_DESCRIPTION\_CHECKCONNECTION ==

Comprueba la conexión del cliente de Plastic SCM contra el servidor.

== CMD\_USAGE\_CHECKCONNECTION ==

Sintaxis:

cm checkconnection

== CMD\_HELP\_CHECKCONNECTION ==

Notas:

Este comando retorna un mensaje indicando si hay una conexión valida contra

el servidor de Plastic SCM o no.

El comando comprueba si el usuario configurado es válido.

También comprueba si la versión del cliente es compatible con la del

servidor.

== CMD\_DESCRIPTION\_CHECKDB ==

Comprueba la integridad de las bases de datos.

== CMD\_USAGE\_CHECKDB ==

Sintaxis:

cm checkdatabase [spec]

spec: Especificación del servidor o de un repositorio.

Usa 'cm help objectspec' para aprender cómo indicar especificaciones

de servidor y repositorio.

== CMD\_HELP\_CHECKDB ==

Notas:

- Si no se especifica [spec], el comando utilizará el valor del servidor

del fichero client.conf.

Ejemplos:

cm checkdatabase repserver:localhost:8084

cm checkdatabase rep:default@localhost:8084

== CMD\_DESCRIPTION\_CHECKIN ==

Crea una nueva revisión de un ítem.

== CMD\_USAGE\_CHECKIN ==

Se pueden proteger tanto ficheros como directorios.

Sintaxis:

cm checkin [opciones] [rutas]

rutas: rutas de los items a proteger.

Opciones:

--nchk: (No check) No chequea el contenido del fichero al proteger.

Ver notas para mas información.

--symlink: Realizar la operacion de checkin sobre el fichero de link y no

al que apunta.

-c: Añade un comentario al changeset creado en la operación de protección.

--commentsfile=comment.txt: Carga el comentario del fichero especificado.

--all (-a): Los ítems cambiados, movidos y borrados localmente, en las

rutas dadas, también serán protegidos.

--dependencies: Incluir las dependencias de los cambios locales entre los

ítems a proteger.

--private: Los ítems privados también serán protegidos.

== CMD\_HELP\_CHECKIN ==

Notas:

- Si no se especifican [opciones] ni [rutas], la operación involucrará a

todos los cambios pendientes en el espacio de trabajo.

- La operación de protección se aplica siempre recursivamente desde la

ruta indicada.

- Requisitos para proteger un ítem:

\* El ítem debe estar controlado por Plastic SCM.

\* El ítem ha de estar desprotegido.

\* La revisión ha de tener cambios. Es decir, no se pueden proteger los

mismos datos. Para eliminar la desprotección descartando los cambios,

se utiliza el comando uncheckout.

\* Con la opción --nchk, se fuerza la protección del ítem, sin comprobar

los datos.

Ejemplos:

cm checkin fichero1.txt fichero2.txt

cm checkin . (Protege el directorio actual)

(Proteger recursivamente el contenido de 'src')

cm checkin link --symlink

(Proteger el fichero de link y no al que apunta, opción disponible en UNIX)

Comentarios:

Para especificar comentarios use el switch "-c" o "-m" del siguiente modo:

cm ci file1.txt -c="my comment"

cm ci file1.txt -m "my comment"

Proteger 'changelist'. Ejemplo:

cm status --short --compact --changelist=pending\_to\_review | cm checkin -

(El comando listará los paths en el changelist llamado 'pending\_to\_review'

y dicha lista será redirigida a la entrada del comando 'checkin').

Si existe una variable de entorno llamada PLASTICEDITOR apuntando al

ejecutable de un editor de texto, y no se especifica un comentario al

hacer checkin, el editor se abrirá automáticamente para que especifique

un comentario.

== CMD\_DESCRIPTION\_CHECKOUT ==

Desprotege un ítem dado.

== CMD\_USAGE\_CHECKOUT ==

Sintaxis:

cm co [Opciones] paths

paths: Rutas de los elementos que se desean desproteger.

Opciones:

-R: Descender recursivamente dentro del ítem.

--symlink: Efectúar la desprotección sobre el fichero de link y no al que

apunta.

--ignorefailed: Si un ítem no se puede bloquear (no se le puede hacer un

checkout exclusivo), la operación de checkout continuará sin él.

== CMD\_HELP\_CHECKOUT ==

Notas:

Cuando un ítem del workspace se desea modificar, se debe desproteger.

La forma de desproteger un ítem es utilizando el comando co.

Se pueden desproteger tanto ficheros como directorios.

Requisitos: Para poder desproteger elementos:

El ítem debe estar controlado por Plastic.

El ítem debe estar protegido previamente.

Ejemplos:

cm co . (Desproteger el directorio actual)

cm co c:\workspace\fichero.txt

cm co fichero1.cs fichero2.cs

cm co –R c:\workspace\src (desprotege recursivamente todo el contenido de ‘src’)

cm co –R \* (desprotege recursivamente todos los elementos del directorio actual)

cm co link --symlink

(desprotege el fichero de link y no el fichero al que apunta, disponible en UNIX)

== CMD\_DESCRIPTION\_CHECKSELECTORSYNTAX ==

Comprueba la sintaxis de un selector.

== CMD\_USAGE\_CHECKSELECTORSYNTAX ==

Uso:

cm checkselectorsyntax [--file=file]

Opciones:

--file=file: El fichero del cual leer el selector; si no se especifica se

supone la entrada estándar.

== CMD\_HELP\_CHECKSELECTORSYNTAX ==

Notas:

Este comando lee el selector desde un fichero o desde la entrada estándar

y comprueba si su sintaxis es válida. Si falla la comprobación se muestra

en la salida estándar.

== CMD\_DESCRIPTION\_CHGREVTYPE ==

Cambia el tipo de todas las revisiones de un ítem.

== CMD\_USAGE\_CHGREVTYPE ==

Sintaxis:

cm changerevisiontype | chgrevtype | crt rutas --type=tipo

rutas: Rutas de los items para alterar el tipo de sus revisiones.

tipo: Tipo al que se fijarán las revisiones.

== CMD\_HELP\_CHGREVTYPE ==

Notas:

Este comando sólo puede aplicarse a ficheros, no a directorios.

El tipo especificado, ha de ser alguno soportado por el sistema:

bin

txt

...

Ejemplos:

cm changerevisiontype c:\workspace\file.txt --type=txt

(Cambia el tipo de revisión de 'file.txt' a 'txt'.)

cm chgrevtype comp.zip "image file.jpg" --type=bin

(Cambia el tipo de revisión de 'comp.zip' e "image file.jpg" a 'bin'.)

cm crt \*.\* --type=txt

(Cambia el tipo de revisión de todos los archivos a 'txt'.)

== CMD\_DESCRIPTION\_TRIGGER\_EDIT ==

Cambia la información asociada a un trigger existente.

== CMD\_USAGE\_TRIGGER\_EDIT ==

Sintaxis:

cm trigger edit subtipo-tipo posición [opciones]

subtipo-tipo: Tipo y subtipo de trigger.

Teclee 'cm showtriggertypes' para ver una lista completa de los tipos de

trigger disponibles. Algunos ejemplos son:

before-mkbranch

after-mkwokspace

after-setselector

posición: posición que ocupa el trigger que deseamos modificar.

Opciones:

--position=posición: nueva posición para el trigger especificado.

--name=nombre: nuevo nombre para el trigger especificado.

--script=ruta: nueva ruta de ejecución para el script del trigger

especificado. Si esta ruta comienza por "webtrigger ", se

considerará como web trigger. Consulte la ayuda del comando para obtener

detalles adicionales.

--filter=filtro: Comprueba sólo aquellos elementos que coincidan con el

filtro de busqueda indicado.

--server=servidor: modifica el trigger en el servidor especificado.

Si no se especifica ninguno lo busca en el que se configuró en el cliente.

Usa 'cm help objectspec' para más información.

== CMD\_HELP\_TRIGGER\_EDIT ==

Notas:

Web triggers: Para crear web triggers, escriba "webtrigger <target-uri>"

como comando de trigger. En este caso, el trigger ejecutará una consulta

POST contra el URI especificado, en la que el cuerpo de la petición

contiene un diccionario JSON con las variables de entorno del trigger.

También estará presente una clave INPUT apuntando a un vector de cadenas de

texto para aquellos triggers que escriban en la entrada estándar de sus

comandos asociados.

Ejemplos:

cm trigger edit after-setselector 6 --name="Gestiona backup2" --script="/nuevo/path/al/script"

cm tr edit before-mklabel 7 --position=4 --server=myserver:8084

cm tr edit after-add 2 --script="webtrigger http://myserver.org/api"

== CMD\_DESCRIPTION\_CODEREVIEW ==

Crea, edita o elimina revisiones de código.

== CMD\_USAGE\_CODEREVIEW ==

Sintaxis:

cm codereview <spec> <título> [--status=nombre\_estado]

[--assignee=nombre\_usuario] [--format=cad\_formato]

[--repository=rep\_spec]

spec Puede tratarse tanto de una especificación de

changeset como de una especificación de rama.

Determina el objetivo de la nueva revisión de código.

título Una cadena de texto para utilizarla como título de la

nueva revisión de código.

cm codereview -e <id> [--status=nombre\_estado]

[--assignee=nombre\_usuario] [--repository=rep\_spec]

cm codereview -d <id> [id...] [--repository=rep\_spec]

id Número identificador de la revisión de código. Puede

utilizarse también su GUID.

Opciones:

-e Edita los parámetros de una revisión de código

existente.

-d Elimina una revisión de código existente.

--status Modifica el estado de una revisión de código. Consulte

las notas para más información.

--assignee Modifica el usuario asignado a la revisión de código.

--repository Indica el repositorio que se utilizará por defecto.

--format Devuelve el mensaje de respuesta con el formato

especificado. Consulte la ayuda para más información.

== CMD\_HELP\_CODEREVIEW ==

Notas:

Este comando permite manipular revisiones de código: crear, editar

y eliminar revisiones de código de changesets o ramas.

Para crear una nueva revisión de código se requiere una especificación de

changeset o rama, junto con un título. Además, puede indicarse tanto

el usuario asignado (revisor de los cambios) como el estado inicial.

Como resultado se devolverá un ID (o GUID, si así se solicitó).

Para editar o eliminar una revisión de código existente se requiere el ID

(o GUID) de la misma. No se mostrarán mensajes si no se detectan errores.

El parámetro 'status' sólo puede tener como valor uno de los siguientes:

'Under review' (valor por defecto), 'Reviewed' o 'Rework required'.

El parámetro 'repository' se encuentra disponible para actuar sobre las

revisiones de código de un repositorio diferente al asociado al espacio

de trabajo actual (o para situaciones en las que no exista espacio de

trabajo).

Personalización del formato de salida:

Este comando acepta una cadena de formato para mostrar su salida.

Los parámetros de salida del comando son los siguientes:

{0} id

{1} guid

Por favor, tenga en cuenta que el parámetro --format sólo tiene efecto

al crear una nueva revisión de código.

Ejemplos:

cm codereview cs:1856@myrepo@myserver:8084 "Mi nueva revisión" --assignee=dummy

cm codereview br:/main/task001@myrepo@myserver:8084 "Mi nueva revisión" --status="Rework required" --assignee=newbie --format="{id} -> {guid}"

cm codereview 1367 -e --assignee=new\_assignee

cm codereview -e 27658884-5dcc-49b7-b0ef-a5760ae740a3 --status=Reviewed

cm codereview -d 1367 --repository=myremoterepo@myremoteserver:18084

cm codereview 27658884-5dcc-49b7-b0ef-a5760ae740a3 -d

== CMD\_DESCRIPTION\_CRYPT ==

Encripta una contraseña.

== CMD\_USAGE\_CRYPT ==

Sintaxis:

cm crypt mypassword

mypassword Contraseña que se va a encriptar.

== CMD\_HELP\_CRYPT ==

Notas:

Este comando encripta una contraseña cualquiera pasada como argumento.

Está diseñado para encriptar las contraseñas de los ficheros de

configuración y aumentar así su seguridad.

Ejemplos:

cm crypt dbconfpassword -> ENCRYPTED: encrypteddbconfpassword

(Encripta la contraseña en el fichero de la base de datos de configuración:

db.conf)

== CMD\_DESCRIPTION\_DEACTIVATEUSER ==

Desactiva un usuario activado previamente.

== CMD\_USAGE\_DEACTIVATEUSER ==

Sintaxis:

cm deactivateuser nombre1 nombre2 ... [opciones]

nombre: Nombre del usuario que se desea desactivar.

Opciones:

--nosolveuser: Intenta desactivar el usuario especificado sin resolver

el nombre dado en el sistema de autenticación subyacente.

--server: desactiva el usuario en el servidor especificado.

== CMD\_HELP\_DEACTIVATEUSER ==

Notas:

Este comando deshabilita a un usuario para el uso de Plastic SCM.

Por favor, consulte la guía de usuario, o el comando "activateuser" para

obtener más información acerca de la habilitación de usuarios de Plastic SCM.

Para activar un usuario por primera vez, basta con utilizar Plastic SCM con

las credenciales del usuario a activar, siempre que no se sobrepase el

número máximo de usuarios autorizados.

El comando comprueba que el usuario a deshabilitar existe en el sistema de

autenticación subyacente (p.ej: ActiveDirectory, LDAP, Usuario/Contraseña ...).

Si se desea desactivar un usuario que ya no existe en el sistema de autenticación,

utilice la opción --nosolveuser

Ejemplos:

cm deactivateuser juan

cm deactivateuser juan --server=miservidor:8084

cm deactivateuser S-1-5-21-3631250224-3045023395-1892523819-1107 --nosolveuser

== CMD\_DESCRIPTION\_DIFF ==

Muestra las diferencias existentes entre dos revisiones o dos changesets

== CMD\_USAGE\_DIFF ==

Sintaxis:

cm differences [opciones] version1 [version2] [ruta]

version1: especificación de rama, primera revisión, changeset, etiqueta o

shelve a comparar.

version2: segunda especificación de revisión, changeset, etiqueta o shelve

a comparar.

ruta: muestra las diferencias en el fichero especificado

Usa 'cm help objectspec' para obtener más información sobre especificaciones.

Opciones:

--added: imprime sólo las diferencias consistentes en items añadidos

--changed: imprime sólo las diferencias consistentes en items cambiados

--moved: imprime sólo las diferencias consistentes en items movidos

o renombrados

--deleted: imprime sólo las diferencias consistentes en items borrados

--all: imprime todas las diferencias

La primera letra de cada línea que se imprime denota el tipo de cambio:

A significa que el ítem fue añadido

C signnifica que el ítem fue cambiado

D significa que el ítem fue borrado

M significa que el ítem fue movido o renombrado. El ítem a la izquierda

es el nombre original, el ítem a la derecha es el nombre en destino

--clean: limpia las diferencias de la rama (omite las diferencias que vienen

de los merges

--download: almacena el contenido de las diferencias en la ruta especificada.

--fullpaths, --fp: Fuerza al comando a imprimir rutas de workspace absolutas

siempre que sea posible.

--repositorypaths: imprime rutas de repositorio en lugar de rutas

de workspace. (Ésta opción prevalece sobre la opción "--fullpaths").

== CMD\_HELP\_DIFF ==

Notas:

Este comando muestra las diferencias en una rama si el segundo comando no se

especifica o entre dos revisiones, changesets o etiquetas si se especifica

el segundo parámetro.

Comparar ramas:

cm diff br:/main/task001

cm diff br:/main/task001 \doc\readme.txt

Comparar changesets:

cm diff 19

cm diff 19 25

cm diff cs:19 cs:25

cm diff cs:19 cs:23 --changed

cm diff cs:19 cs:23 --repositorypaths

cm diff cs:19 cs:23 \doc\readme.txt

Comparar etiquetas:

cm diff lb:EtiquetaPrimeraRelease lb:EtiquetaSegundaRelease

cm diff lb:tag\_193.2 cs:34214

cm diff cs:31492 lb:tag\_193.2

Comaprar shelves:

cm diff sh:2

cm diff sh:2 sh:4

Comparar revisiones:

cm diff rev:readme.txt#cs:19 rev:readme.txt#cs:20

cm diff serverpath:/doc/readme.txt#cs:19@myrepo \

serverpath:/doc/readme.txt#br:/main@myrepo@localhost:8084

== CMD\_DESCRIPTION\_DIFFMETRICS ==

Muestra métricas sobre las diferencias de dos revisiones (líneas añadidas, cambiadas y borradas).

== CMD\_USAGE\_DIFFMETRICS ==

(líneas cambiadas, añadidas y borradas)

Sintaxis:

cm diffmetrics [opciones] spec\_revision1 spec\_revision2

spec\_revision1: Especificación de revisión.

spec\_revision2: La otra especificación de revisión.

Usa 'cm help objectspec' para más información sobre especificaciones.

Opciones:

--format=str: Cadena de formato de salida. Ver notas para más información.

--encoding=tipo\_codificación: Cambia la codificación.

--ignore=método\_de\_comparación: Cambia la forma de calcular las diferencias.

tipo\_codificación : ascii

unicode

bigendian

utf7

utf8

método\_de\_comparación:

none (por defecto)

eol (ignora diferencias en retornos de carra)

whitespaces (ignora diferencias en espacios en blanco)

eol&whitespaces (las dos anteriores)

== CMD\_HELP\_DIFFMETRICS ==

Notas:

Este comando acepta una cadena de formato para mostrar la salida.

Cada parámetro de salida se identifica con un número, y el usuario se

puede referir a él escribiéndolo entre los símbolos '{' y '}'.

Los parámetros de salida de este comando son los siguientes:

0: Número de líneas cambiadas

1: Número de líneas añadidas

2: Número de líneas borradas

Éstas son cadenas de formato válidas:

--format={0}{1}{2}

--format="Hay {0} líneas cambiadas, {1} líneas añadidas y {2} líneas borradas"

Ejemplos:

cm diffmetrics fichero.txt#br:/main#2 fichero.txt#br:/main/scm0211#4

cm diffmetrics fichero.txt#br:/main#2 fichero.txt#br:/main#3 --encoding=utf8 --ignore=whitespaces

== CMD\_DESCRIPTION\_FASTEXPORT ==

Exporta los datos de un repositorio de Plastic SCM a un formato fast-export.

== CMD\_USAGE\_FASTEXPORT ==

Sintaxis:

cm ^fast-export | ^fe <repspec> <fast-export-file>

[--^import-marks=<marks\_file>]

[--^export-marks=<marks\_file>]

[--^branchseparator=<chr\_separator>]

[--^nodata] [--^from=<changesetid>] [--^to=<changesetid>]

Opciones:

repspec Repositorio cuyos datos serán exportados.

(Usa 'cm help objectspec' para aprender cómo especificar

un repositorio.)

fast-export-file Fichero con los datos del repositorio en formato

fast-export de Git.

--import-marks Fichero de marcas usado para la importación incremental.

Este fichero habrá sido previamente exportado con el

comando --export-marks. Los changesets descritos en este

fichero no serán importados porque ya lo fueron en una

importación previa.

--export-marks Fichero donde se guardan los changesets importados.

Este fichero podrá ser usado en una operación fast-import

para indicar los changesets que han sido importados.

--branchseparator Plastic SCM usa "/" como separador por defecto en la

jerarquía de ramas. Esta opción permite usar un carácter

como separador de jerarquía. Por ejemplo, la rama

main-task-sub será mapeada en Plastic como /main/task/sub.

--nodata Realiza la exportaciónExporta pero sin incluir datos.

Esta opción es útil si se desea comprobar que la

exportación se ejecutará sin problemas.

--from Exporta desde el changeset especificado.

--to Exporta hasta el changeset especificado.

== CMD\_HELP\_FASTEXPORT ==

Notas:

- Para importar un repositorio de Plastic a Git, usa el siguiente comando:

cat repo.fe.00 | git fast-import --export-marks=marks.git --import-marks=marks.git

- La exportación incremental es soportado usando un fichero de marcas que

contenga los changesets previamente importados (ficheros --import-marks y

--export-marks).

Esto significa que solamente los nuevos changesets que no fueron

exportados en una operación previa de fast-export serán exportados.

Ejemplos:

cm fast-export repo@localhost:8087 repo.fe.00 --import-marks=marks.cm --export-marks=marks.cm

(Exporta el repositorio "repo" en el servidor local al fichero "repo.fe.00"

en formato fast-export de Git y crea el fichero de marcas para poder

realizar una futura exportación incremental.)

cm ^fast-export repo@localhost:8087 repo.fe.00 --^from=20

(Exporta el repositorio "repo" en el servidor local al fichero "repo.fe.00"

en formato Git fast-export desde el changeset '20'.)

== CMD\_DESCRIPTION\_FASTIMPORT ==

Importa datos de fast-export de Git en el repositorio.

== CMD\_USAGE\_FASTIMPORT ==

Sintaxis:

cm ^fast-import | ^fi <repspec> <fast-export-file>

[--^import-marks=<marks\_file>]

[--^export-marks=<marks\_file>]

[--^stats] [--^branchseparator=<chr\_separator>]

[--^nodata] [--^ignoremissingchangesets] [--^mastertomain]

Opciones:

repspec Repositorio en el cual se importarán los datos.

Si no existe en el momento de la operación, se

crea automáticamente. (Usa 'cm help objectspec'

para aprender cómo especificar un repositorio.)

fast-export-file Fichero con los datos en formato Git

fast-export.

--import-marks El fichero de etiquetas (marks) utilizado para

importaciones incrementales. Dicho fichero se

ha exportado previamente con --export-marks.

Los changesets descritos en este fichero no se

importarán porque ya existían de una importación

previa.

--export-marks El fichero en el cual se guardarán los changesets

importados. Dicho fichero se utiliza en una

operación posterior de fast-import para señalar

los changesets que ya han sido importados.

--stats Muestra estadísticas del proceso de importación.

--branchseparator Plastic SCM usa "/" como separador por defecto

en la jerarquía de ramas. Esta opción permite

usar un carácter como separador en esa

jerarquía. Por ejemplo, la rama main-task-sub

será mapeada en Plastic SCM como /main/task/sub.

--nodata Realiza la importación pero sin incluir datos.

Esta opción es útil si se desea comprobar que la

importación se ejecutará sin problemas.

--ignoremissingchangesets Los changesets que no pueden ser importados se

descartan y la operación de fast-import continúa

sin ellos.

--mastertomain Realiza la importación usando "main" en lugar

de "master".

== CMD\_HELP\_FASTIMPORT ==

Notas:

\* Para exportar un repositorio de Git, utilice un comando como el

siguiente:

git fast-export --all -M --signed-tags=strip

--tag-of-filtered-object=drop> ..\git-fast-export.dat

La opción -M es importante para detectar ítems movidos.

\* El repositorio especificado se crea en caso de que no exista.

\* Se soporta una importación incremental utilizando ficheros de marcas que

contienen los changesets ya importados (--import-marks and --export-marks).

Esto significa que solamente se importarán los changesets que no fueron

importados previamente.

Ejemplos:

cm fast-import mynewrepo@atenea:8084 repo.fast-export

Importa el contenido exportado en el fichero repo.fast-export en el

repositorio "mynewrepo" del servidor atenea:8084.

cm fast-import repo@server:8084 repo.fast-export --export-marks=rep.marks

Importa el contenido del fichero de exportaciones repo.fast-export en el

repositorio "mynewrepo" del servidor atenea:8084 y crea un archivo de

marcas para importaciones incrementales posteriores.

cm fast-import repo@server:8084 repo.fast-export --importmarks=repo.marks

--export-marks=repo.marks

Importa el contenido del fichero repo.fast-export. Solamente se importarán

los changesets que no están en el fichero de marcas. Se utiliza el mismo

fichero de marcas para guardar los changesets que se importarán esta vez,

de nuevo con el fin de otra importación incremental posterior.

== CMD\_DESCRIPTION\_FILEINFO ==

Obtiene información detallada sobre elementos en el espacio de trabajo.

== CMD\_USAGE\_FILEINFO ==

Sintaxis:

cm fileinfo <item\_path> [--fields=field\_list] [--xml[=output\_file]]

[--format=str\_format] [--encoding=name]

item\_path Elementos para mostrar, separados por espacios. Pueden

utilizarse comillas (") para aquellas rutas que

contengan espacios.

Opciones:

--fields Una cadena de valores separados por coma. Indica

qué campos se mostrarán por cada elemento.

Puede encontrar información adicional mediante el

comando 'cm help fileinfo'.

--xml Vuelca la salida en formato XML a la salida estándar.

Es posible especificar un fichero de salida. Esta opción

no puede combinarse con --format.

--format Obtiene el mensaje de salida en un formato específico.

Consulte las notas para más información. No puede

combinarse con --xml.

--symlink Aplica la operación fileinfo al enlace simbólico y no

a su destino.

--encoding Especifica la codificación de salida. P. ej.: UTF-8.

Consulte la documentación MSDN para obtener la lista de

codificaciones soportadas y su formato, al final de la

página. La columna "Name":

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

== CMD\_HELP\_FILEINFO ==

Notas:

Este comando muestra una lista detallada de atributos para cada elemento

seleccionado. Cada atributo se muestra en una nueva línea por defecto.

Se puede modificar la lista de atributos para mostrar únicamente los

atributos que el usuario necesite. A tal efecto se puede utilizar el

argumento --fields=lista, que acepta una cadena de nombres de atributo

separados por comas. De este modo, solo se mostrarán aquellos argumentos

cuyo nombre esté presente en la lista.

Revision Head Changeset:

Esta opción está desactivada por defecto. Tenga en cuenta que la

recuperación de este atributo es notablemete más lenta que el resto, por

lo que recomendamos al usuario agrupar tantos ficheros por ejecución como

sea posible. Esto mejora los tiempos de ejecución, al evitar numerosas

ejecuciones 'cm fileinfo' separadas.

Adicionalmente, esta característica no se encuentra disponible actualmente

para directorios controlados por PlasticSCM.

A continuación se encuentra la lista completa de nombres de atributo. Los

nombres marcados con asterisco ('\*') no se muestran por defecto:

ClientPath Ruta local del elemento en disco.

RelativePath Ruta del elemento relativa al workspace.

ServerPath Ruta del elemento en el repositorio.

(Nota: las rutas afectadas por workspace

transformables no están soportadas actualmente)

Size Tamaño del elemento.

Hash Suma 'hash' del elemento

Owner Nombre del usuario propietario del elemento.

RevisionHeadChangeset\* Changeset de la revisión cargada en el último

changeset de la rama (ver párrafo anterior).

RevisionChangeset Changeset de la revisión cargada en el workspace

RepSpec Especificación del repositorio al que pertenece

el elemento (repositorio@servidor).

Status Estado actual del elemento en el espacio de

trabajo (añadido, eliminado, checkout, etc.).

Type Tipo de la revisión (texto, binario, directorio,

enlace simbólico o desconocido).

Changelist Lista de cambios a la que pertenece el elemento

(si la hubiere).

IsLocked\* Muestra si el elemento está bloqueado o no.

LockedBy\* Usuario que bloqueó el elemento mediante

checkout exclusivo.

LockedWhere\* Localización del elemento bloqueado mediante

checkout exclusivo.

IsUnderXlink Muestra si el elemento se encuentra bajo un

Xlink o no.

UnderXlinkTarget Repositorio destino del Xlink en que se

encuentra el elemento, si lo hubiere.

UnderXlinkPath Ruta en el repositorio apuntado por el Xlink

en que se encuentra el elemento, si lo hubiere.

UnderXlinkWritable Muestra si el Xlink bajo el que se encuentra el

elemento es de escritura o no.

UnderXlinkRelative Muestra si el Xlink bajo el que se encuentra el

elemento es relativo o no.

IsXlink Muestra si el elemento es un Xlink o no.

XlinkTarget Repositorio objetivo del Xlink, si el elemento

lo es.

XlinkName Nombre del Xlink, si el elemento lo es.

XlinkWritable Muestra si el elemento es un Xlink de escritura

o no.

XlinkRelative Muestra si el elemento es un Xlink relativo o no

Personalización del formato de salida:

Este comando acepta una cadena de formato para mostrar su salida.

Los parámetros de salida de este comando son los siguientes:

ClientPath

RelativePath

ServerPath

Size

Hash

Owner

RevisionHeadChangeset

RevisionChangeset

Status

Type

Changelist

IsLocked

LockedBy

LockedWhere

IsUnderXlink

UnderXlinkTarget

UnderXlinkPath

UnderXlinkWritable

UnderXlinkRelative

IsXlink

XlinkTarget

XlinkName

XlinkWritable

XlinkRelative

RepSpec

Tenga en cuenta que las opciones --format y --xml son mutuamente

exclusivas, por lo que no se pueden utilizar simultáneamente.

Ejemplos:

cm fileinfo file1.txt file2.txt dir/

cm fileinfo "New Project.csproj" --xml

cm fileinfo assets.art --fields=ServerPath,Size,IsLocked,LockedBy

cm fileinfo proj\_specs.docx --fields=ServerPath,RevisionChangeset --xml

cm fileinfo samples.ogg --format="{ServerPath}[{Owner}] -> {Size}"

== CMD\_DESCRIPTION\_FIND\_QUERY ==

Obtiene una serie de objetos en base a unos criterios de búsqueda.

== CMD\_USAGE\_FIND\_QUERY ==

Sintaxis:

cm find objeto [where str\_condiciones]

[on repository 'repspec' | on repositories <'repspec'>+]

[--format=str\_format] [--dateformat=date\_format]

[--nototal] [--file=dump\_file] [--xml] [--encoding=name]

cm find objeto [where condiciones] [on repositorios] [opciones]

objeto Objeto a buscar.

Lee la guía sobre'cm find' para ver los objetos que

puedes buscar:

https://www.plasticscm.com/documentation/cmfind/plastic-scm-version-control-query-system-guide.shtml

Opciones:

condiciones Condiciones de búsqueda sobre los atributos del objeto.

repspec Nombre o especificación del repositorio donde buscar.

En el caso de "repositories", debes separar las

especificaciones (repspec) por comas.

Usa 'cm help objectspec' para obtener más información

sobre las especificaciones de repositorios.

--format Devuelve el mensaje de salida en un formato determinado.

Lee la guía sobre 'cm find' para ver todos los atributos

de los objetos que puedes usar para formatear la salida:

https://www.plasticscm.com/documentation/cmfind/plastic-scm-version-control-query-system-guide.shtml

--dateformat Formato de las fechas. Por defecto "dd/MM/yyy HH:mm:ss".

--nototal No muestra el número total de registros.

--file Fichero en el que se vuelcan los resultados.

--xml Formatea salida como XML.

--encoding Especifica la codificación con la que se quiere mostrar

la salida del comando o escribir el fichero de

resultado, por ejemplo, utf-8.

Consulte la documentación de MSDN para obtener

la tabla completa de codificaciones soportadas y su

formato (al final de la página, en la columna "Name"):

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

== CMD\_HELP\_FIND\_QUERY ==

Notas:

Si no se especifica ningún repositorio, la búsqueda se realiza sobre el

repositorio actual.

Cuando se ejecutan búsquedas desde la línea de comandos usando operadores de

comparación (>, <, >=, <=), el intérprete de comandos (o shell) considera a

estos operadores como redirecciones de E/S. Por lo que es necesario escribir

la búsqueda entre dobles comillas.

El comando cm find acepta una cadena de formato para mostrar la salida.

Cada parámetro de salida se identifica con un nombre, y el usuario se

puede referir a él escribiéndolo entre los símbolos '{' y '}'.

Los parámetros de salida se corresponden en la mayoría de los casos con

los atributos del objeto.

Estas son cadenas de formato válidas:

--format={id}{date}{name}

--format="{item}#{branch} con fecha {date}"

Consideraciones sobre XML y codificación (encoding):

Cuando se especifica la opción --xml, se muestra por consola el resultado

como un texto XML. Por defecto se utiliza la codificación estándar del

sistema operativo. Es posible que si hay caracteres extraños (tildes o

eñes), no se visualicen correctamente en la consola. Pero si se redirige la

salida a un fichero, el contenido se visualizará correctamente.

Cuando se especifican simultáneamente las opciones --xml y --file, se

utiliza por defecto la codificación UTF-8.

Ejemplos:

cm find revision

cm find revision "where changeset = 23 and owner = 'user'"

cm find branch "on repository 'rep1'"

cm find label "on repositories 'rep1', 'rep:default@localhost:8084'"

cm find branch "where parent = 'br:/main' on repository 'rep1'"

cm find revision "where item = 'item:.'" --format="{item}#{branch}"

cm find revision "where item = 'item:.'" --xml --file=c:\queryresults\revs.xml

== CMD\_DESCRIPTION\_FINDCHANGED ==

Obtiene una lista de los ficheros que se han modificado fuera del control de

Plastic SCM. Este comando se considera obsoleto y se mantiene únicamente por

retrocompatibilidad. Reemplazado por cm status.

== CMD\_USAGE\_FINDCHANGED ==

fuera del control de Plastic

Sintaxis:

cm findchanged [opciones][ruta]

ruta: Ruta inicial de búsqueda.

Opciones:

-R|-r|--recursive Descender recursivamente dentro de directorios.

--checkcontent Comparar los ficheros por el contenido.

--onlychanged Con esta opción los elementos desprotegidos no serán

devueltos entre los cambiados.

== CMD\_HELP\_FINDCHANGED ==

Notas:

El comando findchanged se utiliza para identificar los items que se han

modificado en el espacio de trabajo. Si no se utiliza opción

–checkcontent solamente se comparan los timestamp para obtener los ficheros

cambiados. Si se utiliza la opción –checkcontent, se compara el contenido del

fichero.

Cuando por alguna razón no podemos obtener conexión con el servidor y se

trabaja desconectado, y se modifica una serie de ficheros, se puede utilizar

este comando para identificar los elementos cambiados.

Ejemplos:

cm findchanged .

cm findchanged -R . | cm checkout - (desprotege los elementos cambiados)

== CMD\_DESCRIPTION\_FINDCHECKEDOUT ==

Obtiene un listado de desprotecciones pendientes. Este comando está deprecado y

se mantiene únicamente por retrocompatibilidad. Reemplazado por cm status.

== CMD\_USAGE\_FINDCHECKEDOUT ==

Sintaxis:

cm findcheckouts [opciones]

Opciones:

--all: Muestra desprotecciones de todos los usuarios.

--format=cadena: Cadena de formato de salida.

Ver notas para más información.

== CMD\_HELP\_FINDCHECKEDOUT ==

Notas:

Este comando se utiliza para encontrar las protecciones pendientes en un

espacio de trabajo. Se puede combinar con el comando 'checkin' para

proteger todos los elementos desprotegidos, utilizando la redirección de

la salida estándar. Vea los ejemplos para más información.

Este comando acepta una cadena de formato para mostrar la salida.

Cada parámetro de salida se identifica con un número, y el usuario se puede

referir a él escribiéndolo entre los símbolos '{' y '}'.

Los parámetros de salida de este comando son los siguientes:

0: Fecha de modificación.

1: Propietario.

2: Información del espacio de trabajo.

3: Máquina.

4: Ruta del ítem.

5: Información de repositorio y rama.

Éstas son cadenas de formato válidas:

--format={0}{1}{2}{3}{4}{5}

--format="El fichero {4} está desprotegido por el usuario {1}"

Ejemplos:

cm findcheckouts

cm findcheckouts --all

cm findcheckouts --format={4} | cm checkin -

(Protege todos los items desprotegidos)

cm findcheckouts --format={4} | cm uncheckout -

(Deshace las desprotecciones de todos los items desprotegidos)

== CMD\_DESCRIPTION\_FINDPRIVATE ==

Obtiene un listado de items privados en el espacio de trabajo. Este comando

está deprecado y se mantiene únicamente por retrocompatibilidad. Reemplazado

por cm status.

== CMD\_USAGE\_FINDPRIVATE ==

Sintaxis:

cm findprivate [opciones][ruta]

rutas: Ruta inicial de búsqueda.

Opciones:

-R|-r|--recursive Descender recursivamente dentro de directorios.

--exclusions Ignora los elementos excluidos.

== CMD\_HELP\_FINDPRIVATE ==

Notas:

Si no se proporciona una ruta, se tomará el directorio actual.

Este comando es útil para añadir solamente los elementos privados de una

ruta al repositorio, utilizando la redirección de la salida estándar.

Ver ejemplos.

Ejemplos:

cm findprivate .

cm findprivate -R | cm add - --coparent

(busca recursivamente items privados, y los añade al

repositorio desprotegiendo los padres si es necesario)

== CMD\_DESCRIPTION\_GETCONFIG ==

Obtiene la información de configuración solicitada.

== CMD\_USAGE\_GETCONFIG ==

Sintaxis:

cm getconfig setting

setting: Parámetro de configuración que se quiere consultar.

Los parámetros disponibles son:

setfileasreadonly : indica si los ficheros protegidos se dejan en sólo lectura.

location: devuelve la ruta del fichero de configuración del cliente.

extensionworkingmode: devuelve el modo de trabajo de la extensión.

extensionprefix: devuelve el prefijo de la extensión configurada.

defaultrepserver: devuelve el servidor de repositorios configurado por defecto.

Devuelve el valor de la clave solicitada.

== CMD\_HELP\_GETCONFIG ==

Ejemplos:

cm getconfig setfileasreadonly

== CMD\_DESCRIPTION\_GETFILE ==

Obtiene una revisión especificada.

== CMD\_USAGE\_GETFILE ==

Sintaxis:

cm cat spec [opciones]

spec: Especificación de la revisión. Usa 'cm help objectspec' para obtener

más información sobre especificaciones.

Opciones:

--file=fichero: Fichero de salida, opcional. Si no se especifica

el resultado, se escribe en la salida estándar.

--debug: Cuando se especifica la revisión de un directorio, el

comando muestra su id de revisión y sus permisos.

== CMD\_HELP\_GETFILE ==

Notas:

Una especificación de revisión se define como:

[rev:]RUTA[#(brSpec|csetSpec|lbSpec)][@repSpec]

[rev:]revid:rev\_id[@repSpec]

[rev:]serverpath:ruta\_de\_servidor#(brSpec|csetSpec|lbSpec)[@repSpec]

brSpec: br:NOMBRE\_RAMA[#LAST]

csetSpec: cs:NUMERO\_CSET

lbSpec: lb:NOMBRE\_ETIQUETA

repSpec: [rep:]NOMBRE\_REPOSITORIO[@[repserver:]NOMBRE\_SERVIDOR:PUERTO\_SERVIDOR]

Ejemplos:

Obtener la última revisión de la rama main del ítem fichero.txt:

cm cat fichero.txt#br:/main

Obtener la revisión del changeset 3 del ítem fichero.txt y escribirlo en el

fichero tmp.txt:

cm cat fichero.txt#cs:3 --file=tmp.txt

Obtener la revisión id 1230:

cm cat revid:1230@rep:myrepos@repserver:london:8084

Obtener el fichero /src/foo.c en el último changeset de la rama

br:/main/task003 del repositorio 'myrepo':

cm cat serverpath:/src/foo.c#br:/main/task003@myrepo

== CMD\_DESCRIPTION\_GETREVISION ==

Trae una revisión especifica al espacio de trabajo actual

== CMD\_USAGE\_GETREVISION ==

Sintaxis:

cm getrevision revision\_spec

revision\_spec: Especificación de la revisión que se desea obtener. Usa

'cm help objectspec' para obtener más información sobre especificaciones.

== CMD\_HELP\_GETREVISION ==

Notas:

Una especificación de revisión se define como:

[rev:]RUTA[#(brSpec|csetSpec|lbSpec)][@repSpec]

[rev:]revid:[#(brSpec|csetSpec|lbSpec)][@repSpec]

brSpec: br:NOMBRE\_RAMA[#LAST]

csetSpec: cs:NUMERO\_CSET

lbSpec: lb:NOMBRE\_ETIQUETA

repSpec: [rep:]NOMBRE\_REPOSITORIO[@[repserver:]NOMBRE\_SERVIDOR:PUERTO\_SERVIDOR]

Ejemplos:

cm getrevision fichero.txt#cs:3 (Establece la revisión del changeset 3 del

fichero.txt)

== CMD\_DESCRIPTION\_GETSTATUS ==

Obtiene el estado de un ítem.

== CMD\_USAGE\_GETSTATUS ==

Sintaxis:

cm getstatus [opciones] rutas

rutas: Rutas de los items sobre los que se desea consultar su estado.

Opciones:

-R|-r|--recursive Descender recursivamente dentro de directorios.

== CMD\_HELP\_GETSTATUS ==

Notas:

Los estados posibles de un ítem son los siguientes:

privado: El ítem no está controlado por Plastic SCM.

desprotegido: El ítem está desprotegido, y por tanto controlado por Plastic SCM.

protegido: El ítem está protegido, y por tanto controlado por Plastic SCM.

Ejemplos:

cm getstatus fichero1.txt fichero2.txt

== CMD\_DESCRIPTION\_GETTASKBRANCHES ==

Obtiene las ramas asociadas con la tarea especificada.

== CMD\_USAGE\_GETTASKBRANCHES ==

Sintaxis:

cm gettaskbranches tarea [opciones]

tarea: el identificador de la tarea.

Opciones:

--format=cadena: Cadena de formato de salida.

--dateformat: Formato de las fechas.

== CMD\_HELP\_GETTASKBRANCHES ==

Notas:

Este comando admite una cadena de formato para mostrar la salida.

Los parametros de salida de este comando son los siguientes:

tab: Introduce un tabulador.

newline: Introduce una nueva línea.

name: El nombre de la rama.

owner: El propietario de la rama.

date: La fecha en la que fue creada la rama.

type: El tipo de la rama ('T' si es smart o 'F' si no lo es).

parent: La rama padre.

comment: El comentario de la rama.

repname: El repositorio donde se encuentra la rama.

repserver: El nombre del servidor.

Ejemplos:

cm gtb 4311

cm gtb 4311 --format="br:{name}"

cm gtb 4311 --format="br:{name} {date}" --dateformat="yyyy/MM/dd HH:mm:ss"

== CMD\_DESCRIPTION\_GETWWI ==

Muestra información acerca del selecetor del espacio de trabajo.

== CMD\_USAGE\_GETWWI ==

Uso:

cm wi [ruta\_workspace]

ruta\_workspace: Ruta en disco del workspace a consultar.

== CMD\_HELP\_GETWWI ==

Notas:

Este comando muestra información acerca de la configuración del selector

del espacio de trabajo (en qué repositorio, rama y etiqueta se está

trabajando).

Ejemplos:

cm wi c:\mywk

== CMD\_DESCRIPTION\_GWP ==

Obtiene información acerca del workspace a partir de un path.

== CMD\_USAGE\_GWP ==

Uso:

cm getworkspacefrompath | gwp <ruta> [--format=str\_format]

ruta Fichero o directorio en disco.

Opciones:

--format Devuelve el mensaje de salida con un formato determinado.

Más información en las notas.

== CMD\_HELP\_GWP ==

Notas:

Este comando muestra por pantalla información del workspace al que

pertenece una ruta determinada.

Este comando acepta una cadena especificando el formato con el que se

debe mostrar la salida. Los parámetros de salida son los siguientes:

{0} | {wkname} Nombre del workspace.

{1} | {wkpath} Ruta del workspace.

{2} | {machine} Nombre de la máquina del workspace.

{3} | {owner} Propietario del workspace.

{4} | {guid} GUID del workspace.

{tab} Inserta un tabulador.

{newline} Inserta un salto de línea.

Ejemplos:

cm gwp c:\myworkspace\code\file1.cpp --format="Nombre del wkspace: {wkname}"

== CMD\_DESCRIPTION\_HELP ==

Muestra ayuda acerca de un comando.

== CMD\_USAGE\_HELP ==

Sintaxis:

cm help nombre\_comando

== CMD\_HELP\_HELP ==

== CMD\_DESCRIPTION\_IOSTATS ==

Muestra estadísticas de rendimiento relativas al hardware

== CMD\_USAGE\_IOSTATS ==

Sintaxis:

cm iostats [server\_spec] list\_of\_tests [options]

server\_spec: Servidor válido de PlasticSCM para ejecutar las pruebas de red,

tales como "serverUploadTest" y/o "serverDownloadTest".

Si no se especifica ningún parámetro, se intentará usar el servidor

configureado por defecto.

Usa 'cm help objectspec' para aprender cómo indicar la especificación

del servidor.

list\_of\_tests: Test disponibles:

--serverUploadTest: mide la velocidad de subida de datos a través de la

red desde el cliente de Plastic SCM al servidor.

--serverDownloadTest: mide la velocidad de bajada de datos a través de

la red desde el servidor de Plastic SCM al cliente.

--diskTest: mide la velocidad de escritura y lectura a disco.

--systemNetworkUsage: muestra el uso actual de los recursos de red del

sistema. (Se usan contadores de rendimiento del sistema operativo).

(Sólo disponible en plataformas Windows).

--systemDiskUsage: muestra el uso actual de los discos físico del

sistema. (Se usan contadores de rendimiento del sistema operativo).

(Sólo disponible en plataformas Windows).

Opciones (options):

--netTotalMB: cantidad de datos de usuario (en MegaBytes) que se

transmitirán en un test de red, tales como"serverDownloadTest"

o "serverUploadTest".

Debe ser un valor comprendido entre "4" y "512" (por defecto: 16).

--networkIterations: indica el número de iteraciones para los tests de red:

"serverUploadTest" o "serverDownloadTest".

Debe ser un valor comprendido entre "1" y "100". (por defecto: 1).

--diskDataSize: cantidad de datos (en MegaBytes) que se escribirán y leerán

en el test de disco ("diskTest").

Debe ser un valor comprendido entre "100" y "4096". (por defecto: 512).

--diskTestPath: Ruta donde el test de disco escribe los datos para la

realización de la prueba. Si no se especifica ningún parámetro, se

intentará usar un directorio dentro del directorio temporal del sistema.

--systemDisplayTime: Espacio de tiempo (en segundos) en el que se muestra el uso

de los recursos del sistema. Éste parámetro es tenido en cuenta para los tests:

"systemNetworkUsage" y "systemDiskUsage".

Debe ser un valor comprendido entre "1" y "3600". (por defecto: 5 segundos).

--systemDisplayTimeInterval: Periodo de tiempo (en segundos) entre las muestras

que se toman sobre el uso de los recursos del sistema. Este parámetro

es tenido en cuenta para los tests: "systemNetworkUsage" y "systemDiskUsage".

Debe ser un valor comprendido entre "1" y "60". (por defecto: 1 segundo).

Ejemplo:

cm iostats MYSERVER:8087 --serverUploadTest --serverDownloadTest --netTotalMB=32

== CMD\_HELP\_IOSTATS ==

Notas:

Éste comando necesita un servidor de Plastic SCM que se usará para las

pruebas de velocidad de la red ("serverUploadTest" y/o

"serverDownloadTest").

El parámetro "--diskTestPath" debe apuntar a una ruta que pertenezca a la

unidad física de disco que se pretende probar.

Si no se especifica ninguna ruta, el comando intentará utilizar la ruta

de almacenamiento temporal del sistema.

La unidad de disco del directorio de pruebas de disco debe contener suficiente

espacio libre para ejecutar la prueba.

Durante la ejecución del comando se puede experimentar una caída de rendimiento

de la máquina debido a las pruebas realizadas.

== CMD\_DESCRIPTION\_ISSUETRACKER ==

Obtiene, actualiza o busca el estado de una tarea del sistema de incidencias especificado.

== CMD\_USAGE\_ISSUETRACKER ==

Sintaxis:

cm issuetracker <name> status get <task\_id> <parameter>+

cm issuetracker <name> status update <task\_id> <status> <parameter>+

cm issuetracker <name> status find <status> <parameter>+

cm issuetracker <name> connection check <parameter>+

name: Nombre del sistema de seguimiento de incidencias contra

el que se va a conectar.

Sólo Jira está soportado por el momento.

task\_id: Número de la tarea que se quiere consultar o actualizar.

status: Un estado de tarea válido en el sistema especificado.

Parámetros de Jira (son todos obligatorios):

--user=user El usuario a autenticar.

--password=pwd La contraseña a autenticar.

--host=url La url del sistema de seguimiento de incidencias.

--projectkey=clave La clave del proyecto en Jira.

== CMD\_HELP\_ISSUETRACKER ==

Ejemplos:

cm issuetracker jira status get 11 --user=user@mail.es --password=pwd --host=https://user.atlassian.net --projectkey=PRJ

(Obtiene el estado de la tarea 11 para el proyecto PRJ)

cm issuetracker jira status update 11 "Done" --user=user@mail.es --password=pwd --host=https://user.atlassian.net --projectkey=PRJ

(Actualiza el estado a "Done" de la tarea 11 para el proyecto PRJ)

cm issuetracker jira status find "Done" --user=user@mail.es --password=pwd --host=https://user.atlassian.net --projectkey=PRJ

(Obtiene los ids de tarea cuyo estado es "Done" para el proyecto PRJ)

cm issuetracker jira connection check --user=user@mail.es --password=pwd --host=https://user.atlassian.net

(Comprueba si los parámetros de configuración son válidos o no)

== CMD\_DESCRIPTION\_LICENSEINFO ==

Muestra información de la licencia y de su uso.

== CMD\_USAGE\_LICENSEINFO ==

Sintaxis:

cm licenseinfo | li [--server=repserverspec] [--inactive] [--active] [--sort={name|status}]

Opciones:

--server: Obtiene la información de la licencia para el servidor

especificado. Si no se especifica ningún servidor, el

comando se ejecutará contra el servidor configurado en

el cliente.

Usa 'cm help objectspec' para obtener más información

sobre especificaciones.

--inactive Muestra únicamente usuarios inactivos en la sección de

"uso de licencia".

--active Muestra únicamente usuarios activos en la sección de

"uso de licencia".

--sort Ordena usuarios usando el campo especificado:

'name' o 'status'.

== CMD\_HELP\_LICENSEINFO ==

Notas:

Se muestra información acerca de la fecha de expiración, los usuarios

activos y no activos, etc.

Ejemplos:

cm licenseinfo

cm licenseinfo --server=myserver:8084

== CMD\_DESCRIPTION\_LINKTASK ==

Enlaza un changeset a tareas pertenecientes a un control de tareas.

== CMD\_USAGE\_LINKTASK ==

Sintaxis:

cm linktask changesetSpec extensionPrefix taskKey

changesetSpec: Especificación completa del changeset para ser enlazado a una

tarea en el sistema de control de tareas. Usa 'cm help objectspec' para

obtener más información sobre especificaciones.

extensionPrefix: Prefijo del sistema de control de tareas configurado.

taskKey: Identificador de la tarea para asociar al changeset especificado.

== CMD\_HELP\_LINKTASK ==

Ejemplos:

cm lt cs:8@rep:default@repserver:localhost:8084 jira PRJ-1

== CMD\_DESCRIPTION\_LOCK\_LIST ==

Muestra los elementos bloqueados (checkouts exclusivos) en el servidor especificado.

== CMD\_USAGE\_LOCK\_LIST ==

Sintaxis:

cm lock list [revspec [revspec ...]] [flags]

revspec: especificación de revisión. Usa 'cm help objectspec' para aprender

cómo indicar las especificaciones de las revisiones.

Opciones:

flags:

--server=<server>: muestra los elementos bloqueados en el

servidor, sobreescribiendo el valor por defecto (tomado del espacio

de trabajo actual o del fichero client.conf).

Usa 'cm help objectspec' para más información sobre especificaciones.

--onlycurrentuser: Filtra los resultados mostrando únicamente los

bloqueos efectuados por el usuario actual de Plastic SCM.

--onlycurrentworkspace: Filtra los resultados mostrando únicamente los

bloqueos efectuados en el espacio de trabajo actual. (Se cotejan

los resultados por nombre de espacio de trabajo).

--ignorecase: Ignora mayúsculas y minúsculas cuando una servepath spec

es usada. Con esta opción el comando funcionará para "/src/foo.c"

incluso si el usuario indica"/sRc/fOO.c".

== CMD\_HELP\_LOCK\_LIST ==

Ejemplos:

cm lock list

cm lock ls --server=DIGITALIS:8084

cm lock ls serverpath:/src/foo.c#cs:99@default@localhost:8084

cm lock ls revid:3521@default itemid:2381@secondary --onlycurrentuser

cm lock ls --onlycurrentuser

cm lock ls --onlycurrentuser --onlycurrentworkspace

Notas:

El comando muestra una lista de los elementos actualmente bloqueados en el

servidor por defecto. También acepta una lista de especificaciones de

revisión; en ese caso, solo se mostrarán los elementos bloqueados que

estén asociados a al menos una de las revisiones especificadas como

argumentos del comando. Adicionalmente, se puede utilizar el parámetro

opcional --server=<server> para modificar el servidor que se

consultará por defecto.

El comando muestra una línea por cada bloqueo almacenado en el servidor,

con la siguiente información:

GUID del elemento bloqueado.

Nombre de usuario que realizó el bloqueo.

Ruta del elemento bloqueado (formato de ruta de servidor).

Nombre del espacio de trabajo donde se realizó el bloqueo.

== CMD\_DESCRIPTION\_LISTUSERS ==

Lista los usuarios y grupos conocidos por el servidor.

== CMD\_USAGE\_LISTUSERS ==

Sintaxis:

cm listusers server [opciones]

server: especificación del servidor de repositorios.

Usa 'cm help objectspec' para más información sobre especificaciones.

Opciones:

--onlyusers (-u): Lista sólo los usuarios.

--onlygroups (-g): Lista sólo los grupos.

--filter : Lista sólo aquellos usuarios y/o grupos que coincidan con el

filtro de busqueda indicado.

== CMD\_HELP\_LISTUSERS ==

Ejemplos:

cm listusers localhost:8084

cm listusers localhost:8084 -u

== CMD\_DESCRIPTION\_LOCATION ==

Muestra la ruta completa donde se ubica el ejecutable del cliente.

== CMD\_USAGE\_LOCATION ==

Sintaxis:

cm location

== CMD\_HELP\_LOCATION ==

== CMD\_DESCRIPTION\_LOCK ==

Use este comando para administrar workspaces.

== CMD\_USAGE\_LOCK ==

Sintaxis:

cm lock command [options]

Comandos:

list | ls

unolck

Para obtener más información sobre cada comando:

cm lock comando --usage

cm lock comando --help

== CMD\_HELP\_LOCK ==

Ejemplos:

cm lock list

cm lock

(list es opcional si no hay argumentos)

cm lock ls serverpath:/src/foo.c#cs:99@default@localhost:8084

cm lock unlock 91961b14-3dfe-4062-8c4c-f33a81d201f5

== CMD\_DESCRIPTION\_LOG ==

Devuelve información sobre las revisiones creadas en la rama del changeset

especificado, o de un intervalo de changesets.

== CMD\_USAGE\_LOG ==

Sintaxis:

cm log [repSpec|csSpec] [--from|-f=csSpecFrom] [--allbranches|-a]

[--csFormat=format] [--itemFormat=format] [--xml[=outputfile]]

[--encoding] [--repositorypaths | --fullpaths]

repSpec: Especificación del repositorio del cual se obtendrá información de

las revisiones en sus changesets. Usa 'cm help objectspec' para más información.

csSpec: Especificación del changeset del cual se obtendrá la información de

sus revisiones. Usa 'cm help objectspec' para más información.

Opciones:

--from, -f=csSpecFrom: Especificación del changeset a partir del cual se

mostrará información de revisiones. Dicho changeset no se incluye en la

salida. Ignorado si se indica una especificación de un repositorio.

--allbranches, -a: Muestra información de los changesets creados en un

intervalo especificado para cualquier rama.

--ancestors: muestra la información acerca de los changesets alcanzables

siguiendo los enlaces de parentesco y de merge para el changeset dado

([csSpec]). Si el changeset from ([csSpecFrom]) también es proporcionado,

este será usado como límite inferior para todos los paths.

Notas: los cambios de los changeset no se mostrarán con esta opción.

--csFormat=format: Especificación del formato en el cual se desea obtener

la información de los changesets.

--itemFormat=format: Especificación del formato en el cual se desea obtener

la información de los items involucrados en cada changeset.

--xml: Permite volcar la salida del comando en formato xml a la salida

estándar. También se permite especificar un fichero de salida

(--xml=output.xml).

--fullpaths, --fp: Fuerza al comando a imprimir rutas de workspace

absolutas siempre que sea posible.

--repositorypaths: imprime rutas de repositorio en lugar de rutas

de workspace. (Ésta opción prevalece sobre la opción "--fullpaths").

--encoding: si se utiliza con la opción --xml, permite especificar el

encoding con el que se guardará la salida del comando. Consulte la

documentación de la MSDN para obtener la tabla completa de codificaciones

soportadas y su formato; al final de la página en la columna "Name":

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

== CMD\_HELP\_LOG ==

Notas:

Si no se especifica ni changeset ni opción, mostrará información de todos

los changeset creados en el último mes en cualquier rama.

Si solamente se incluye la opción -f=[csSpecFrom], se mostrará la

información de todos los changesets desde el especificado hasta el momento

actual en la rama en la cual se creó dicho changeset.

Si se especifica la opción -a sin un intervalo, a todos los efectos el

comando funcionará exactamente igual que si solamente se especificara

[csSpec].

Si se utiliza la opción -f=[csSpecFrom], la salida muestra información

desde el changeset [csSpecFrom]+1 en adelante.

El repositorio que se utiliza para mostrar la información de los changesets

es el repositorio cargado en el path desde el cual se lanza el comando.

Este comando acepta una cadena de formato para mostrar la salida.

Se puede especificar un formato para la información de los changeset y

otro para la información de los ítems.

Cada parámetro de salida se identifica con un descriptor, y el usuario se

puede referir a él escribiéndolo entre los símbolos '{' y '}'.

Los parámetros de salida de este comando son los siguientes:

csFormat=

tab: Introduce un tabulador.

newline: Introduce una nueva línea.

changesetid: Número de changeset.

branch: Información de rama.

date: Fecha de creación del changeset.

owner: Propietario del changeset.

comment: Comentarios del changeset.

items: Items afectados en el changeset.

repository: Nombre del repositorio donde se encuentra el changeset.

repserver: Nombre del servidor.

itemFormat=

tab: Introduce un tabulador.

newline: Introduce una nueva línea.

path: Ruta del ítem en disco.

branch: Información de rama.

owner: Propietario del changeset.

date: Fecha de creación del changeset.

shortstatus: Formato corto. Ver más abajo.

fullstatus: Formato largo. Ver más abajo.

Formato corto y su correspondiente formato largo:

A = Added.

C = Changed.

M = Moved.

D = Deleted.

Éstas son cadenas de formato válidas:

--csFormat="{newline}Changeset {changesetid} generado el día {date};{tab} ítems cambiados: {items}."

--itemFormat="{newline}El ítem {path} se ha cambiado en la rama {branch} en el changeset {changesetid}."

Ejemplos:

cm log

(Muestra información de todos los changesets creados en el último mes en

cualquier rama).

cm log cs:16

(Muestra la información de los cambios realizados en el changeset 16 en la

rama en que se creó dicho changeset).

cm log cs:16 --csFormat="{newline}Changeset {changesetid} generado el día {date};{tab} ítems cambiados: {items}."

(Muestra la información el el formato especificado).

cm log --from=cs:20 cs:50

(Muestra información de todos los cambios de cada changeset comprendido

entre el 21 y el 50 en la rama br:/main).

cm log --from=cs:20 cs:50 -a

(Muestra información de todos los cambios de cada changeset comprendido

entre el 21 y el 50 en todas las ramas del repositorio).

cm log rep:myrep@localhost:8084

(Muestra la información de todos los changeset de otro repositorio diferente

al configurado en el selector del workspace. Puede ejecutarse fuera de

un workspace).

== CMD\_DESCRIPTION\_LS ==

Lista el contenido de un workspace o de un árbol de changeset, rama o selector.

== CMD\_USAGE\_LS ==

Sintaxis:

cm ls [opciones][rutas]

rutas: Rutas de los items para obtener el listado.

Opciones:

--format=cadena: cadena de formato

--symlink: aplica la operación al propio symlink y no a su destino

--selector: obtiene el contenido del selector del workspace actual

--selector="selector": obtiene el contenido del selector especificado

--tree=spec: lista el árbol de un changeset (a partir de su número o guid)

o de una rama (a partir de su nombre completo). Usa 'cm help objectspec'

para aprender cómo indicar una especificación.

-R|-r|--recursive: recorre los directorios recursivamente

--xml: Permite volcar la salida del comando en formato xml a la salida

estándar. También se permite especificar un fichero de salida

(--xml=output.xml).

--encoding: si se utiliza con la opción --xml, permite especificar el

encoding con el que se guardará la salida del comando. Consulte la

documentación de la MSDN para obtener la tabla completa de codificaciones

soportadas y su formato; al final de la página en la columna "Name":

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

== CMD\_HELP\_LS ==

Notas:

Las rutas se pueden escribir con metacaracteres (\*, ?, ...)

La salida del comando 'ls' se puede formatear usando una 'cadena de formato'

en la cual cada parámetro se identifica mediante un nombre (o un número para

mantener compatibilidad con las versiones anteriores).

La cadena de formato por defecto es:

"{<size>,10} {<date>:dd/MM/yyyy} {<date>:HH:mm} \

{<type>,-6} {<location>,-12} {<co>,-5} \

{name} {symlinktarget}"

También se pueden especificar los nombres como {nombre} para cadenas de

formato más sencillas:

"{size} {date} {name}"

Los nombres de los parámetros válidos para la cadena de formato son:

size 0

date 1

type 2

location 3 imprime br:branch#cset

checkout 4

co 4

name 5

changeset 6

cset 6

fullpath 7

path 7

repspec 8

owner 9

formattedsize 10

revid 11

parentrevid 12

itemid 13

brid 14

repid 15

server 16

symlinktarget 17

hash 18

chmod 19

wkpath 20 imprime un path relativo a la raíz del workspace

branch 21

newlocation 22 imprime cset@branch

guid 23 imprime el guid de cada elemento (tarda más tiempo)

itemguid 24

transformed 25 muestra las reglas aplicadas en cada elemento que

haya sufrido transformación.

Se puede personalizar la salida del comando ls sin necesidad de especificar

--format cada vez, definiendo la variable de entorno PLASTIC\_LS\_FORMAT.

Ejemplos:

cm ls

cm ls c:\workspace\src

cm ls --format={name} (muestra nombres de ficheros y directorios)

cm ls --symlink (muestra información de los symlinks y no del fichero o

directorio al que apunta. Disponible en entornos Linux/Mac/Unix)

cm ls code --selector (muestra el contenido del subdirectorio 'code' cargado

del selector asociado al workspace actual)

cm ls /code --selector="rep 'arevalo' path '/' branch '/main'" (muestra el

contenido del subdirectorio '/code' cargado del selector especificado.

Es importante destacar que el path se especifica en "formato servidor")

cm ls /code --tree=44@arevalo@cyl:7070 (lista el contenido del

subdirectorio /code en el changeset 44 del repositorio 'arevalo' en el

servidor 'cyl:7070')

cm ls /code --tree=br:/main/scm13596@arevalo@cyl:7070 (lista el contenido del

subdirectorio /code en el último changeset de la rama /main/scm13596

del repositorio 'arevalo' en el servidor 'cyl:7070')

cm ls /code --tree=ae1390ed-7ce9-4ec3-a155-e5a61de0dc77@arevalo@cyl:7070

(lista el subdirectorio /code en el changeset con guid

ae1390ed-7ce9-4ec3-a155-e5a61de0dc77

en el repositorio 'arevalo' del servidor 'cyl:7070')

== CMD\_DESCRIPTION\_TRIGGER\_LIST ==

Muestra la información relativa a los triggers de un determinado tipo.

== CMD\_USAGE\_TRIGGER\_LIST ==

Sintaxis:

cm trigger list [subtipo-tipo] [opciones]

subtipo-tipo: momento de ejecución del trigger y operacion del trigger.

Teclee 'cm showtriggertypes' para ver una lista completa de los tipos de

trigger disponibles. Algunos ejemplos son:

before-mkbranch

after-mkwokspace

after-setselector

Opciones:

--server=servidor: Lista los triggers del servidor especificado.

--format: formato de salida.

== CMD\_HELP\_TRIGGER\_LIST ==

Notas:

Si no se especifica tipo, muestra todos los triggers del servidor

configurado en el cliente.

Formato de salida:

{0}: Posición.

{1}: Nombre del trigger.

{2}: Ruta del ejecutable.

{3}: Propietario.

{4}: Tipo.

{5}: Filtro del trigger.

Ejemplos:

cm trigger list after-mklabel

cm tr ls before-mkbranch --server=myserver:8084

== CMD\_DESCRIPTION\_MANIPULATESELECTOR ==

Permite cambiar el selector para reflejar cambios en los comandos.

== CMD\_USAGE\_MANIPULATESELECTOR ==

Uso:

cm manipulateselector [wkpath | wkspec] [options]

path: Ruta del espacio de trabajo.

wkspec: Especificación del espacio de trabajo. Usa 'cm help objectspec'

para más información sobre especificaciones.

Opciones:

--atdate=date: Devuelve un selector que recreará el espacio de trabajo con

los contenidos que se cargarían en la fecha especificada.

== CMD\_HELP\_MANIPULATESELECTOR ==

Notas:

Si no se indica alguna de las rutas o especificaciones de espacio de

trabajo el comando supondrá el directorio actual como ruta al espacio de

trabajo.

Ejemplos:

cm manipulateselector c:\workspace --atdate=yyyy-MM-ddTHH:mm:ss

cm manipulateselector --atdate=yyyy-MM-ddTHH:mm:ss

cm manipulateselector > mySelector.txt --atdate=yyyy-MM-ddTHH:mm:ss

cm manipulateselector wk:build\_wk@BUILDER --atdate=yyyy-MM-ddTHH:mm:ss

== CMD\_DESCRIPTION\_MERGE ==

Mezcla el contenido de una rama con otra.

== CMD\_USAGE\_MERGE ==

Sintaxis:

cm merge spec\_origen [--merge] [--cherrypicking] [--forced]

[--mergetype=merge\_type]

[--interval-origin=csetspec | --ancestor=csetspec]

[--keepsource | --ks] [--keepdestination | --kd]

[--automaticresolution=<conflict-types>+]

[--subtractive] [--mount] [--printcontributors]

[--noprintoperations] [--silent]

[(--to=brspec | --destination=brspec) [--shelve]]

[-c=str\_comment | --commentsfile=<comments\_file>]

[--resolveconflict --conflict=index

--resolutionoption=src|dst|<rename --resolutioninfo=strname>

--mergeresultfile=path --solvedconflictsfile=path]

[--nointeractiveresolution]

[--machinereadable [--startlineseparator=sep]

[--endlineseparator=sep] [--fieldseparator=sep]]

spec\_origen Especificación del origen del merge. Este puede ser:

- una rama: [br:]/nombre\_rama

- una etiqueta: lb:nombre\_etiqueta

- un changeset: cs:numero\_changeset

- un shelve: sh:numero\_shelve

Usa 'cm help objectspec' para más información sobre

especificaciones.

Opciones:

--merge Realiza el merge. Si no se especifica esta opción,

se muestran los conflictos encontrados.

--cherrypicking Realiza el merge de los cambios incluidos en los

changesets origen. Esta opción no se considera si

el origen del merge es una etiqueta.

--forced No comprueba si el origen y el destino ya están

conectados.

Esta opción solo es válida para el merge de intervalo

y para el cherrypicking.

--mergetype Especifica el tipo de merge. Puede ser uno de los

siguientes: onlyone, onlysrc, onlydst, try, forced.

--interval-origin Especificación del changeset que queremos que actúe

como origen del intervalo, de manera que el merge

introducirá los cambios realizados desde el

changeset común hasta el changeset especificado

como origen del merge.

--ancestor Alias para --interval-origin.

--keepsource Acepta todos los cambios del contribuidor origen

para elementos con conflictos.

--keepdestination Preserva los cambios del contribuidor destino

para elementos con conflictos.

--automaticresolution Opción para resolver conflictos de directorio. Esta

opción permite elegir si el contribuidor origen

(source) o el contribuidor destino (destination)

debe ser seleccionado automáticamente para

resolver el conflicto.

Debes especificar los tipos de conflictos separados

por punto y coma ";".

Más información en las notas.

--subtractive Se utiliza para quitar cambios ya introducidos en

el sistema, ya sea por un checkin o por un merge.

El parámetro que se pasa al comando (spec\_origen)

es el origen desde donde se quieren eliminar los

cambios. Puede ser la especificación de un

changeset o de un intervalo de ellos. En el caso

de un intervalo de changesets, se utiliza la opción

--interval-origin para definir el origen del

intervalo.

--mount El punto de montaje del respositorio especificado.

--printcontributors Imprime los contribuidores (base, origen y destino).

--noprintoperations Resuelve las mezclas de manera silenciosa sin

mostrar información sobre la resolución.

--silent No muestra ninguna salida.

--to | --destination Realiza una operación de merge-to a la rama de

destino especificada, con resolución de conflictos

completa. Una mezcla "merge-to" (o mezcla sin

workspace) es una mezcla hecha en el lado del

servidor. Mientras que en mezclas normales el

destino es el workspace, en una operación de

"merge-to" se debe especificar el destino (por

eso se llama "to", 'hacia' en inglés).

En la siguiente página se puede encontrar más

información sobre esta funcionalidad:

https://www.plasticscm.com/download/help/mergeto

--shelve Crea un shelve con los cambios del resultado del

merge (incluida la trazabilidad) en lugar de crear

un changeset. Esta opción no es válida cuando el

origen del merge es un shelve. Esta opción sólo

está disponible para el merge en el lado del

servidor (merge-to). Por ello, se requieren las

opciones "--to" y "--merge".

--no-dst-changes Asegura que el contribuidor destino no tiene cambios

(que el changeset destino es también el antecesor

común). Cuando hay cambios en el destino, no se

permite hacer el merge.

-c Añade un comentario al changeset creado en la

operación de mezcla.

--commentsfile Carga el comentario del fichero especificado y lo

añade al changeset creado en la operación de

mezcla.

--resolveconflict (Usado principalmente por plugins. Más información

en las notas.)

Usado para resolver un conflicto de directorio.

--conflict Usado con '--resolveconflict', especifica el

índice del conflicto a resolver empezando por 1.

--resolutionoption Usado con '--resolveconflict', indica el tipo de

resolución de conflicto. Puede ser una de las

siguientes opciones: src, dst, rename.

Más información en las notas.

--resolutioninfo Usado con '--resolveconflict', proporciona el

nombre que se usará cuando la opción

'--resolutionoption' es "rename".

--mergeresultfile Usado con '--resolveconflict', guarda en un fichero

la información del resultado de la mezcla entre

diferentes llamadas. El fichero especificado se

crea durante la primera llamada y se actualiza en

cada siguiente llamada.

--solvedconflictsfile Usado con '--resolveconflict', guarda en un fichero

la información de los conflictos solucionados

entre diferentes llamadas. El fichero especificado

se crea durante la primera llamada y se actualiza

en cada siguiente llamada.

--nointeractiveresolution (Usado principalmente por plugins. Más información

en las notas.)

Evita preguntar al usuario cuando hay un conflicto

manual. Así, por ejemplo, un conflicto de directorio

no podrá ser resuelto.

--machinereadable (Usado principalmente por plugins. Más información

en las notas.)

Muestra la salida en un formato fácil de tratar.

--startlineseparator Usado con '--machinereadable', especifica cómo

deben comenzar las líneas. Por defecto, una cadena

de texto vacía.

--endlineseparator Usado con '--machinereadable', especifica cómo

deben terminar las líneas. Por defecto, una cadena

de texto vacía.

--fieldseparator Usado con '--machinereadable', especifica cómo

deben separarse los campos. Por defecto, un

espacio en blanco.

== CMD\_HELP\_MERGE ==

Notas:

Este comando se utiliza para meclar cambios entre dos ramas, o entre una

etiqueta y una rama. El destino del merge siempre debe ser una rama.

El origen del merge se especifica como argumento.

El destino de la rama es (salvo en el merge-to) el contenido actual del

workspace.

Por ejemplo, para mostrar los elementos que serán mezclados desde la rama

task001 a la rama principal, el selector debe apuntar a la rama principal,

el espacio de trabajo debe estar actualizado, y entonces ejecutar:

cm merge br:/task001

Para que el merge se realice, se añade el modificador "--merge":

cm merge br:/task001 --merge

El origen del merge es la especificación de un objeto (rama, etiqueta,

changeset o shelve).

Usa 'cm help objectspec' para más información sobre especificaciones.

Para resolver automáticamente conflictos de directorio debes usar la opción

"--automaticresolution" y especificar el tipo de conflicto seguido por el

contribuidor (origen o destino) que se seleccionará durante el merge.

(Recuerda separar cada par "tipo de conflicto"-"contribuidor" con punto y

coma (;).)

Por ejemplo:

cm merge cs:2634 --merge --automaticresolution=eviltwin-src;changedelete-src

(La operación de merge desde el changeset 2634 resuelve los conflictos

"eviltwin" y "changedelete" manteniendo al contribuidor origen ("-src")

en ambos conflictos.)

- El sufijo "-src" añadido después del tipo de conflicto indica al comando

merge que debe mantener los cambios del contribuidor origen (source).

- El sufijo "-dst" indicará que se mantendrán los cambios del contribuidor

destino (destination).

Esta es la lista de tipos de conflictos que soporta el comando merge:

"movedeviltwin", "eviltwin", "changedelete", "deletechange", "movedelete",

"deletemove", "loadedtwice", "addmove", "moveadd", "divergentmove",

"cyclemove", "all".

El valor "all" sobrescribe al resto de opciones. En el siguiente ejemplo,

el conflicto "eviltwin-dst" será ignorado:

cm merge br:/main/task062 --merge --automaticresolution=all-src;eviltwin-dst

Consulta el siguiente enlace para aprender más sobre conflictos de merge:

https://www.plasticscm.com/download/help/directorymerges

Estas son las opciones para el argumento --mergetype:

onlyone Merge automático si solo uno de los contribuidores modificó

el elemento.

onlysrc Merge automático si solo el contribuidor origen modificó

el elemento.

onlydst Merge automático si solo el contribuidor destino modificó

el elemento.

try Merge automático si solo un contribuidor ha modificado la

parte del fichero en conflicto (para cada conflicto).

forced Intentar resolver siempre los conflictos no automáticos.

Estas son las opciones que se usan principalmente por plugins e integraciones:

- "--resolveconflict" para resolver un conflicto de directorio. Se

deben usar las siguientes opciones:

- "--conflict" es el índice del conflicto que se quiere resolver

(empezando por 1).

- "--resolutionoption" indica el tipo de resolución de conflicto.

Puede ser:

- "src" para mantener el cambio del origen y descartar el

cambio del destino

- "dst" para mantener el cambio del destino y descartar el

cambio del origen

- "rename" (solo si el tipo de conflicto soporta esta

resolución), para renombar el destino al nombre

proporcionado en la opción "--resolutioninfo".

- "--resolutioninfo" para proporcionar el nombre a usar

con el tipo de resolución "rename"

- "--mergeresultfile" y "--solvedconflictsfile", ambos se usan

para guardar informacación de la mezcla entre las diferentes

llamadas.

- "nointeractiveresolution" indica a la mezcla que no pregunte al usuario

en caso de necesitarse una resolución manual de un conflicto.

- "--machinereadable" y "--startlineseparator", "--endlineseparator",

"fieldseparator" se usan para imprimir la salida a un formato fácil

de leer, tratar o parsear.

Por ejemplo:

cm merge --machinereadable --startlineseparator=start@\_@line

--endlineseparator=new@\_@line --fieldseparator=def#\_#sep

--mergeresultfile=C:\Users\Borja\AppData\Local\Temp\2tmp4D6C.tmp

--solvedconflictsfile=C:\Users\Borja\AppData\Local\Temp\2tmp4D6D.tmp

--resolveconflict --conflict=1 --resolutionoption=rename --resolutioninfo=bin\_dst

br:/main/task --merge

Ejemplos:

cm merge br:/task001

(No realiza la mezcla, simplemente escribe los elementos que van a ser

mezclados.)

cm merge br:/task001 --merge

(Realiza el merge desde la rama task001.)

cm merge cs:5 --merge --cherrypicking --interval-origin=cs:2

(Cherrypick del intervalo de changesets (2,5].)

cm merge cs:8 --merge --subtractive --keepdestination

(Merge sustractivo del changeset 8, preservando los cambios del destino

para aquellos elementos con conflictos.)

cm merge br:/main/task001 --to=br:/main --merge -c="Integrated new UI"

(Realiza el merge en el lado del servidor -a.k.a. "merge-to"-, desde la

rama task001 hasta la rama main, con un comentario.)

cm merge br:/main/task001 --to=br:/main --merge --shelve

(Realiza el merge en el lado del servidor desde la rama task001 a la rama

main y deja el resultado en un shelve.)

cm merge sh:2 --to=br:/main --merge --no-dst-changes

(Aplica el shelve 2 en la rama main solo si fue creado desde la cabeza

actual de main.)

== CMD\_DESCRIPTION\_ATTRIBUTE\_CREATE ==

Crea un atributo nuevo.

== CMD\_USAGE\_ATTRIBUTE\_CREATE ==

Sintaxis:

cm attribute create nombre

nombre: Nombre del atributo.

Opciones:

-c: Añade un comentario al atributo creado.

--commentsfile=comment.txt: Carga el comentario del fichero especificado.

== CMD\_HELP\_ATTRIBUTE\_CREATE ==

Ejemplos:

cm attribute create Estado

== CMD\_DESCRIPTION\_BRANCH ==

Use este comando para administrar ramas.

== CMD\_USAGE\_BRANCH ==

Sintaxis:

cm branch command [options]

Comandos:

[(create)|delete|rename|history|showmain|showmerges]

Para obtener más información sobre cada comando:

cm branch command --usage

cm branch command --help

== CMD\_HELP\_BRANCH ==

Ejemplos:

cm branch /main/scm21345

cm branch create /main/scm21345

cm branch delete /main/scm21345

cm branch rename /main/scm21345 scm21346

cm branch history /main/scm21345

cm branch showmain

cm branch showmerges file.txt

== CMD\_DESCRIPTION\_BRANCH\_CREATE ==

Crea una rama en el repositorio actual.

== CMD\_USAGE\_BRANCH\_CREATE ==

Sintaxis:

cm branch | br [create | mk] brspec [--changeset=csetspec | --label=lbspec]

[-c=str\_comment | -commentsfile=<comments\_file>]

brspec Nombre o especificación de la nueva rama.

Usa 'cm help objectspec' para más información sobre

especificaciones.

Opciones:

--changeset Changeset usado como punto de inicio de la nueva rama.

Especificación del changeset: [cs:]cs\_number.

--label Etiqueta usada como punto de inicio de la nueva rama.

Usa 'cm help objectspec' para más información sobre

especificaciones.

-c Rellena el comentario de la nueva rama con el texto

especificado.

--commentsfile Rellena el comentario de la nueva rama con el contenido

del fichero de texto especificado.

== CMD\_HELP\_BRANCH\_CREATE ==

Notas:

Para crear una rama de primer nivel, especifique el nombre sin jerarquía.

Por ejemplo:

cm br /dev

Si no se especifica el parámetro 'changeset', se utilizará el último

changeset de la rama madre. En caso de que la rama sea de primer nivel, se

utilizará como base el changeset 0.

Para especificar comentarios use el switch "-c" o "-m" del siguiente modo:

cm branch main/task001 -c="Este es el comentario"

cm branch main/task001 -m "Este es el comentario"

Se puede escoger qué editor se lanzará para escribir el comentario

mediante la variable de entorno PLASTICEDITOR. Si dicha variable de entorno

tiene un valor, y no se especifica un comentario, el editor escogido se

abrirá automáticamente para que se espefique el comentario.

Ejemplos:

cm branch task001

cm branch create task001

cm branch mk task001

cm br mk task001

(Crea la rama de primer nivel 'task001' en el repositorio al que apunta

el workspace actual.)

cm branch br:/task001/task002

(Crea la rama 'task002' como hija de 'task001'.)

cm br /main/task001@myrep@myserver:8084 -c="my comment"

(Crea la rama 'task001' como hija de 'main' en el repositorio

'myrepo@myserver:8084' con el comentario 'my comment'.)

cm br mk br:/main/task001 --changeset=2837 -commentsfile=commenttask001.txt

(Crea la rama 'task001' tomando como base el changeset '2837' de la rama

'main' con el comentario contenido en el fichero 'commenttask001.txt'.)

== CMD\_DESCRIPTION\_BRANCH\_DELETE ==

Borra una o más ramas.

== CMD\_USAGE\_BRANCH\_DELETE ==

Sintaxis:

cm branch | br delete | rm <spec\_rama>+

spec\_rama: Ramas a borrar, separadas por espacios.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de ramas.

== CMD\_HELP\_BRANCH\_DELETE ==

Notas:

Este comando borra una o más ramas.

Ejemplos:

cm branch delete /main/task001

(Borra la rama con nombre 'task001' que sea hija de 'main' en el

repositorio al que apunta el workspace actual.)

cm br rm main/task002 /main/task012@reptest@myserver:8084

(Borra la rama '/main/task002' del repositorio al que apunta el workspace

actual, y la rama '/main/task012' del repositorio 'reptest@myserver:8084'.)

== CMD\_DESCRIPTION\_BRANCH\_RENAME ==

Renombra una rama.

== CMD\_USAGE\_BRANCH\_RENAME ==

Sintaxis:

cm branch | br rename spec\_rama nuevo\_nombre

spec\_rama: Especificación de la rama que se va a renombrar.

Usa 'cm help objectspec' para obtener más información sobre

la especificación de ramas.

nuevo\_nombre Nuevo nombre para la rama.

== CMD\_HELP\_BRANCH\_RENAME ==

Notas:

Este comando renombra una rama.

Ejemplos:

cm branch rename /main/task0 task1

(Renombra la rama '/main/task0' a '/main/task1'.)

cm br rename br:/main@reptest@server2:8084 secondary

(Renombra la rama principal del repositorio 'reptest' a 'secondary'.)

== CMD\_DESCRIPTION\_BRANCH\_HISTORY ==

Muestra la historia de una rama (merges y punto de partida).

== CMD\_USAGE\_BRANCH\_HISTORY ==

Sintaxis:

cm branch | br history spec\_rama

[--dateformat=date\_format] [--machinereadable]

spec\_rama Especificación de la rama para obtener la historia.

Usa 'cm help objectspec' para obtener más información.

Opciones:

--dateformat Formato utilizado para mostrar fechas.

--machinereadable Devuelve el resultado en un formato sencillo de

parsear.

== CMD\_HELP\_BRANCH\_HISTORY ==

Ejemplos:

cm branch history br:/main/scm001@myrepository@myserver:8084

(Muestra la historia de '/main/scm001' en el repositorio 'myrepository'.)

cm br history main --dateformat="yyyy, dd MMMM" --machinereadable

(Muestra la historia de 'main' con un formato de fecha determinado y con

un formato sencillo de parsear.)

== CMD\_DESCRIPTION\_BRANCH\_SHOWMAIN ==

Muestra la rama principal del repositorio.

Este es un comando de automatización, diseñado para automatizar el comando cm.

Muy probablemente la rama principal de sus repositorios sea /main.

== CMD\_USAGE\_BRANCH\_SHOWMAIN ==

Sintaxis:

cm branch | br showmain [repspec] [--encoding=name]

[--format=format\_str] [--dateformat=date\_format]

repspec Especificación del repositorio donde buscar la rama principal.

Si no se especifica, se utilizará el repositorio al que apunte

el workspace actual.

Opciones:

--encoding Permite especificar la codificación con el que se

guardará la salida del comando. Consulte la

documentación de la MSDN para obtener la tabla completa

de codificaciones soportadas y su formato; al final de

la página en la columna "Name":

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

--format Devuelve el mensaje de salida con un formato determinado.

Más información en las notas.

--dateformat Formato utilizado para mostrar fechas.

== CMD\_HELP\_BRANCH\_SHOWMAIN ==

Notas:

Este comando muestra la rama principal de un repositorio.

Este comando acepta una cadena especificando el formato con el que se

debe mostrar la salida. Los parámetros de salida son los siguientes:

{id} Id de la rama.

{comment} Comentario.

{date} Fecha.

{owner} Propietario.

{name} Nombre.

{parent} Nombre de la rama madre.

{repository} Repositorio.

{repname} Nombre del repositorio.

{repserver} Servidor.

{changeset} Id del último changeset de la rama.

Ejemplos:

cm branch showmain

(Muestra la rama principal del repositorio del workspace actual.)

cm branch showmain repo@repserver:8084

(Muestra la rama principal del repositorio 'repo' en el servidor

'repserver:8084'.)

cm br showmain --dateformat="yyyy, dd MMMM" --encoding=utf-8

(Muestra la rama principal del repositorio, con las fechas en un determinado

formato, y la salida está codificada en utf-8.)

cm br showmain --format="{id} - {name}"

(Muestra la rama principal del repositorio, devolviendo únicamente su id y

nombre.)

== CMD\_DESCRIPTION\_BRANCH\_SHOWMERGES ==

Muestra información sobre elementos pendientes en ramas a mezclar.

== CMD\_USAGE\_BRANCH\_SHOWMERGES ==

Este es un comando de automatización, diseñado para automatizar el comando cm.

Sintaxis:

cm branch | br showmerges <item\_path>+

[--format=format\_str] [--dateformat=date\_format]

Opciones:

--format Devuelve el mensaje de salida con un formato determinado.

Más información en las notas.

--dateformat Formato utilizado para mostrar fechas.

== CMD\_HELP\_BRANCH\_SHOWMERGES ==

Notas:

Este comando acepta una cadena especificando el formato con el que se

debe mostrar la salida. Los parámetros de salida son los siguientes:

{id} Id de la rama.

{comment} Comentario.

{date} Fecha.

{name} Nombre.

{owner} Propietario.

{parent} Nombre de la rama madre.

{parentid} Id de la rama madre.

{repid} Id del repositorio.

{repository} Repositorio.

{repname} Nombre del repositorio.

{repserver} Servidor.

Ejemplos:

cm branch showmerges file.txt

(Muestra las ramas involucradas en el merge pendiente del fichero file.txt.)

cm branch showmerges file.txt --format="{date} {name}" --dateformat="ddMMyy"

(Muestra las ramas involucradas en el merge, mostrando solo la fecha y el

nombre, con las fechas en un determinado formato.)

== CMD\_DESCRIPTION\_REPOSITORY ==

Administra repositorios.

== CMD\_USAGE\_REPOSITORY ==

Sintaxis:

cm repository | repo commmand [options]

Comandos:

create | mk

delete | rm

list | ls

rename

add

Para obtener más información sobre cada comando:

cm repository command --usage

cm repository command --help

== CMD\_HELP\_REPOSITORY ==

Ejemplos:

cm repository

cm repository list

cm repository newrepo

cm repository create newrepo

cm repository rename oldname newname

cm repository add C:\repo\

== CMD\_DESCRIPTION\_REPOSITORY\_CREATE ==

Crea un repositorio en un servidor.

== CMD\_USAGE\_REPOSITORY\_CREATE ==

Sintaxis:

cm repository rep\_name

cm repository servidor <nombre\_repositorio>+

cm repository [create | mk] rep\_name

servidor Especificación del servidor.

Usa 'cm help objectspec' para más información sobre

especificaciones.

nombre\_repositorio Nombre del nuevo repositorio.

== CMD\_HELP\_REPOSITORY\_CREATE ==

Ejemplos:

cm repository MyRep

cm repo 192.168.1.140:8087 Rep01 Rep01/ModuleA Rep01/ModuleB

cm repo create Rep01

cm repo mk list

== CMD\_DESCRIPTION\_REPOSITORY\_DELETE ==

Borra un repositorio del servidor.

== CMD\_USAGE\_REPOSITORY\_DELETE ==

Sintaxis:

cm repository delete | rm repositoryspec

repositoryspec Especificación del repositorio a borrar.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de repositorios.

== CMD\_HELP\_REPOSITORY\_DELETE ==

Nota:

Elimina un repositorio del servidor de repositorios.

No borra los datos del sistema gestor de bases de datos,

simplemente desconecta el repositorio del sistema de modo

que ya no sea accesible (los datos pueden ser añadidos de

nuevo, véase 'cm repository add').

Ejemplos:

cm repository delete myrepository@repserver:myserver:8084

cm repository rm myrepository@myserver:8084

cm repo rm myrepository

== CMD\_DESCRIPTION\_REPOSITORY\_LIST ==

Lista los repositorios de un servidor.

== CMD\_USAGE\_REPOSITORY\_LIST ==

Sintaxis:

cm repository

cm repository list [servidor] [--format=str\_format]

servidor Especificación del servidor.

Usa 'cm help objectspec' para más información sobre

especificaciones.

--format Devuelve la salida en el formato especificado.

== CMD\_HELP\_REPOSITORY\_LIST ==

Notas:

Este comando acepta una cadena de texto formateada para mostrar la

salida.

Los parámetros de salida del comando son los siguientes:

{repid} | {0} Identificador del repositorio.

{repname} | {1} Nombre del repositorio.

{repserver} | {2} Nombre del servidor del repositorio.

{repowner} | {3} Propietario del repositorio.

{repguid} | {4} Identificador único del repositorio.

{tab} Inserta un tabulador.

{newline} Inserta una nueva línea.

Si el valor del formato es "TABLE", la salida se imprimirá en formato

tabla con los siguientes campos: {repid}, {repname} y {repserver}.

Ejemplos:

cm repository

(Lista todos los repositorios.)

cm repository list localhost:8084 --format="{1, -20} {3}"

(Escribe el campo "nombre del repositorio" utilizando 20 espacios,

alineado a la izquierda, un espacio, y a continuación el propietario.)

cm repository ls localhost:8084 --format="{repname, -20} {repowner}"

(Escribe lo mismo que el ejemplo anterior.)

cm repo ls localhost:8084 --format=TABLE

(Escribe la lista de repositorios en formato tabla con los siguientes

campos: el id del repositorio, el nombre del repositorio, y el nombre

del servidor de repositorios.)

== CMD\_DESCRIPTION\_REPOSITORY\_RENAME ==

Renombra un repositorio.

== CMD\_USAGE\_REPOSITORY\_RENAME ==

Sintaxis:

cm repository rename [spec\_repositorio] nuevo\_nombre

spec\_repositorio Especificación del repositorio a renombrar.

Usa 'cm help objectspec' para obtener más

información sobre la especificación de repositorios.

nuevo\_nombre Nuevo nombre para el repositorio.

== CMD\_HELP\_REPOSITORY\_RENAME ==

Notas:

Este comando renombre un repositorio.

Si no se proporciona ninguna especificación de repositorio, se tomará el

repositorio actual como spec\_repositorio.

Ejemplos:

cm repository rename desarrollo

(El repositorio actual se renombra a 'desarrollo'.)

cm repo rename rep:default@SERVER:8084 desarrollo

(El repositorio 'default' del servidor 'SERVER' se renombra a 'desarrollo'.)

== CMD\_DESCRIPTION\_REPOSITORY\_ADD ==

Añade una base de datos de repositorio al sistema.

== CMD\_USAGE\_REPOSITORY\_ADD ==

Uso:

cm repository add databasename nombre repserverspec

databasename El nombre de la base de datos a reconectar.

nombre Nombre del repositorio.

repserverspec Especificación del servidor de repositorios.

Usa 'cm help objectspec' para obtener más

información sobre la especificación de servidores.

== CMD\_HELP\_REPOSITORY\_ADD ==

Nota:

Reconecta una base de datos de repositorio al servidor. Puede

usarse para mover un repositorio de un servidor a otro o para

restaurar datos.

Ejemplos:

cm repository add rep\_27 myrepository myserver:8084

== CMD\_DESCRIPTION\_TRIGGER\_CREATE ==

Crea un nuevo trigger en un determinado servidor.

== CMD\_USAGE\_TRIGGER\_CREATE ==

Sintaxis:

cm trigger create subtipo-tipo nombre script [opciones]

subtipo-tipo: momento de ejecución del trigger y operacion del trigger.

Teclee 'cm showtriggertypes' para ver una lista completa de los tipos de

trigger disponibles. Algunos ejemplos son:

before-mkbranch

after-mkwokspace

after-setselector

nombre: nombre que queremos dar al trigger.

script: ruta del servidor en la que se encuentra el script

que deseamos ejecutar. Si esta ruta comienza por "webtrigger ", se

considerará como web trigger. Consulte la ayuda del comando para obtener

detalles adicionales.

Opciones:

--position=posicion: Intenta crear el trigger en la posición especificada

si esta no esta cogida por otro del mismo tipo.

--filter=filtro: Comprueba sólo aquellos elementos que coincidan con el

filtro de busqueda indicado. Se pueden especificar varios filtros

separados por comas, si es así el el trigger se ejecutará si el elemento

coincide con cualquiera de los filtros. Si uno de los filtros comienza

con "rep:", el trigger se ejecutara sólo para las operaciones realizadas

sobre el repositorio con el nombre especificado (ver ejemplo).

--server=servidor: Crea el trigger en el servidor especificado. Si no se

especifica ninguno se crea en el servidor configurado en el cliente.

== CMD\_HELP\_TRIGGER\_CREATE ==

Notas:

Web triggers: Para crear web triggers, escriba "webtrigger <target-uri>"

como comando de trigger. En este caso, el trigger ejecutará una consulta

POST contra el URI especificado, en la que el cuerpo de la petición

contiene un diccionario JSON con las variables de entorno del trigger.

También estará presente una clave INPUT apuntando a un vector de cadenas de

texto para aquellos triggers que escriban en la entrada estándar de sus

comandos asociados.

Ejemplos:

cm trigger create after-setselector "backup" "/path/al/script" --position=4

cm tr mk before-mklabel new "/path/al/script" --server=myserver:8084

cm tr mk after-mklabel Log "/path/al/script" --filter="rep:myRep,LB\*"

Este trigger se ejecutará sólo si el nombre de la etiqueta creada

comienza por 'LB' y si está siendo creada en el repositorio 'myrep'.

cm tr mk after-checkin Notify "webtrigger http://myserver.org/endpoint"

== CMD\_DESCRIPTION\_MOVE ==

Mueve o renombra un ítem.

== CMD\_USAGE\_MOVE ==

Sintaxis:

cm move ruta\_origen ruta\_destino

ruta\_origen: Ruta origen del ítem.

ruta\_destino: Ruta destino del ítem.

== CMD\_HELP\_MOVE ==

Notas:

Este comando mueve o renombra un ítem en el repositorio.

Los cambios también se realizan en el sistema de ficheros local.

Si la ruta de origen es un fichero, el destino puede ser un fichero o un

directorio. En el primer caso, el fichero se renombra, si no, el ítem se

mueve al directorio destino.

Si la ruta de origen es un directorio, la ruta destino ha de serlo también.

Requisitos para mover o renombrar un ítem:

El ítem ha de estar controlado por Plastic SCM.

El directorio padre del ítem origen y del ítem destino han de estar desprotegidos.

Ejemplos:

cm move fichero.txt fichero.old (renombra el ítem)

cm move .\fichero.old .\antiguos (mueve fichero.old al directorio antiguos)

cm move .\src .\src2 (renombra el directorio)

== CMD\_DESCRIPTION\_LABEL ==

Este comando permite al usuario efectuar operaciones con etiquetas.

== CMD\_USAGE\_LABEL ==

Sintaxis:

cm label | lb command [options]

Comandos:

create | mk

delete | rm

rename

Para obtener más información sobre cada uno de los comandos ejecute:

cm label command --usage

cm label command --help

== CMD\_HELP\_LABEL ==

Ejemplos

cm label myNewLabel cs:42

(el comando 'create' es opcional)

cm label rename myNewLabel newLabelName

cm label delete newLabelName

== CMD\_DESCRIPTION\_LABEL\_CREATE ==

Crea una etiqueta nueva o etiqueta una changeset dado.

== CMD\_USAGE\_LABEL\_CREATE ==

Sintaxis:

cm label [create] spec\_etiqueta [spec\_changeset | ruta]

[--allxlinkedrepositories]

[-c=str\_comment | -commentsfile=<comments\_file>]

spec\_etiqueta: Especificación de la etiqueta. Usa 'cm help objectspec' para

obtener más información sobre especificaciones.

spec\_changeset: Nombre o especificación del changeset a etiquetar. Usa

'cm help objectspec' para obtener más información sobre especificaciones.

ruta: Ruta del workspace a etiquetar

Si no se especifica spec\_changeset ni ruta se asumirá el directorio actual

como workspace a etiquetar.

Opciones:

--allxlinkedrepositories (-s): Crea la etiqueta en todos los repositorios

enlazados.

-c: Añade un comentario a la etiqueta creada.

--commentsfile=comment.txt: Carga el comentario del fichero especificado.

== CMD\_HELP\_LABEL\_CREATE ==

Notas:

Si existe una variable de entorno llamada PLASTICEDITOR apuntando al

ejecutable de un editor de texto el editor se abrirá automáticamente

para que especifique un comentario.

Ejemplos:

cm label create lb:BL001 cs:1203 -commentsfile=commentlb001.txt

cm label BL002 cs:=1203 -c="first release"

== CMD\_DESCRIPTION\_LABEL\_DELETE ==

Borra una o más etiquetas.

== CMD\_USAGE\_LABEL\_DELETE ==

Sintaxis:

cm label delete spec\_etiqueta1 spec\_etiqueta2 ...

spec\_etiqueta: Etiquetas a borrar, separadas por espacios.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de etiquetas.

== CMD\_HELP\_LABEL\_DELETE ==

Notas:

Este comando borra una o más etiquetas creadas previamente.

Ejemplos:

cm label delete lb:BL001 (Se borra la etiqueta 'BL001')

cm label delete lb:BL001 lb:BL002

== CMD\_DESCRIPTION\_LABEL\_RENAME ==

Renombra una etiqueta.

== CMD\_USAGE\_LABEL\_RENAME ==

Sintaxis:

cm label rename spec\_etiqueta nuevo\_nombre

sepec\_etiqueta: Especificación de la etiqueta que se va a renombrar.

Usa 'cm help objectspec' para obtener más información sobre

la especificación de etiquetas.

nuevo\_nombre: Nuevo nombre para la etiqueta.

== CMD\_HELP\_LABEL\_RENAME ==

Notas:

Este comando renombra una etiqueta.

Ejemplos:

cm label rename lb:BL001 BL002

(La etiqueta 'BL001' se renombra a 'BL002')

== CMD\_DESCRIPTION\_OBLITERATE ==

DEPRECATED

== CMD\_USAGE\_OBLITERATE ==

== CMD\_HELP\_OBLITERATE ==

== CMD\_DESCRIPTION\_OBJECTSPEC ==

Explica cómo escribir object specs (especificaciones de objetos).

== CMD\_USAGE\_OBJECTSPEC ==

Sintaxis:

cm objectspec

Para obtener toda la información sobre cómo construir specs de objetos.

== CMD\_HELP\_OBJECTSPEC ==

Muchos comandos de Plastic SCM esperan 'object specs' como entrada para

referirse a un objeto concreto (normalmente una rama, un changeset, un

repositorio, etc.).

Esta documentación describe los diferentes tipos de "specs" disponibles y

cómo construirlas.

Cada tipo de especificación comienza con un tag único, por ejemplo "rep:" o "cs:".

El tag se debe indicar para comandos que reciban especificaciones generales

de objetos, como por ejemplo "cm setowner object\_spec", pero a menudo se pueden

omitir en comandos que solo acepten un tipo de especificación, como por ejemplo

"cm getfile revision\_spec".

== Repository server spec (repserverspec) ==

repserver:name:port

Ejemplos:

cm lrep repserver:skull:8084

cm lrep skull:8084

Nota:

Lo llamamos "rep" server spec (con el 'rep' delante) en lugar de

simplemente 'server spec' por motivos históricos. Hace mucho tiempo

Plastic tenía servidores de workspaces y repositorios separados, y

el nombre ha sobrevivido.

== Repository spec (repspec) ==

rep:rep\_name@[repserverspec]

Ejemplos:

cm showowner rep:codice@localhost:6060

(Aquí la "rep:" es necesaria porque el comando showowner admite

todo tipo de objetos).

== Branch spec (brspec) ==

br:[/]br\_name[@repspec]

Ejemplos:

cm switch br:/main@rep:plastic@repserver:skull:9095

(En este caso "br:", "rep:" y "repserver:" no son necesarios así

que el comando se puede escribir de forma mucho más compacta:

"cm switch main@plastic@skull:9095" .)

cm find revisions "where branch='br:/main/task001'"

Nota:

La barra inicial '/' en las ramas no es obligatoria. Solíamos escribir

todas las ramas como /main, /main/task001, etc. Pero ahora preferimos

user formatos más cortos tipo main, main/task001.

== Changeset spec (csetspec) ==

cs:cs\_number|cs\_guid[@repspec]

Se puede especificar el número o el GUID del changeset.

Ejemplos:

cm ls /code --tree=ae1390ed-7ce9-4ec3-a155-e5a61de0dc77@code@skull:7070

== Label spec (labelspec) ==

lb:lb\_name[@repspec]

Ejemplos:

cm switch lb:RELEASE2.0

cm switch lb:RELEASE1.4@myrep@MYSERVER:8084

== Revision spec ==

Hay distintos tipos de specs de revisiones:

rev:item\_path[#(brspec|csetspec|labelspec)]

rev:serverpath:item\_path#(brspec|cset\_spec|lb\_spec)

rev:revid:rev\_id[@rep\_spec]

rev:itemid:item\_id#(br\_spec|cset\_spec|lb\_spec)

Ejemplos:

cm diff rev:readme.txt#cs:19 rev:readme.txt#cs:20

cm diff serverpath:/doc/readme.txt#cs:19@myrepo \

serverpath:/doc/readme.txt#br:/main@myrepo@localhost:8084

cm cat revid:1230@rep:myrep@repserver:myserver:8084

== Item spec ==

Se usa muy raras veces.

item:path

Ejemplo:

cm find revision "where item='item:.'"

== Attribute spec ==

att:att\_name[@repspec]

Ejemplo:

cm rmattributereal att:merged@code@doe:8084 cs:25@code@doe:8084

== Workspace specs ==

wk:name@clientmachine

Se usa muy poco porque solamente aplica a los comandos relacionados con los

workspaces. Es útil para indicar el nombre del workspace en lugar del path.

Ejemplo:

cm showselector wk:codebase@modok

Nota:

Estas specs vienen de los viejos tiempos de Plastic SCM 2.x en los que

los 'workspace servers' eran una forma de almacenar metadatos de forma

centralizada. Se eliminaron por cuestiones de rendimiento.

== CMD\_DESCRIPTION\_PARTIAL ==

Ejecuta comandos en un espacio de trabajo parcial.

== CMD\_USAGE\_PARTIAL ==

Sintaxis:

cm partial comando [opciones]

Comandos:

configure

add

undo

co | checkout

unco | undocheckout

ci | checkin

mv | move

rm | remove

stb | switch

upd | update

Para obtener más información de cada uno de los comandos ejecute:

cm partial comando --usage

cm partial comando --help

== CMD\_HELP\_PARTIAL ==

Ejemplos:

cm partial configure +/fondo-azul.png

cm partial update paisaje-1024.png

cm partial checkin ojos-verde.png ojos-negro.png

== CMD\_DESCRIPTION\_PARTIAL\_ADD ==

Añade un ítem al repositorio.

== CMD\_USAGE\_PARTIAL\_ADD ==

Sintaxis:

cm partial add [-R | -r | --recursive] [--silent] [--parents] [--ignorefailed]

[--skipcontentcheck] <rutas>+

rutas Rutas de los ítems a añadir separados por espacios.

Usa comillas (") para especificar rutas con espacios.

Usa \* para añadir todo el contenido del directorio

actual.

Opciones:

-R Añade los ítems recursivamente.

--silent No muestra ninguna salida.

--parents Incluye los directorios padre de los ítems indicados en

la operación.

--ignorefailed Si un ítem no se puede añadir, la operación de add

continuará sin él. Importante: si un directorio no se

puede añadir, su contenido tampoco se añadirá.

--skipcontentcheck Cuando la extensión no es suficiente para determinar si

el fichero es de tipo texto o binario, se asumirá que es

binario en lugar de usar el contenido para detectar el

tipo.

== CMD\_HELP\_PARTIAL\_ADD ==

Notas:

Requisitos para añadir ítems:

- El directorio padre del ítem a añadir debe existir en el repositorio.

Ejemplos:

cm partial add pic1.png pic2.png

(Añade los ítems pic1.png y pic2.png.)

cm partial add c:\workspace\imagen.png

(Añade el ítem imagen.png en el directorio c:\workspace.)

cm partial add -R c:\workspace\src

(Añade recursivamente todos los ítems del directorio especificado.)

cm partial add --parents ejemplos\diseño01.png

(Añade el ítem diseño01.png y su directorio padre ejemplos.)

cm partial add -R \*

(Recursively adds all the contents of the current directory.)

== CMD\_DESCRIPTION\_PARTIAL\_CHECKIN ==

Crea una nueva revisión de un ítem.

== CMD\_USAGE\_PARTIAL\_CHECKIN ==

Sintaxis:

cm partial checkin | ci [opciones] [<ruta>+]

ruta Rutas de los ítems a proteger, separados por espacios.

Usa comillas (") para especificar rutas con espacios.

Usa . para aplicar la operación de checkin al directorio

actual.

Opciones:

-c Añade un comentario al changeset creado en la operación

de protección.

-commentsfile Añade el comentario existente en el fichero especificado

añ changeset creado en la operación de protección.

--all | -a Los ítems cambiados, movidos y borrados localmente en

las rutas especificadas también son protegidos.

--applychanged Protege los ítems cambiados localmente junto con los

ítems desprotegidos.

--keeplock Mantiene el bloqueo de aquellos elementos que estaban

bloqueados después de la operación de protección.

--symlink Realizar la operación de checkin sobre el fichero de

link y no al que apunta.

--ignorefailed Los cambios locales que no se pueden aplicar (porque el

bloqueo (checkout exclusivo) no se puede obtener o

porque están en conflicto con los cambios del servidor)

son descartados y el checkin continúa sin ellos.

== CMD\_HELP\_PARTIAL\_CHECKIN ==

Notas:

- Si no se especifican [opciones] ni [rutas], la operación involucrará a

todos los cambios pendientes en el espacio de trabajo.

- La operación de protección se aplica siempre recursivamente desde la

ruta indicada.

- Requisitos para proteger un ítem:

- El ítem debe estar controlado por Plastic SCM.

- El ítem ha de estar desprotegido.

- Si un ítem ha sido cambiado pero no desprotegido, no es necesario

especificar la opción --applychanged, salvo que en las rutas a ser

protegidas se incluyan directorios o rutas con wildcards ('\*').

La revisión ha de tener cambios. Es decir, no se pueden proteger los

mismos datos.

Si existe una variable de entorno llamada PLASTICEDITOR apuntando al

ejecutable de un editor de texto, y no se especifica un comentario al

hacer checkin, el editor se abrirá automáticamente para que especifique

un comentario.

Ejemplos:

cm partial checkin figura.png paisaje.png

(Protege los ítems figura.png and paisaje.png.)

cm partial checkin . -commentsfile=micomentario.txt

(Protege los ítems de la ruta actual y aplica el comentario existente en el

fichero micomentario.txt.)

cm partial ci fondo.png -c="Nuevo fondo añadido" --keeplock

(Protege el ítem fondo.png, incluye el comentario especificado y mantiene el

bloqueo.)

cm partial checkin --applychanged

(Protege todos los cambios pendientes en el espacio de trabajo.)

cm partial checkin link --symlink

(Proteger el fichero de link y no al que apunta, opción disponible en UNIX)

== CMD\_DESCRIPTION\_PARTIAL\_CHECKOUT ==

Desprotege los ítems dejándolos preparados para ser modificados.

== CMD\_USAGE\_PARTIAL\_CHECKOUT ==

Síntaxis:

cm partial checkout | co [<ruta>+] [--resultformat=str\_format]

[--silent] [--ignorefailed]

Opciones:

ruta Rutas de los ítems a desproteger, separados por

espacios.

Usa comillas (") para especificar rutas con espacios.

Usa . para aplicar la operación de checkout al

directorio actual.

--resultformat Devuelve la salida en el formato especificado.

--silent No muestra ninguna salida.

--ignorefailed Si un ítem no se puede desproteger/bloquear (no se puede

hacer un checkout exclusivo), la operación de checkout

continuará sin él.

== CMD\_HELP\_PARTIAL\_CHECKOUT ==

Notas:

Para desproteger un ítem:

- El ítem debe estar controlado por Plastic.

- El ítem debe estar protegido.

Si se han configurado bloqueos en el servidor (existe el fichero lock.conf),

entonces cada vez que se produzca una desprotección, Plastic comprobará si

se aplica alguna de las reglas. Si es así, la ruta permanecerá en checkout

exclusivo (bloqueada), con lo que nadie podrá desprotegerlo simultáneamente.

Puedes obtener todos los bloqueos del servidor usando 'cm listlocks'.

Lee la documentación Administrator Guide para más información.

Examples:

cm partial checkout imagen1.png imagen2.png

(Desprotege los ítems imagen1.png and imagen2.png.)

cm partial co \*.png

(Desprotege todos los ítems png.)

cm partial checkout .

(Desprotege el directorio actual.)

cm partial checkout -R c:\workspace\paisaje

(Desprotege recursivamente el directorio paisaje.)

cm partial co diseño01.png --exclusive

(Desprotege el ítem exclusivamente.)

== CMD\_DESCRIPTION\_PARTIAL\_CONFIGURE ==

Permite configurar tu espacio de trabajo, cargando o descargando elementos.

== CMD\_USAGE\_PARTIAL\_CONFIGURE ==

Sintaxis:

cm partial configure <+|-ruta>+ [--silent] [--ignorefailed] [--ignorecase]

[--restorefulldirs]

ruta Rutas a cargar o descargar, separadas por espacios.

Usa comillas (") para especificar rutas con espacios.

Todas las rutas deben comenzar con "/".

Opciones:

--silent No muestra ninguna salida.

--ignorefailed Ignora cualquier error durante la operación.

--ignorecase Ignora mayúsculas y minúsculas. Con esta opción el

comando funcionará para "/Data/Textures" incluso si el

usuario indica "/data/teXtures".

--restorefulldirs Resetea una configuración de directorio incorrecta

(ocurre cuando una operación no parcial es ejecutada en

un espacio de trabajo parcial).

Los directorios de esta lista se configuran por

completo (full check), lo que significa que bajarán

automáticamente el nuevo contenido durante el update.

Esta operación no descarga ningún fichero, simplemente

restaura la configuración de directorio en un espacio

de trabajo parcial.

== CMD\_HELP\_PARTIAL\_CONFIGURE ==

Notas:

El comando siempre asume recursividad.

Ejemplos:

cm partial configure +/paisaje-gris.png

(Carga paisaje-gris.png.)

cm partial configure -/paisaje-negro.png

(Descarga paisaje-negro.)

cm partial configure +/textura -/textura/textura01.png

(Carga todo el contenido del directorio textura excepto textura01.png.)

cm partial configure -/

(Descarga todo el directorio de trabajo.)

cm partial configure -/ +/

(Carga todo el directorio de trabajo.)

cm partial configure +/figura-64.png --ignorefailed

(Carga figura-64.png, estuviera o no ya cargado previamente.)

cm partial configure +/ --restorefulldirs

(Fija todos los directorios para que descarguen el nuevo contenido automáticamente.)

cm partial configure +/src/lib --restorefulldirs

(Fija sólo /src/lib y sus subdirectorios para que descarguen el nuevo contenido automáticamente.)

== CMD\_DESCRIPTION\_PARTIAL\_MOVE ==

Mueve o renombra un archivo o directorio.

== CMD\_USAGE\_PARTIAL\_MOVE ==

Sintaxis:

cm partial move | mv src\_path dst\_path [--format=str\_format]

src\_path Ruta origen del ítem.

dst\_path Ruta destino del ítem.

Options:

--format Devuelve la salida en el formato especificado.

== CMD\_HELP\_PARTIAL\_MOVE ==

Notas:

Este comando mueve o renombra un ítem en el repositorio.

Los cambios también se aplican localmente.

Si la ruta de origen es un fichero, el destino puede ser un fichero o un

directorio. En el primer caso, el fichero se renombra. En el segundo, el

fichero se renombra.

Si la ruta de origen en un directorio, el destino ha de serlo también.

El ítem a mover o renombrar debe existir.

Formato:

{0} Ruta de origen.

{1} Ruta de destino.

Examples:

cm partial move fichero.png fichero-azul.png

(Renombra el ítem.)

cm partial mv .\fichero-azul.png .\azulFicheros

(Mueve fichero-azul.png a azulFicheros.)

cm partial move .\diseño .\marketing

(Renombra un directorio.)

== CMD\_DESCRIPTION\_PARTIAL\_RM ==

Borra un ítem del control de versiones.

== CMD\_USAGE\_PARTIAL\_RM ==

Sintaxis:

cm partial remove | rm <ruta>+ [--nodisk]

ruta Ítems a borrar, separados por espacios.

Usa comillas (") para especificar rutas con espacios.

Opciones:

--nodisk Borra del control de versiones, pero mantiene el ítem en

disco.

== CMD\_HELP\_PARTIAL\_RM ==

Notas:

Los ítems se borran de disco. Los ítems borrados son borrados del directorio

padre en el control de versiones.

Requisitos:

- El ítem debe estar controlado.

Ejemplos:

cm partial remove src

(Borra src. Si src es un directorio, equivale a ejecutar:

cm partial remove -R src.)

cm partial remove c:\workspace\pic01.png --nodisk

(Borra pic01.png del control de versiones, pero lo mantiene en el disco.)

== CMD\_DESCRIPTION\_PARTIAL\_SWITCH ==

Establece la rama especificada como rama de trabajo.

== CMD\_USAGE\_PARTIAL\_SWITCH ==

Sintaxis:

cm switch branch\_spec [--report | --silent] [--workspace=path]

(Establece la rama de trabajo y actualiza el workspace.)

cm switch branch\_spec --configure <+|-ruta>+ [--silent]

[--ignorefailed] [--ignorecase] [--workspace=path]

(Establece la rama de trabajo y configura el workspace como haría el

comando 'cm partial configure'.)

branch\_spec Especificación de rama.

ruta Rutas a cargar o descargar, separadas por espacios.

Usa comillas (") para especificar rutas con espacios.

Todas las rutas deben comenzar con "/".

Opciones:

--silent No muestra salida.

--report Imprime una lista de los cambios efectuados una vez

termina la operación. Pierde efecto si se utiliza

la opción --silent conjuntamente.

--configure Configura (carga / descarga rutas) el espacio de

trabajo después de haber actualizado la rama de trabajo.

Usa 'cm partial configure --help' para aprender más

sobre como especificar las rutas a configurar.

--ignorefailed Ignora cualquier error durante la operación

de configuración.

--ignorecase Ignora mayúsculas y minúsculas en las rutas a

configurar. Con esta opción el comando funcionará

para "/Data/Textures" incluso si el usuario indica

"/data/teXtures".

--workspace=path Ruta del espacio de trabajo a cambiar.

--forcedetailedprogress Fuerza mostrar progreso detallado incluso cuando se

redirige la salida estándar.

Usa 'cm help objectspec' para aprender más sobre cómo indicar especificaciones

de ramas.

== CMD\_HELP\_PARTIAL\_SWITCH ==

Notas:

Este comando permite a los usuarios especificar una rama, como rama de

trabajo. Después de actualizar la rama, actualiza el espacio de trabajo

a la nueva rama como haría un 'cm partial update'. Sin embargo, si se

especifica la opción --configure, permite configurar el espacio de trabajo

teniendo en cuenta los elementos cargados en la nueva rama como haría un

'cm partial configure'.

Ejemplos:

cm switch br:/main/task

(Establece /main/task como rama de trabajo y actualiza el espacio de

trabajo.)

cm switch br:/main/task --configure +/art/images

(Establece /main/task como rama de trabajo y configura el directorio

/art/images.)

== CMD\_DESCRIPTION\_PARTIAL\_UNCO ==

Deshace la desprotección de un ítem.

== CMD\_USAGE\_PARTIAL\_UNCO ==

Sintaxis:

cm partial undocheckout | unco <ruta>+ [--silent]

ruta Ítems sobre los que aplicar la operación, separados por

espacios.

Usa comillas (") para especificar rutas con espacios.

Usa . para aplicar la operación al directorio actual.

Opciones:

--silent No muestra ninguna salida.

== CMD\_HELP\_PARTIAL\_UNCO ==

Notas:

Si un ítem está desprotegido y no quieres protegerlo, puede deshacer la

desprotección usando este comando.

Se puede deshacer la desprotección tanto a ficheros como a carpetas.

El ítem se actualizará al estado anterior previo a ser desprotegido.

Requisitos:

- El ítem debe estar controlado.

- El ítem debe estar desprotegido.

Ejemplos:

cm partial undocheckout .

(Deshace las desprotecciones del directorio actual.)

cm partial undocheckout pic1.png pic2.png

cm unco c:\workspace\diseño01.png

(Deshace las desprotecciones de los ítems especificados.)

== CMD\_DESCRIPTION\_PARTIAL\_UNDO ==

Deshace cambios en un workspace.

== CMD\_USAGE\_PARTIAL\_UNDO ==

Syntaxis:

cm partial undo [path]+ [--symlink] [-r | --recursive] [Filtros]+

[Opciones de salida]

path Rutas de los ficheros o carpetas a los que aplicar

la operación, separados por espacios. Se pueden

utilizar comillas (") para especificar rutas con

espacios. Si no se especifica ninguna ruta, por

defecto la operación se ejecutará sobre los ficheros

en el directorio actual.

Opciones:

--symlink Aplica la operación de undo al enlace simbólico y

no a su destino.

-r Ejecuta el undo de manera recursiva.

Filtros:

Si ninguna de estas opciones está presente, por defecto se deshacen todos

los tipos de cambio, pero los ficheros pueden filtrarse utilizando estas

opciones.

Si un fichero o directorio tiene uno o más de los tipos de cambio

especificados, todos los cambios en ese fichero o directorio serán

deshechos.

Por ejemplo, si se especifica tanto --checkedout como --moved, si un fichero

está tanto movido como checkedout, ambos cambios se desharán.

--checkedout Selecciona ficheros y directorios marcados para

cambios.

--unchanged Selecciona ficheros sin modificaciones en su

contenido.

--changed Selecciona ficheros y directorios cambiados o

marcados para cambios.

--deleted Selecciona ficheros y directorios eliminados.

--moved Selecciona ficheros y directorios movidos.

--added Selecciona ficheros y directorios añadidos.

Opciones de salida:

--silent | --machinereadable [--startlineseparator=sep]

[--endlineseparator=sep] [--fieldseparator=sep]

--silent No muestra ninguna salida.

--machinereadable Muestra la salida en un formato fácil de tratar.

--startlineseparator Usado en conjunto con la opción '--machinereadable',

especifica cómo deberían comenzar las líneas.

--endlineseparator Usado en conjunto con la opción '--machinereadable',

especifica cómo deberían terminar las líneas.

--fieldseparator Usado en conjunto con la opción '--machinereadable',

especifica cómo deberían separarse las líneas.

== CMD\_HELP\_PARTIAL\_UNDO ==

Notas:

El comand undo es peligroso - deshace trabajo de una manera irreversible.

Una vez que el undo termina, no hay manera posible de recuperar el estado

anterior de los ficheros y directorios afectados. Si no se especifica

ninguna ruta en los argumentos, por defecto se desharán los cambios en todos

los elementos del directorio actual, pero NO de manera recursiva.

Estos comandos son equivalentes ejecutados desde el directorio /src:

/src

|- file.txt

|- code.cs

\- /test

|- test\_a.py

\- test\_b.py

$ cm partial undo

$ cm partial undo \*

$ cm partial undo file.txt code.cs /test

$ cm partial undo .

$ cm partial undo /src file.txt code.cs

Si se quiere que la operación sea recursiva, se necesita especificar la

opción -r.

Para deshacer todos los cambios debajo de un directorio (incluyendo los

cambios sobre el propio directorio):

$ cm partial undo dirpath -r

Si dirpath es la ruta de un workspace, todos los cambios dentro del

workspace serán deshechos.

Examples:

$ cm partial undo . -r

(Deshace todos los cambios en el directorio actual de manera recursiva. Si

se ejecuta desde la raíz del workspace, deshace todos los cambios en el

workspace completo.)

$ cm partial co file.txt

$ cm partial undo file.txt

(Deshace el checkout en file.txt.)

$ echo content >> file.txt

$ cm partial undo file.txt

(Deshace el cambio local en file.txt.)

$ cm partial undo src

(Deshace los cambios en el directorio src y en todos los ficheros

controlados que contenga)

$ cm partial undo src/\*

(Deshace los cambios en todos los elementos contenidos en src, sin afectar

al propio directorio. Por la expansión del wildcard, es equivalente a

'cm partial undo src/file1.txt src/file2.txt').

$ cm partial undo \*.cs

(Deshace cambios en cada elemento que encaje con el patrón \*.cs en el

directorio actual.)

$ cm partial undo \*.cs -r

(Deshace cambios en cada elemento que encaje con el patrón \*.cs en el

directorio actual, y en cada directorio por debajo de una manera recursiva.)

$ cm partial co file1.txt file2.txt

$ echo content >> file1.txt

$ cm partial undo --unchanged

(Deshace el checkout en file2.txt por no estar modificado, ignorando

file1.txt ya que tiene cambios locales.)

$ echo content >> file1.txt

$ echo content >> file2.txt

$ cm partial co file1.txt

$ cm partial undo --checkedout

(Deshace el cambio en el fichero en checkout file1.txt, ignorando file2.txt

ya que no está en checkout.)

$ cm partial add file.txt

$ cm partial undo file.txt

(Deshace el añadido de file.txt, dejándolo de nuevo como privado.)

$ rm file1.txt

$ echo content >> file2.txt

$ cm partial add file3.txt

$ cm partial undo --deleted --added \*

(Deshace el borrado de file1.txt y el añadido de file3.txt, ignorando

el cambio en file2.txt.)

== CMD\_DESCRIPTION\_PARTIAL\_UPDATE ==

Actualiza el espacio de trabajo parcial.

== CMD\_USAGE\_PARTIAL\_UPDATE ==

Sintaxis:

cm partial update [<ruta>+]

[--changeset=número]

[--silent | --report]

ruta Rutas a actualizar, separadas por espacios. Las dobles

comillas (") deben usarse para especificar rutas que

contienen espacios.

Si se usa . entonces se actualiza el directorio actual.

Si no se especifica ninguna ruta, entonces se actualiza todo

el espacio de trabajo parcial.

Opciones:

--changeset Actualiza el espacio de trabajo parcial al changeset

especificado.

--silent No muestra salida.

--report Imprime una lista de los cambios efectuados una vez que

termina la operación. Esta opción no tiene efecto si se

utiliza la opción --silent.

--dontmerge No procesa los conflictos de fichero. El resto de cambios

se procesan con normalidad. Esta opción puede ser útil en

automatización para evitar interacción con el usuario.

== CMD\_HELP\_PARTIAL\_UPDATE ==

Notas:

El comando update actualiza los ficheros desactualizados.

El comando update siempre asume recursividad.

Si todas las rutas especificadas son ficheros dentro del mismo xlink y

se usa la opción --changeset, entonces las versiones a actualizar se buscan

dentro del changeset del repositorio del xlink.

Ejemplos:

cm partial update

(Actualiza todo el espacio de trabajo parcial.)

cm partial update .

(Actualiza todos los hijos del directorio actual.)

cm partial update fichero1.txt

(Actualiza el fichero fichero1.txt.)

cm partial update fichero1.txt fichero2.txt

(Actualiza los ficheros fichero1.txt y fichero2.txt.)

cm partial update src --report

(Actualiza todos los hijos del directorio src, mostrando una lista de los

cambios aplicados al finalizar.)

cm partial update src --changeset=4

(Actualiza todos los hijos del directorio src, al contenido

que cargaban en el changeset 4.)

cm partial update xlink/first.png --changeset=4

(Actualiza el fichero xlink/first.png al contenido que cargaba en el

changeset 4 del repositorio del xlink.)

== CMD\_DESCRIPTION\_PATCH ==

Genera un parche a partir de una spec o aplica un parche generado en el workspace actual.

== CMD\_USAGE\_PATCH ==

Sintaxis:

cm patch [source\_spec] source\_spec

[--output=output\_file] [--tool=/path/to/diff]

cm patch --apply patch\_file [--tool=/path/to/patch]

source\_spec Especificación de rama o changeset

output\_file Fichero que contendrá la salida del comando. Se

mostrará en pantalla si no se especifica.

Opciones:

--output Indica el fichero de destino para almacenar los

contenidos del parche.

--tool Especifica la ruta hacia la herramienta necesaria para

ejecutar la operación (diff o patch).

== CMD\_HELP\_PATCH ==

Notas:

Este comando genera un fichero patch que contiene las diferencias de una

rama o un changeset, así como las diferencias entre changesets. Encuentra

diferencias para ficheros tanto de texto como binarios.

El parámetro --apply permite aplicar los contenidos de un fichero patch

en el workspace actual.

Limitaciones:

Si el fichero de salida de patch ya existe, el comando no lo

sobreescribirá.

Si el fichero con modificaciones en el patch no existe, el comando no

creará uno nuevo para aplicar los cambios.

Importante:

Este comando requiere las utilidades Diff y Patch, disponibles

en http://gnuwin32.sourceforge.net/packages/patch.htm y

http://gnuwin32.sourceforge.net/packages/diffutils.htm

Tras su instalación se recomienda agregar su ubicación a la variable

de entorno PATH.

Ejemplos:

cm patch cs:4@default@localhost:8084

(Muestra en pantalla las diferencias del changeset 4 en formato patch.)

cm patch br:/main --output=file.patch

(Obtiene un fichero file.patch con las diferencias de la rama "main".)

cm patch br:/main --output=file.patch --tool=C:\gnu\diff.exe

(Misma operación con un ejecutable indicado explícitamente.)

cm patch cs:2@default cs:4@default

(Muestra en pantalla las diferencias entre los changeset 2 y 4.)

cm patch --apply file.patch --tool=C:\gnu\patch.exe

(Aplica el parche contenido en file.patch al workspace actual con un

ejecutable indicado explícitamente.)

== CMD\_DESCRIPTION\_QUERY ==

Ejecuta una consulta SQL contra la base de datos del servidor.

== CMD\_USAGE\_QUERY ==

Uso:

cm query comandosql [--outputfile=ruta]

[--solveuser=nombre\_columna1,nombre\_columna2,...]

[--solvepath=nombre\_columna1,nombre\_columna2,...]

[--columnwidth=valor]

--outputfile: Escribe en un fichero el resultado de la consulta.

--solveuser: Indica al intérprete de consultas que la(s) columna(s)

especificada(s) contiene(n) usuarios. Se tratarán de resolver

identificadores de usuario a nombres de usuario.

--solvepath: Indica al intérprete de consultas que la(s) columna(s)

especificadas(s) contiene(n) identificadores de ítem. Se tratarán de

resolver los identificadores a rutas de disco.

--columnwidth: Especifica la anchura de cada columna del resultado de la

consulta.

== CMD\_HELP\_QUERY ==

Notas:

Este comando permite a los usuarios ejecutar consultas SQL directamente

contra la base de datos del servidor.

A continuación se detalla el esquema (tablas y campos de cada tabla)

disponibles para realizar consultas.

ITEMS

OBJECTID (integer)

CREATIONDATE (datetime)

OWNER (string)

REVISIONS

OBJECTID (integer)

SIZEBYTES (integer)

ITEMID (integer)

BRANCHID (integer)

CHANGESET (integer)

COMMENT (string)

CREATIONDATE (datetime)

OWNER (string)

CHECKOUTS

REVISIONID (integer)

WORKSPACESERVER (string)

CLIENTMACHINE (string)

EXCLUSIVE (char)

CREATIONDATE (datetime)

OWNER (string)

BRANCHES

OBJECTID (integer)

NAME (string)

PARENTBRANCHID (integer)

CREATIONDATE (datetime)

OWNER (string)

LABELS

OBJECTID (integer)

NAME (string)

CREATIONDATE (datetime)

OWNER (string)

LABELEDREVISIONS

LABELID (integer)

REVISIONID (integer)

CREATIONDATE (datetime)

OWNER (string)

LINKS

OBJECTID (integer)

NAME (string)

CREATIONDATE (datetime)

OWNER (string)

LINKEDOBJECTS

LINKID (integer)

SOURCEOBJECTID (integer)

DESTINATIONOBJECTID (integer)

CREATIONDATE (datetime)

OWNER (string)

ATTRIBUTES

OBJECTID (integer)

NAME (string)

CREATIONDATE (datetime)

OWNER (string)

OBJECTSWITHATTRIBUTES

ATTRIBUTEID (integer)

SOURCEOBJECTID (integer)

ATTRIBUTEVALUE (string)

CREATIONDATE (datetime)

OWNER (string)

CHANGESETS

CHANGESETNUMBER (integer)

CREATIONDATE (datetime)

OWNER (string)

Además, a la hora de escribir consultas, existen dos functiones predefinidas

para trabajar con usuarios y rutas de disco.

Estas funciones son las siguientes:

\* SolveUser(nombre\_usuario), resuelve el nombre de usuario dado al formato que

maneja Plastic SCM.

\* SolvePath(path), resuelve rutas de disco a identificadores de ítem itemid.

Para mostrar el resultado de las consultas de forma legible al usuario, se

puede indicar al intérprete de consultas que convierta identificadores de

usuario a nombres de usuario, e identificadores de ítem a rutas de disco.

Para ello se utilizan las opciones:

--solveuser=nombre\_columna

--solvepath=nombre\_columna

Se pueden especificar varios nombres de columnas separados por comas.

Ejemplos:

cm query "SELECT \* FROM revision"

cm query "SELECT b.sname as br\_name, o.dtimestamp as date from branch b, object o, seid s

where b.iobjid=o.iobjid and o.fidowner=s.iseidid and s.scode='SolveUser(juan)'"

cm query "SELECT \* FROM REVISION WHERE itemid=SolvePath(c:\mi\_workspace)"

cm query "SELECT r.iobjid, r.fiditem as ruta, s.scode as nombre\_usuario FROM revision r, object o, seid s

WHERE r.iobjid=o.iobjid and o.fidowner=s.iseidid" --solveuser=nombre\_usuario --solvepath=ruta

== CMD\_DESCRIPTION\_ATTRIBUTE\_DELETE ==

Borra uno o más atributos.

== CMD\_USAGE\_ATTRIBUTE\_DELETE ==

Sintaxis:

cm attribute delete spec\_atributo1 spec\_atributo2 ...

spec\_atributo Atributos a borrar, separados por espacios.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de atributos.

== CMD\_HELP\_ATTRIBUTE\_DELETE ==

Notas:

Este comando borra uno o más atributos creados previamente.

Los parámetros son especificaciones de atributos, dados en el formato

att:nombre.

Ejemplos:

cm attribute delete att:Status

(Borra el atributo 'Status'.)

cm att rm status att:integrated@reptest@server2:8084

(Borrar los atributos 'status' e 'integrated'.)

== CMD\_DESCRIPTION\_ATTRIBUTE\_UNSET ==

Borra un atributo que ha sido fijado previamente en un objeto.

== CMD\_USAGE\_ATTRIBUTE\_UNSET ==

Sintaxis:

cm attribute unset spec\_atributo spec\_objeto

spec\_atributo Especificación del atributo.

spec\_objeto Especificación del objeto sobre el que fijar

el atributo.

Se pueden fijar atributos sobre ramas, changesets,

shelvesets, etiquetas, items y revisiones.

Usa 'cm help objectspec' para obtener más información sobre especificación

de objetos.

== CMD\_HELP\_ATTRIBUTE\_UNSET ==

Notas:

Un atributo que ha sido fijado previamente a un objeto puede ser borrado

con este comando. Se borra la realización del atributo en el objeto pero

no el atributo.

Usa 'cm help objectspec' para aprender cómo indicar especificaciones de

objetos.

Ejemplo:

cm attribute unset att:Status br:/main/SCM105

== CMD\_DESCRIPTION\_ATTRIBUTE\_RENAME ==

Renombra un atributo.

== CMD\_USAGE\_ATTRIBUTE\_RENAME ==

Sintaxis:

cm attribute rename spec\_atributo nuevo\_nombre

spec\_atributo: Especificación del atributo que se va a renombrar.

Usa 'cm help objectspec' para obtener más información sobre

la especificación de atributos.

nuevo\_nombre: Nuevo nombre para el atributo.

== CMD\_HELP\_ATTRIBUTE\_RENAME ==

Notas:

Este comando renombra un atributo.

Ejemplos:

cm attribute rename att:Status state

(El atributo 'Status' se renombra a 'state')

== CMD\_DESCRIPTION\_ATTRIBUTE\_EDIT ==

Editar el comentario de un atributo.

== CMD\_USAGE\_ATTRIBUTE\_EDIT ==

Sintaxis:

cm attribute edit spec\_atributo nuevo\_comentario

spec\_atributo: Especificación del atributo objetivo del cambio.

Usa 'cm help objectspec' para obtener más información sobre

la especificación de atributos.

nuevo\_comentario: Nuevo comentario para el atributo. También puedes

especificar una lista de valores por defecto para el

atributo.

== CMD\_HELP\_ATTRIBUTE\_EDIT ==

Notas:

Este comando cambia el comentario de un atributo.

Para especificar la lista de valores por defecto para un atributo, solo

tienes que incluir un texto como este en el comentario del atributo:

'default: valor\_uno, "valor dos", valor3, "Valor final"'.

Ejemplos:

cm attribute edit att:status "The status of a branch in the CI pipeline."

(Edita el comentario del atributo "status".)

cm attribute edit att:status "Status of a branch. default: open, resolved, reviewed"

(Edita el comentario del atributo "status". Y también especifica una lista

de valores. De este modo, cuando apliques el atribute "status" a un objeto,

podrás seleccionar uno de los siguientes valores: "open", "resolved" o

"reviewed".)

== CMD\_DESCRIPTION\_REPLICATE ==

ATENCIÓN: Este comando se ha declarado obsoleto.

Ha sido reemplazado por 'pull' (equivalente a 'replicate') y por 'push'

(equivalente a 'replicate --push').

Replica datos desde un repositorio remoto.

== CMD\_USAGE\_REPLICATE ==

Uso:

cm replicate br\_spec dst\_rep\_spec [--nodata] [translateOptions] [authOptions]

cm replicate hydrate br\_spec [src\_rep\_spec] [authOptions]

cm replicate hydrate cs\_spec [src\_rep\_spec] [authOptions]

cm replicate br\_spec --package=packagename [--changeset=número]

cm replicate dst\_rep\_spec --import=packagename

br\_spec: la rama para ser replicada/hidratada. Es una especificación

completa de rama con el siguiente formato:

br:/NOMBRE\_RAMA[@rep:NOMBRE\_REP[@repserver:NOMBRE\_SERVIDOR:puerto]]

cs\_spec: changeset para ser hidratado. Es una espec. de changeset:

cs:NUMERO\_CHANGESET[@rep:NOMBRE\_REP[@repserver:NOMBRE\_SERVIDOR:puerto]]

src\_rep\_sec: repositorio origen de los datos. Es una especificación

completa de repositorio:

rep:NOMBRE\_REPOSITORIO@repserver:NOMBRE\_SERVIDOR:puerto

dst\_rep\_sec: repositorio destino de la replica. Es una especificación

completa de repositorio:

rep:NOMBRE\_REPOSITORIO@repserver:NOMBRE\_SERVIDOR:puerto

Opciones:

--package para crear un paquete de replicación.

--import para importar un paquete concreto.

--nodata para replicar los cambios de una rama sin replicar sus datos. Esta

opción no se puede usar para la réplica con paquetes ni con la opción

--push.

Opciones de traducción:

--trmode=[copy | name | table] para especificar cómo tratar los

nombres de usuario cuando se importen los datos en el

repositorio de destino:

\* copy -> es el utilizado por defecto. Copia los SEIDs en el

repositorio de destino.

\* name -> realiza una traducción por nombre entre el modo

de autentificación de origen y el de destino.

\* table -> se usará una tabla de traducción en la que se

especifican los nombres de origen y cómo han de quedar

en el destino.

--trtable=[translationTable]

una tabla de traducción es un fichero que contiene entradas

en la forma nombre antiguo;nombre nuevo.

Opciones de autentificación:

Hay dos formas de especificar los datos de autentificación:

1) Especificando un fichero de autentificación.

--authfile=fichero, un fichero que contiene dos líneas: el modo de

autentificación en la primera línea y los datos de autentificación

(ver --authdata) o una cadena vacía en la segunda.

2) Especificando la autentificación mediante parámetros.

Para ello hay que especificar el modo de autentificación mediante el

modificador:

--authmode=[NameWorkingMode | NameIDWorkingMode |

LDAPWorkingMode | ADWorkingMode | UPWorkingMode]

Si está trabajando en UPWorkingMode o LDAPWorkingMode, puede

especificar los datos de autentificación de dos formas:

2.1) Para UPWorkingMode o LDAPWorkingMode:

--authdata=datos\_de\_autentificación

Ejemplos:

--authdata=ActiveDirectory:192.168.1.3:389:john.doe@factory.com:fPBea2rPsQaagEW3pKNveA==:dc=factory,dc=com (LDAPWorkingMode)

--authdata=john:fPBea2rPsQaagEW3pKNveA== (UPWorkingMode)

2.2) Solamente para UPWorkingMode:

--user=usuario

--password=contraseña

== CMD\_HELP\_REPLICATE ==

Notas:

El comando 'replicate' replica datos desde una rama hacia un

repositorio de destino. Todos los datos de la rama de origen, incluyendo

revisiones, ítems, changesets, etiquetas, revisiones de código, links

y seguridad serán replicados.

Su cliente Plastic SCM debe estar configurado para trabajar con el

repositorio de destino.

La replicación también gestiona la 'reconciliación' de ramas que han

sido modificadas tanto en origen como en destino, mediante la creación

de 'ramas de entrega' (fetch branches) que podrán ser integradas

posteriormente para resolver los conflictos.

La replicación puede funcionar en tres modos diferentes:

1- Comunicación directa entre servidores: lo que significa que el servidor

de destino se comunicará con el de origen para sincronizar la rama

especificada.

2- Generación de paquete de exportación: el cliente se conectará

únicamente con el destino para generar un paquete de replicación con los

datos y los metadatos de la rama especificada. Se usará el modificador

--package. El usuario podrá especificar, opcionalmente, un changeset

inicial desde el cual generar la replicación.

3- Importar un paquete generado previamente, usando la opción --import.

En los modos 1 y 2 será necesaria la autentificación contra el servidor

de origen. El comando 'replicate' acepta parámetros para especificar los

diferentes modos de autentificación.

Ejemplos:

cm replicate br:/main@rep:default@repserver:LONDRES:8084 rep:myrep@repserver:MADRID:9090

cm replicate br:/main@rep:default@repserver:LONDRES:8084 rep:myrep@repserver:MADRID:9090 --trmode=name

cm replicate br:/main@rep:default@repserver:LONDRES:8084 rep:myrep@repserver:MADRID:9090 --trmode=table --trtable=tabla.txt

cm replicate br:/main@rep:default@repserver:LONDRES:8084 rep:myrep@repserver:MADRID:9090 --authmode=NameWorkingMode --user=john.doe

Significa que se autentificará contra el servidor 'MADRID' usando NameWorkingMode como usuario 'john.doe'

cm replicate br:/main/releaseBL060@rep:blackbird@repserver:barcelona:9090 --package=replicationpackage.data

Creará un paquete de replicación

cm replicate rep:mine@repserver:casa:9094 --import=replicationpackage.data

Importará el paquete creado con anterioridad

cm replicate br:/main/releaseBL060@rep:blackbird@repserver:barcelona:9090 --package=replicationpackage.data --changeset=1230

Generará un paquete de replicación de la rama br:/main/releaseBL060 desde el changeset 1230

cm replicate /main@project1@LONDRES:8084 projectx@localhost:8084 --nodata

Replicará la rama /main del servidor LONDRES a mi servidor sin replica los datos.

cm replicate hydrate /main@projectx@localhost:8084 project1@LONDRES:8084

Hidratará los datos de la rama /main de mi servidor, usando los datos del servidor LONDRES.

Nota adicional:

La tabla de traducción tiene entradas (una por línea) con el siguiente

formato:

nombre antiguo;nombre nuevo

== CMD\_DESCRIPTION\_PULL ==

Replica datos desde un repositorio remoto.

== CMD\_USAGE\_PULL ==

Uso:

cm pull src\_br\_spec dst\_rep\_spec

[--preview] [--nodata] [translateOptions]

[--user=usr\_name [--password=pwd] | AuthOptions]

(Replicación directa entre servidores. Hace pull de una rama desde un

repositorio.)

cm pull dst\_rep\_spec --package=pack\_file [AuthOptions]

(Replicación basada en paquetes. Importa el paquete en el servidor de

destino.)

cm pull hydrate dst\_br\_spec [src\_rep\_spec]

[--user=usr\_name [--password=pwd] | AuthOptions]

(Introduce los datos restantes para todos los changesets de una rama

replicada previamente con la opción --nodata. Si no se especifica un

repositorio del que obtener los datos, Plastic intentará usar el origen

de replicación original de la rama.)

cm pull hydrate dst\_cs\_spec [src\_rep\_spec]

[--user=usr\_name [--password=pwd] | AuthOptions]

(Introduce los datos restantes para un changeset replicado previamente

con la opción --nodata. Si no se especifica un repositorio del que obtener

los datos, Plastic intentará usar el origen de replicación original de

la rama.)

src\_br\_spec La rama a ser replicada desde un repositorio remoto.

dst\_br\_spec La rama a ser hidratada.

Especificación de ramas:

[br:][/]nombre\_rama[@[rep:]nombre\_rep[@[repserver:]nombre\_servidor:puerto]]

dst\_cs\_spec El changeset a ser hidratado.

Especificación de changesets:

cs:numero\_cset[@[rep:]nombre\_rep[@[repserver:]nombre\_servidor:puerto]]

dst\_rep\_spec El repositorio de destino.

Especificación de repositorio:

[rep:]nombre\_rep[@[repserver:]nombre\_servidor:puerto]

--package El fichero de donde se importa un paquete de replicación

creado anteriormente.

Es útil para mover datos entre servidores sin una

conexión de red directa.

Consule 'cm push' para crear paquetes de replicación.

Opciones:

--preview Proporciona información sobre qué cambios serán traídos

durante el pull, pero no se ejecuta ninguna acción más.

Esta opción es útil para comprobar qué datos serán

replicados antes de ejecutar el pull.

--user, --password Credenciales para utilizar en caso de que el modo de

autenticación de los servidores origen y destino

difieran, y no exista un profile para autenticarse

contra el remoto.

--nodata Replica los metadatos de una rama sin replicar los

datos. Esta opción no se puede utilizar al importar un

paquete de replicación.

Opciones de traducción (translateOptions):

--trmode={copy|name|table}

El servidor de origen y de destino pueden utilizar modos distintos de

autenticación. Este argumento especifica cómo se han de traducir los

nombres de usuario del origen al destino.

- copy El modo por defecto, indica que los nombres de usuario serán

simplemente copiados.

- name Los nombres de usuario serán emparejados entre origen y

destino por coincidencia.

- table Se utilizará una tabla de traducción (más información a

continuación).

--trtable={translation\_table\_file}

Si el modo de traducción es 'table', entonces una tabla de traducción

es un fichero que contiene líneas de la forma nombreantiguo;nombrenuevo.

Cuando la rama se escribe en el repositorio de destino, los objetos

creados por un usuario identificado por "nombreantiguo" en el

repositorio de origen serán asignados al usuario identificado por

"nombrenuevo" en el repositorio de destino.

Opciones de autentificación (AuthOptions):

Hay dos formas de especificar los datos de autentificación:

1) Especificando la autentificación mediante parámetros:

--authmode={NameWorkingMode|LDAPWorkingMode|ADWorkingMode|UPWorkingMode}

(LDAPWorkingMode) --authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(UPWorkingMode) --authdata=dave:fPBea2rPsQaagEW3pKNveA==

El parámetro '--authdata' es el contenido del campo <SecurityConfig>

en los ficheros client.conf o profiles.conf. El fichero profiles.conf

puede ser generado desde la GUI de Plastic SCM (Windows).

Si se utiliza UPWorkingMode, se puede especificar simplemente:

--authmode=UPWorkingMode --user=user --password=pwd

2.1) Para UPWorkingMode o LDAPWorkingMode:

--authdata=datos\_de\_autentificación

Ejemplos:

--authdata=ActiveDirectory:192.168.1.3:389:john.doe@factory.com:fPBea2rPsQaagEW3pKNveA==:dc=factory,dc=com (LDAPWorkingMode)

--authdata=john:fPBea2rPsQaagEW3pKNveA== (UPWorkingMode)

2.2) Solamente para UPWorkingMode:

--user=usuario

--password=contraseña

1) Especificando un fichero de autentificación. Se pueden tener varios

ficheros de autenticación, uno por servidor al que se conecte,

conteniendo las credenciales específicas para dicho servidor.

--authfile=fichero

Contiene 2 líneas:

Línea 1) modo de autenticación, igual que para --authmode.

Línea 2) datos de autenticación, igual que para --authdata.

== CMD\_HELP\_PULL ==

Notas:

El comando pull replica ramas (junto a sus changesets) entre un repositorio

origen y otro destino. Los repositorios pueden encontrarse en diferentes

servidores.

Hay dos operaciones de replicación: push y pull.

Una operación de 'pull' significa que la operación de replicación hará que

el servidor destino obtenga datos desde el origen. El cliente se conectará

con el servidor destino, y, desde ese host, establecerá una conexión con el

repositorio origen para recuperar los datos pedidos. Durante el pull, será

el servidor destino el que se conecte al origen.

Aunque en un escenario distribuido típico el desarrollador haga pùsh de los

cambios en su servidor local al servidor central, también puede ser

necesario hacer pull de los últimos cambios en el servidor central al

servidor local.

La replicación puede resolver situaciones en las que se hagan cambios

concurrentes en la misma rama en dos repositorios replicados:

- Push: si se intenta hacer push de datos locales a un repositorio que tenga

datos más nuevos que los que se están enviando, el sistema pedirá hacer

pull de los últimos cambios, resolver el merge resultante y, finalmente,

tratar de repetir el push.

- Pull: cuando se hace pull de changesets de una rama remota, estos estarán

correctamente enlazados a sus changesets padre. Si el changeset que se ha

traído no es un hijo del último changeset de la rama, entonces aparecerá

un escenario de múltiples cabezas. La rama tendrá más de una 'cabeza', o

dicho de otro modo, más de un changeset final de rama. Será necesario

hacer merge de ambas cabezas antes de que se pueda hacer push de nuevo.

El pull puede funcionar de dos modos:

1) Comunicación directa entre servidores: lo que significa que el servidor

de destino se comunicará con el de origen para sincronizar la rama

especificada.

2) Importación de un paquete de replicación generado con push, con la opción

--package.

El modo 1) requere que el usuario que ejecute el comando esté autenticado

contra el servidor remoto, bien o usando la autenticación por defecto en

el fichero client.conf, mediante un profile creado previamente, o

especificando los argumentos --authmode y --authdata (o --user y --password

en caso de que el modo de autenticación sea UPWorkingMode).

El modo 2) require usar un paquete de replicación generado previamente con

el comando push.

Recuerde que la replicación pull funciona de una manera indirecta. Cuando

se ejecuta, el comando pide al repositorio destino que se conecte con el

origen para obtener la rama especificada.

Sin embargo, esto se puede hacer de manera directa utilizando el comando

push, que hará que sea el repositorio origen quien envíe los datos

directamente al destino.

Ejemplos:

cm pull br:/main@project1@remoteserver:8084 projectx@myserver:8084

(Hace pull de la rama principal de remoteserver a myserver. En este caso,

ambos servidores están configurados con el mismo modo de autenticación.)

cm pull br:/main@project1@remoteserver:8084 projectx@myserver:8084 --authmode=LDAPWorkingMode --authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(Hace pull de la misma rama que antes, pero ahora el servidor remoto está

configurado para autenticar usuarios contra un Active Directory. Por ejemplo,

se puede conectar de una máquina Linux a un servidor Windows configurado

para usar Active Directory. Para ello, se especificará el usuario y la

contraseña cifrada.)

cm pull br:/main@project1@remoteserver:8084 projectx@myserver:8084 --authmode=UPWorkingMode --user=dave --password=mysecret

(Hace pull de la misma rama, pero ahora ambos usuarios están autenticados

en el servidor remoto mediante el sistema de gestión de usuarios integrado

en Plastic SCM.)

cm pull br:/main@project1@remoteserver:8084 projectx@myserver:8084 --nodata

(Hace pull de la rama principal de remoteserver a myserver pero sin datos.)

cm pull hydrate br:/main@projectx@myserver:8084 projectx@remoteserver:8084

(Introduce los datos de todos los changesets de la rama principal obteniendo

dichos datos del repositorio en remoteserver.)

cm pull hydrate cs:122169@projectx@myserver:8084 projectx@remoteserver:8084

(Introduce los datos del changeset 122169 en myserver obteniendo dichos

datos del repositorio en remoteserver.)

Nota adicional:

La tabla de traducción tiene entradas (una por línea) de la forma:

nombre antiguo;nombre nuevo

== CMD\_DESCRIPTION\_PUSH ==

Replica datos hacia un repositorio remoto.

== CMD\_USAGE\_PUSH ==

Uso:

cm push src\_br\_spec dst\_rep\_spec

[--preview] [translateOptions]

[--user=usr\_name [--password=pwd] | AuthOptions]

(Replicación directa entre servidores. Hace push de una rama hacia un

repositorio.)

cm push src\_br\_spec --package=pack\_file [AuthOptions]

(Replicación basada en paquetes. Crea un paquete de replicación con la

rama especificada.)

src\_br\_spec La rama a ser replicada desde un repositorio remoto.

Especificación de ramas:

[br:][/]nombre\_rama[@[rep:]nombre\_rep[@[repserver:]nombre\_servidor:puerto]]

dst\_cs\_spec El changeset a ser hidratado.

Especificación de changesets:

cs:numero\_cset[@[rep:]nombre\_rep[@[repserver:]nombre\_servidor:puerto]]

dst\_rep\_spec El repositorio de destino.

Especificación de repositorio:

[rep:]nombre\_rep[@[repserver:]nombre\_servidor:puerto]

--package Exporta un paquete de replicación al fichero especificado.

Es útil para mover datos entre servidores sin una

conexión de red directa.

Opciones:

--preview Proporciona información sobre qué cambios serán enviados

durante el push, pero no se ejecuta ninguna acción más.

Esta opción es útil para comprobar qué datos serán

replicados antes de ejecutar el push.

--user, --password Credenciales para utilizar en caso de que el modo de

autenticación de los servidores origen y destino

difieran, y no exista un profile para autenticarse

contra el remoto.

--nodata Replica los metadatos de una rama sin replicar los

datos. Esta opción no se puede utilizar al exportar un

paquete de replicación.

Opciones de traducción (translateOptions):

--trmode={copy|name|table}

El servidor de origen y de destino pueden utilizar modos distintos de

autenticación. Este argumento especifica cómo se han de traducir los

nombres de usuario del origen al destino.

- copy El modo por defecto, indica que los nombres de usuario serán

simplemente copiados.

- name Los nombres de usuario serán emparejados entre origen y

destino por coincidencia.

- table Se utilizará una tabla de traducción (más información a

continuación).

--trtable={translation\_table\_file}

Si el modo de traducción es 'table', entonces una tabla de traducción

es un fichero que contiene líneas de la forma nombreantiguo;nombrenuevo.

Cuando la rama se escribe en el repositorio de destino, los objetos

creados por un usuario identificado por "nombreantiguo" en el

repositorio de origen serán asignados al usuario identificado por

"nombrenuevo" en el repositorio de destino.

Opciones de autentificación (AuthOptions):

Hay dos formas de especificar los datos de autentificación:

1) Especificando la autentificación mediante parámetros:

--authmode={NameWorkingMode|LDAPWorkingMode|ADWorkingMode|UPWorkingMode}

(LDAPWorkingMode) --authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(UPWorkingMode) --authdata=dave:fPBea2rPsQaagEW3pKNveA==

El parámetro '--authdata' es el contenido del campo <SecurityConfig>

en los ficheros client.conf o profiles.conf. El fichero profiles.conf

puede ser generado desde la GUI de Plastic SCM (Windows).

Si se utiliza UPWorkingMode, se puede especificar simplemente:

--authmode=UPWorkingMode --user=user --password=pwd

2.1) Para UPWorkingMode o LDAPWorkingMode:

--authdata=datos\_de\_autentificación

Ejemplos:

--authdata=ActiveDirectory:192.168.1.3:389:john.doe@factory.com:fPBea2rPsQaagEW3pKNveA==:dc=factory,dc=com (LDAPWorkingMode)

--authdata=john:fPBea2rPsQaagEW3pKNveA== (UPWorkingMode)

2.2) Solamente para UPWorkingMode:

--user=usuario

--password=contraseña

1) Especificando un fichero de autentificación. Se pueden tener varios

ficheros de autenticación, uno por servidor al que se conecte,

conteniendo las credenciales específicas para dicho servidor.

--authfile=fichero

Contiene 2 líneas:

Línea 1) modo de autenticación, igual que para --authmode.

Línea 2) datos de autenticación, igual que para --authdata.

== CMD\_HELP\_PUSH ==

Notas:

El comando pull replica ramas (junto a sus changesets) entre un repositorio

origen y otro destino. Los repositorios pueden encontrarse en diferentes

servidores.

Hay dos operaciones de replicación: push y pull.

Una operación de 'push' significa que la operación de replicación enviará

datos desde el repositorio origen al repositorio destino. En este caso, el

cliente se conectará al repositorio origen, recuperando los datos para ser

replicados, y entonces los enviará al repositorio destino. Mientras que el

origen necesita tener conectividad con el destino, el destino no se

conectará con el origen.

En un escenario distribuido típico el desarrollador hace push de los

cambios en su servidor local al servidor central. También puede ser

necesario hacer pull de los últimos cambios en el servidor central al

servidor local.

La replicación puede resolver situaciones en las que se hagan cambios

concurrentes en la misma rama en dos repositorios replicados:

- Push: si se intenta hacer push de datos locales a un repositorio que tenga

datos más nuevos que los que se están enviando, el sistema pedirá hacer

pull de los últimos cambios, resolver el merge resultante y, finalmente,

tratar de repetir el push.

- Pull: cuando se hace pull de changesets de una rama remota, estos estarán

correctamente enlazados a sus changesets padre. Si el changeset que se ha

traído no es un hijo del último changeset de la rama, entonces aparecerá

un escenario de múltiples cabezas. La rama tendrá más de una 'cabeza', o

dicho de otro modo, más de un changeset final de rama. Será necesario

hacer merge de ambas cabezas antes de que se pueda hacer push de nuevo.

El push puede funcionar de dos modos:

1) Comunicación directa entre servidores: lo que significa que el servidor

de origen se comunicará con el de destino para sincronizar la rama

especificada.

2) Exportación de un paquete de replicación: El cliente únicamente se

conectará con el origen y generará un paquete de replicación que contenga

tanto los datos como los metadatos para la rama especificada. Para ello

se utilizará el argumento --package.

Ambos modos requeren que el usuario ejecutando el comando esté autenticado

contra el servidor, bien o usando la autenticación por defecto en el fichero

client.conf, mediante un profile creado previamente, o especificando los

argumentos --authmode y --authdata (o --user y --password en caso de que

el modo de autenticación sea UPWorkingMode).

La replicación push funciona de una manera directa. Cuando se ejecuta, el

comando replicará la rama seleccionada del origen al destino, en vez de

pedir al repositorio destino que se conecte al repositorio origen para

obtener la rama especificada (como hace pull).

Ejemplos:

cm push br:/main@project1@myserver:8084 projectx@remoteserver:8084

(Hace push de la rama main desde myserver a remoteserver. En este caso,

ambos servidores están configurados con el mismo modo de autenticación.)

cm push br:/main@project1@remoteserver:8084 projectx@myserver:8084 --authmode=LDAPWorkingMode --authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(Hace push de la misma rama que antes, pero ahora el servidor remoto está

configurado para autenticar usuarios contra un Active Directory. Por ejemplo,

se puede conectar de una máquina Linux a un servidor Windows configurado

para usar Active Directory. Para ello, se especificará el usuario y la

contraseña cifrada.)

cm push br:/main@project1@remoteserver:8084 projectx@myserver:8084 --authmode=UPWorkingMode --user=dave --password=mysecret

(Hace push de la misma rama, pero ahora ambos usuarios están autenticados

en el servidor remoto mediante el sistema de gestión de usuarios integrado

en Plastic SCM.)

Nota adicional:

La tabla de traducción tiene entradas (una por línea) de la forma:

nombre antiguo;nombre nuevo

== CMD\_DESCRIPTION\_CLONE ==

Clona un repositorio remoto.

== CMD\_USAGE\_CLONE ==

Sintaxis:

cm clone src\_rep\_spec [dst\_rep\_spec | repserver:dst\_repserver\_spec]

[--user=usr\_name [--password=pwd] | AuthOptions]

[TranslateOptions]

(Clonado directo repositorio-a-repositorio.)

cm clone src\_rep\_spec --package=pack\_file

[--user=usr\_name [--password=pwd] | AuthOptions]

(Clonado utilizando un paquete intermedio, que se podrá importar

posteriormente al repositorio destino mediante un pull.)

src\_rep\_spec Repositorio origen de la operación de clonado.

dst\_rep\_spec Repositorio destino de la operación de clonado.

Si existe, debe estar vacío. Si no existe, será creado

automáticamente.

Si no se especifica, el comando utilizará el servidor

de repositorios por defecto.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de repositorios.

dst\_repserver\_spec Servidor de repositorios destino de la operación de

clonado. Si existe un repositorio con el mismo nombre

que src\_rep\_spec, debe estar vacío. Si no existe, será

creado automáticamente.

Si no se especifica, el comando utilizará el servidor

de repositorios por defecto.

Usa 'cm help objectspec' para obtener más información

sobre la especificación servidores de repositorio.

pack\_file Ruta de destino del paquete.

Options:

--user, --password Credenciales para utilizar en caso de que el modo de

autenticación de los servidores origen y destino

difieran, y no exista un profile para autenticarse

contra el remoto.

--package Exporta un paquete de replicación al fichero

especificado.

Es útil para mover datos entre servidores sin una

conexión de red directa.

El paquete resultante ha de ser importado mediante el

comand pull.

Opciones de traducción (TranslateOptions):

--trmode={copy|name|table}

El servidor de origen y de destino pueden utilizar modos distintos de

autenticación. Este argumento especifica cómo se han de traducir los

nombres de usuario del origen al destino.

- copy El modo por defecto, indica que los nombres de usuario serán

simplemente copiados.

- name Los nombres de usuario serán emparejados entre origen y

destino por coincidencia.

- table Se utilizará una tabla de traducción (más información a

continuación).

--trtable={translation\_table\_file}

Si el modo de traducción es 'table', entonces una tabla de traducción

es un fichero que contiene líneas de la forma nombreantiguo;nombrenuevo.

Cuando la rama se escribe en el repositorio de destino, los objetos

creados por un usuario identificado por "nombreantiguo" en el

repositorio de origen serán asignados al usuario identificado por

"nombrenuevo" en el repositorio de destino.

Opciones de autentificación (AuthOptions):

Hay dos formas de especificar los datos de autentificación:

1) Especificando la autentificación mediante parámetros:

--authmode={NameWorkingMode|LDAPWorkingMode|ADWorkingMode|UPWorkingMode}

(LDAPWorkingMode) --authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(UPWorkingMode) --authdata=dave:fPBea2rPsQaagEW3pKNveA==

El parámetro '--authdata' es el contenido del campo <SecurityConfig>

en los ficheros client.conf o profiles.conf. El fichero profiles.conf

puede ser generado desde la GUI de Plastic SCM (Windows).

Si se utiliza UPWorkingMode, se puede especificar simplemente:

--authmode=UPWorkingMode --user=user --password=pwd

2.1) Para UPWorkingMode o LDAPWorkingMode:

--authdata=datos\_de\_autentificación

Ejemplos:

--authdata=ActiveDirectory:192.168.1.3:389:john.doe@factory.com:fPBea2rPsQaagEW3pKNveA==:dc=factory,dc=com (LDAPWorkingMode)

--authdata=john:fPBea2rPsQaagEW3pKNveA== (UPWorkingMode)

2.2) Solamente para UPWorkingMode:

--user=usuario

--password=contraseña

1) Especificando un fichero de autentificación. Se pueden tener varios

ficheros de autenticación, uno por servidor al que se conecte,

conteniendo las credenciales específicas para dicho servidor.

--authfile=fichero

Contiene 2 líneas:

Línea 1) modo de autenticación, igual que para --authmode.

Línea 2) datos de autenticación, igual que para --authdata.

== CMD\_HELP\_CLONE ==

Notas:

El comando clone replica ramas (junto a sus changesets, etiquetas, atributos,

reviews, etc.) desde un repositorio origen a otro repositorio destino.

Los repositorios pueden encontrarse en diferentes servidores.

El repositorio destino puede existir de antemano, pero si contiene datos

previos, la operación de clone fallará.

La operación de clone NO clona submódulos, ni repositorios que se encuentren

debajo de un xLink.

Ejemplos:

cm clone awesomeProject@tardis@cloud

(Clona el repositorio 'awesomeProject' de la organización Cloud tardis@cloud

en un repositorio local del mismo nombre.)

cm clone repo@server.home:9095 repo-local

(Clona 'repo' desde 'server.home:9095' en 'repo-local' localizado en el

servidor de repositorios por defecto del usuario.)

cm clone project@192.168.111.130:8084 repserver:192.168.111.200:9095

(Clona 'project' desde '192.168.111.130:8084' en

'project@192.168.111.200:9095'.)

cm clone project@ldapserver:8084 --authfile=credentials.txt --trmode=table --trtable=table.txt

(Clona el repositorio 'project' desde 'ldapserver:8084' utilizando un fichero

para autenticarse contra el servidor remoto, y traduciendo los usuarios

siguiendo la tabla de traducción especificada.)

cm clone project@server.home:9095 --package=project.plasticpkg

cm mkrep project@mordor.home:8084

cm pull --package=project.plasticpkg project@mordor.home:8084

(Clona 'project' desde 'server.home:9095' al paquete 'project.plasticpkg',

que se importa posteriormente en el repositorio 'project' de

'mordor.home:8084' mediante una operación de pull.)

Nota adicional:

La tabla de traducción tiene entradas (una por línea) de la forma:

nombre antiguo;nombre nuevo

== CMD\_DESCRIPTION\_REVERT ==

Carga en el espacio de trabajo la revisión especificada y desprotege el elemento.

== CMD\_USAGE\_REVERT ==

Sintaxis:

cm revert cset\_spec

cset\_spec Especificación del changeset que contiene la revisión del ítem

cuyo contenido se desea cargar en el espacio de trabajo.

Usa 'cm help objectspec' para obtener más información sobre la

especificación de changesets.

== CMD\_HELP\_REVERT ==

Notas:

El ítem ha de esta estar protegido.

Ejemplos:

cm revert dir#cs:0

cm revert C:\mywks\dir\file1.txt#cs:23456

== CMD\_DESCRIPTION\_REVISION\_HISTORY ==

Muestra la historia de revisiones de un ítem.

== CMD\_USAGE\_REVISION\_HISTORY ==

Sintaxis:

cm history [opciones] rutas

rutas: Rutas de los ítems a mostrar, separadas por comas.

También pueden ser especificaciones de revisiones en servidor.

Usa comillas (") para indicar rutas que contengan espacios.

Usa 'cm help objectspec' para obtener más información sobre la

especificación de estas rutas.

Opciones:

-R: Descender recursivamente dentro de los directorios.

--long: Muestra información adicional.

--format: Muestra la salida con el formato especificado.

No compatible con la opción '--long'. Significado:

{0} fecha, {1} changeset, {2} rama, {3} etiquetas,

{4} comentario, {5} propietario, {6} revisionid

--symlink: Efectúa la operación "history" sobre el fichero de link y

no sobre el ítem al que apunta

--xml: Permite volcar la salida del comando en formato xml a la salida

estándar. También se permite especificar un fichero de salida

(--xml=output.xml).

--encoding: si se utiliza con la opción --xml, permite especificar el

encoding con el que se guardará la salida del comando. Consulte la

documentación de la MSDN para obtener la tabla completa de codificaciones

soportadas y su formato; al final de la página en la columna "Name":

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

== CMD\_HELP\_REVISION\_HISTORY ==

Notas:

Este comando muestra una lista con todas revisiones de un ítem determinado.

Para cada una de ellas se muestra la rama a la que pertenece, comentarios

de checkin y información sobre las etiquetas.

Este comando acepta formato para mostrar la salida.

Los parámetros que acepta son los siguientes:

{0} | {date} Fecha.

{1} | {changesetid} Número de changeset.

{2} | {branch} Rama.

{4} | {comment} Comentario.

{5} | {owner} Propietario.

{6} | {id} Id de revisión.

{7} | {repository} Repositorio.

{8} | {server} Servidor.

{9} | {repspec} Especificación de repositorio.

{tab} Inserta un tabulador.

{newline} Inserta una nueva línea.

Ejemplos

cm history fichero1.txt fichero2.txt

cm history c:\workspace

cm history link --symlink

(Recupera la historia del fichero de link y no sobre el elemento al que

apunta)

cm history serverpath:/src/foo/bar.c#br:/main/task001@myserver

(Recupera la historia del elemento /src/foo/bar.c presente en el último

changeset de la rama /main/task001 del repositorio 'myserver')

== CMD\_DESCRIPTION\_REVISION\_TREE ==

Muestra un árbol de revisiones para el ítem dado.

== CMD\_USAGE\_REVISION\_TREE ==

Sintaxis:

cm tree ruta

ruta: Ruta del ítem sobre el que se mostrará el árbol.

Opciones:

--symlink: Efectúa la operación sobre el fichero de link y no sobre el ítem

al que apunta.

== CMD\_HELP\_REVISION\_TREE ==

Ejemplos:

cm tree fichero1.txt

cm tree c:\workspace

cm tree link --symlink

(Efectúa la operación sobre el fichero de link

y no sobre el que apunta, disponible en entornos UNIX).

== CMD\_DESCRIPTION\_RM ==

Use este comando para borrar ficheros y directorios.

== CMD\_USAGE\_RM ==

Sintaxis:

cm remove | em comando [opciones]

Comandos:

controlled (opcional)

private

Para obtener más información sobre cada comando:

cm remove comando --usage

cm remove comando --help

== CMD\_HELP\_RM ==

Ejemplos:

cm remove \fichero\_controlado.txt

cm remove private \fichero\_privado.txt

== CMD\_DESCRIPTION\_RM\_CONTROLLED ==

Borra un ítem del control de versiones.

== CMD\_USAGE\_RM\_CONTROLLED ==

Sintaxis:

cm remove [opciones] rutas

rutas: Rutas de los ítem a borrar.

Opciones:

-R: Descender recursivamente dentro de directorios.

== CMD\_HELP\_RM\_CONTROLLED ==

Notas:

El ítem nunca borra del disco.

Requisitos para borrar:

El ítem ha de estar controlado en Plastic SCM.

El directorio padre debe estar desprotegido.

El elemento no debe estar desprotegido.

Si el ítem es un directorio, no debe tener desprotecciones pendientes.

Ejemplos:

cm remove src (borra el directorio 'src')

cm remove c:\workspace\fichero.txt

== CMD\_DESCRIPTION\_RM\_PRIVATE ==

Borra ficheros y directorios privados.

Advertencia: este comando elimina permanentemente ficheros y directorios, y no

son recuperables. Es recomendable utilizar antes la opción '--dry-run' para

comprobar qué ficheros y directorios serán afectados por el comando.

== CMD\_USAGE\_RM\_PRIVATE ==

Sintaxis:

cm remove | rm private <ruta>+ [-r] [--verbose] [--dry-run]

ruta Ruta del fichero o directorio a borrar.

Se pueden usar comillas (") para especificar rutas que

contengan espacios.

Opciones:

--r Elimina ficheros privados recursivamente dentro de

directorios controlados.

--ignored También se elimina ficheros y directorios ignorados y

cloaked.

--verbose Muestra todos las rutas afectadas.

--dry-run Ejecuta el comando sin hacer cambios en disco.

== CMD\_HELP\_RM\_PRIVATE ==

Notas:

Si la ruta especificada corresponde a un fichero o directorio privados, este

será eliminado de disco.

Si la ruta corresponde a un fichero bajo control de versiones, el comando

fallará.

Si la ruta pertenece a un directorio bajo control de versiones, el comando

fallará salvo que se especifique la opción '-r', en cuyo caso se borrarán

todos los ficheros y directorios privados dentro del directorio controlado

especificado.

Ejemplos:

cm remove private private\_directory

(Eliminina el directorio 'private\_directory'.)

cm remove private c:\workspace\controlled\_directory

(Falla, pues el directorio 'controlled\_directory' no es privado.)

cm remove private -r c:\workspace\controlled\_directory

(Elimina todos los ficheros y directorios privados dentro de

'controlled\_directory'.)

cm rm private --dry-run --verbose c:\workspace\controlled\_directory -r

(Muestra todas las rutas afectadas por la operación de borrado de elementos

privados dentro de 'controlled\_directory' sin borrar nada.)

cm rm private --verbose c:\workspace\controlled\_directory -r

(Muestra todas las rutas afectadas por la operación de borrado de elementos

privados dentro de 'controlled\_directory', ejecutando el borrado.)

== CMD\_DESCRIPTION\_TRIGGER\_DELETE ==

Elimina un trigger identificado por un tipo y una posicion de un servidor dado.

== CMD\_USAGE\_TRIGGER\_DELETE ==

Sintaxis:

cm trigger delete subtipo-tipo posicion [opciones]

subtipo-tipo: momento de ejecución del trigger y operacion del trigger.

Teclee 'cm showtriggertypes' para ver una lista completa de los tipos

de trigger disponibles. Algunos ejemplos son:

before-mkbranch

after-mkwokspace

after-setselector

posicion: posicion que se le asigno al trigger en el momento de su creación.

Opciones:

--server=servidor: Elimina el trigger del servidor especificado.

Si no se especifica ninguno lo elimina en aquél configurado

en el cliente.

== CMD\_HELP\_TRIGGER\_DELETE ==

Ejemplos:

cm trigger delete after-setselector 4

cm tr rm after-setselector 4

== CMD\_DESCRIPTION\_ATTRIBUTE\_SET ==

Fija un atributo al objeto indicado.

== CMD\_USAGE\_ATTRIBUTE\_SET ==

Sintaxis:

cm attribute set spec\_atributo spec\_objeto valor\_atributo

spec\_atributo Especificación del atributo.

spec\_objeto Especificación del objeto sobre el que fijar el atributo.

Objetos válidos para este comando son: rama, changeset,

shelveset, etiqueta, item y revisión.

valor\_atributo Valor con el que se quiere fijar el atributo indicado

para ese objeto.

Usa 'cm help objectspec' para obtener más información sobre especificaciones.

== CMD\_HELP\_ATTRIBUTE\_SET ==

Notas:

A los objetos se les puede fijar un atributo para poder guardar información

adicional en el objeto.

Se pueden fijar atributos sobre los siguientes objetos:

ramas, changesets, shelvesets, etiquetas, items y revisiones.

Ejemplo:

cm attribute set att:Status br:/main/SCM105 Integrated

== CMD\_DESCRIPTION\_SETOWNER ==

Establece el propietario de un objeto.

== CMD\_USAGE\_SETOWNER ==

Sintaxis:

cm setowner | sto --user=usuario | --group=grupo spec\_objeto

--user Nombre de usuario. Nuevo propietario del objeto.

--group Nombre de grupo. Nuevo propietario del objeto.

spec\_objeto Especificación del objeto sobre el que asignar el nuevo

propietario.

Se puede asignar nuevo usuario a los siguientes objetos:

repserver, repositorio, rama, changeset, etiqueta, item,

revisión y atributo.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de objetos.

== CMD\_HELP\_SETOWNER ==

Notas:

Es posible modificar el propietario de un objeto mediante el comando

setowner. El propietario de un objeto puede ser un usuario o un grupo.

Para especificar un usuario se utiliza la opción –-user=xxx y para

especificar un grupo se utiliza la opción -–group=xxx. El objeto que se

quiere modificar se especifica mediante una especificación de objeto.

Se puede modificar el propietario de los siguientes objetos:

servidor de repositorios, repositorio, rama, changeset, etiqueta, item,

revisión y atributo.

Ejemplos:

cm setowner --user=danipen repserver:localhost:8084

(establece a 'danipen' como propietario del servidor de repositorios)

cm sto --group=development rep:principal@PlasticServer:8084

(establece al grupo 'development' como el propietario del repositorio 'principal')

== CMD\_DESCRIPTION\_SETSELECTOR ==

Establece un selector para un espacio de trabajo.

== CMD\_USAGE\_SETSELECTOR ==

Sintaxis:

cm setselector [opciones] [wk\_ruta | wk\_spec]

wk\_ruta Ruta del espacio de trabajo sobre el que establecer el

selector.

wk\_spec Especificación del espacio de trabajo sobre el que

establecer el selector.

Usa 'cm help objectspec' para obtener más información

sobre especificaciones de espacios de trabajo.

Opciones:

--file Fichero del que cargar un selector.

--ignorechanges Ignora el mensaje de aviso de cambios pendientes que se

muestra si existen cambios pendientes cuando se

actualiza el espacio de trabajo.

--forcedetailedprogress Fuerza mostrar progreso detallado incluso cuando se

redirige la salida estándar.

== CMD\_HELP\_SETSELECTOR ==

Notas:

Este comando establece un selector para un espacio de trabajo determinado.

Un workspace necesita información acerca de dónde obtener las revisiones

con las que trabajará el usuario. Para especificar esa información se

utilizan los selectores.

Mediante un selector se puede indicar, por ejemplo, que se carguen las

revisiones desde una rama determinada o desde la rama ‘main’, desde una

etiqueta, y especificar la rama donde se alojarán las desprotecciones.

Si no se especifica ningún fichero del que cargar el selector se abrirá

un editor de texto.

Indicar una ruta o una especificación de workspace es opcional. Si no se

especifica ninguno el selector se fijará para el espacio de trabajo en

el que se encuentra el usuario.

Un selector de ejemplo:

repository "default" // indica el repositorio con el que se trabaja

path "/" // se aplica al path raíz

branch "/main" // se obtendrán las últimas revisiones de br:/main

checkout "/main" // los checkouts se ubicarán en la rama br:/main

Ejemplos:

cm setselector

cm setselector --file=c:\selectores\miRama.xml

cm setselector --file=c:\selectores\rama5.xml MiWorkspace

cm setselector wk:workspace\_projA@BEARDTONGUE

== CMD\_DESCRIPTION\_SHELVE ==

El comando shelve almacena el contenido de las revisiones en checkout.

== CMD\_USAGE\_SHELVE ==

Este comando está obsoleto. Se ha reemplazado por 'cm shelveset'.

Sintaxis:

cm shelve [opciones] [rutas]

rutas: rutas de los items a guardar.

Opciones:

--apply=shelve\_spec : Restaura los contenidos almancenados en el

repositorio, identificados por "shelve\_spec" en el espacio de trabajo

local.

--delete=shelve\_spec: Elimina los contenidos almancenados en el repositorio

identificados por "shelve\_spec"

--all (-a): Los ítems cambiados, movidos y borrados localmente, en las

rutas dadas, también serán guardados.

--dependencies: Incluir las dependencias de los cambios locales entre los

ítems a guardar.

-c=comentarios: Añade un comentario al shelve creado en la operación.

--commentsfile=comment.txt: Carga el comentario del fichero especificado.

--mount: El punto de montaje del respositorio especificado.

--encoding=codificación: Especifica la codificación de los ficheros de

origen (por ejemplo, UTF-8 o Unicode)

--comparisonmethod=tipo: cualquiera de las siguientes opciones:

IgnoreEol: Ignora diferencias de final de línea.

IgnoreWhitespaces: Ignora diferencias de espacios en blanco.

IgnoreEolWhitespaces: Ignora diferencias de final de línea y espacios en

blanco.

NotIgnore: Detecta diferencias de final de línea y espacios en

blanco.

== CMD\_HELP\_SHELVE ==

Notas:

- Si no se especifican [opciones] ni [rutas], la operación involucrará a

todos los cambios pendientes en el espacio de trabajo.

- La operación de shelve se aplica siempre recursivamente desde la

ruta indicada.

- Requisitos para hacer shelve de un ítem:

\* El ítem debe estar controlado por Plastic SCM.

\* El ítem ha de estar desprotegido.

Ejemplos:

cm shelve

cm shelve fichero1.txt fichero2.txt

Comentarios:

Para especificar comentarios use el switch "-c" o "-m" del siguiente modo:

cm shelve -c="my comment"

cm shelve -m "my comment"

Si existe una variable de entorno llamada PLASTICEDITOR apuntando al

ejecutable de un editor de texto, y no se especifica un comentario al

crear una shelve, el editor se abrirá automáticamente para que especifique

un comentario.

Ejemplo: Aplicar un shelve almacenado en el repositorio:

cm shelve --apply=sh:3

Ejemplo: Eliminar un shelve almacenado en el repositorio

cm shelve --delete=sh:3

Ejemplo: Shelve 'changelist'

cm status --short --changelist=pending\_to\_review | cm shelve -

(El comando listará los paths en el changelist llamado 'pending\_to\_review'

y dicha lista será redirigida a la entrada del comando 'shelve').

== CMD\_DESCRIPTION\_SHELVESET ==

Use este comando para administrar shelvesets.

== CMD\_USAGE\_SHELVESET ==

Sintaxis:

cm shelveset comando [opciones]

Comandos:

create | mk

delete | rm

apply

Para obtener más información sobre cada comando:

cm shelveset command --usage

cm shelveset command --help

== CMD\_HELP\_ATTRIBUTE ==

Ejemplos:

cm shelveset create -c="my comment"

cm shelveset delete sh:3

cm shelve apply sh:3

== CMD\_DESCRIPTION\_SHELVESET\_CREATE ==

El comando shelve almacena los cambios pendientes.

== CMD\_USAGE\_SHELVESET\_CREATE ==

Sintaxis:

cm shelveset create [opciones] [rutas]

rutas: rutas de los items a guardar.

Opciones:

--all (-a): Los ítems cambiados, movidos y borrados localmente, en las

rutas dadas, también serán guardados.

--dependencies: Incluir las dependencias de los cambios locales entre los

ítems a guardar.

-c=comentarios: Añade un comentario al shelve creado en la operación.

--commentsfile=comment.txt: Carga el comentario del fichero especificado.

== CMD\_HELP\_SHELVESET ==

Ejemplos:

cm shelveset create -c="my comment"

cm shelveset delete sh:3

cm shelve apply sh:3

== CMD\_HELP\_SHELVESET\_CREATE ==

El comando shelveset create almacena los ítems en checkout especificados

dentro del repositorio. De este modo el contenido queda almacenado en el

servidor sin necesidad de hacer check in.

Notas:

- Si no se especifican [opciones] ni [rutas], la operación involucrará a

todos los cambios pendientes en el espacio de trabajo.

- La operación de shelve se aplica siempre recursivamente desde la

ruta indicada.

- Requisitos para hacer shelve de un ítem:

\* El ítem debe estar controlado por Plastic SCM.

\* El ítem ha de estar desprotegido.

Ejemplos:

cm shelveset create

cm shelveset fichero1.txt fichero2.txt

Comentarios:

Para especificar comentarios use el switch "-c" o "-m" del siguiente modo:

cm shelveset -c="my comment"

cm shelveset -m "my comment"

Si existe una variable de entorno llamada PLASTICEDITOR apuntando al

ejecutable de un editor de texto, y no se especifica un comentario al

crear una shelve, el editor se abrirá automáticamente para que especifique

un comentario.

Ejemplo: Shelve 'changelist'

cm status --short --changelist=pending\_to\_review | cm shelveset -

(El comando listará los paths en el changelist llamado 'pending\_to\_review'

y dicha lista será redirigida a la entrada del comando 'shelve').

== CMD\_DESCRIPTION\_SHELVESET\_DELETE ==

El comando shelveset delete borra un shelveset.

== CMD\_USAGE\_SHELVESET\_DELETE ==

Sintaxis:

cm shelveset delete shelve\_spec

shelve\_spec: Elimina los contenidos almancenados en el repositorio

identificados por "shelve\_spec"

== CMD\_HELP\_SHELVESET\_DELETE ==

El comando shelveset delete borra un shelveset creado anteriormente.

Ejemplo: Eliminar un shelve almacenado en el repositorio

cm shelveset delete sh:3

== CMD\_DESCRIPTION\_SHELVESET\_APPLY ==

El comando shelveset apply restaura los contenidos almancenados en el

repositorio.

== CMD\_USAGE\_SHELVESET\_APPLY ==

Sintaxis:

cm shelveset apply shelve\_spec [opciones]

shelve\_spec: Restaura los contenidos almancenados en el repositorio

identificados por "shelve\_spec"

Opciones:

--mount: El punto de montaje del respositorio especificado.

--encoding=codificación: Especifica la codificación de los ficheros de

origen (por ejemplo, UTF-8 o Unicode)

--comparisonmethod=tipo: cualquiera de las siguientes opciones:

IgnoreEol: Ignora diferencias de final de línea.

IgnoreWhitespaces: Ignora diferencias de espacios en blanco.

IgnoreEolWhitespaces: Ignora diferencias de final de línea y espacios en

blanco.

NotIgnore: Detecta diferencias de final de línea y espacios en

blanco.

== CMD\_HELP\_SHELVESET\_APPLY ==

El comando shelveset apply restaura los contenidos almancenados en el

repositorio.

Ejemplo: Aplicar un shelve almacenado en el repositorio:

cm shelveset apply sh:3

== CMD\_DESCRIPTION\_SHOW\_FIND\_OBJECTS ==

Muestra la lista de objetos disponibles y sus atributos correspondientes.

== CMD\_USAGE\_SHOW\_FIND\_OBJECTS ==

Sintaxis:

cm showfindobjects

== CMD\_HELP\_SHOW\_FIND\_OBJECTS ==

Objetos y atributos disponibles:

attribute:

Se puede buscar atributos filtrando por los siguientes campos:

Type : string.

Value : string.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para más

info.

Owner : usuario.

Admite el usuario especial 'me'.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Comment : string.

SrcObj : object spec.

ID : integer.

attributetype:

Se puede buscar tipos de atributo filtrando por los siguientes campos:

Name : string.

Value : string.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para más

info.

Owner : usuario.

Admite el usuario especial 'me'.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Comment : string.

Source : object spec.

ID : integer.

Campos de replicación. Consulte la sección "campos relacionados con la réplica"

en esta guía para más información sobre:

ReplLogId

ReplSrcDate

ReplSrcId

ReplSrcRepository

ReplSrcServer

branch:

Se puede buscar ramas filtrando por los siguientes campos:

Name : string.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para más

info.

Changesets : fecha (de los changesets en la rama).

Consulte la sección "constantes de fecha" en esta guía para más

info.

Attribute : string.

AttrValue : string.

Owner : usuario.

Admite el usuario especial 'me'.

Parent : branch spec.

Comment : string.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Item : item spec or item id (integer).

ID : integer.

Campos de replicación. Consulte la sección "campos relacionados con la réplica"

en esta guía para más información sobre:

ReplLogId

ReplSrcDate

ReplSrcId

ReplSrcRepository

ReplSrcServer

changeset:

Se puede buscar changesets filtrando por los siguientes campos:

Branch : branch spec.

ChangesetId : integer.

Attribute : string.

AttrValue : string.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía

para más info.

Owner : usuario.

Admite el usuario especial 'me'.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Comment : string.

OnlyWithRevisions : boolean.

ReturnParent : boolean.

Parent : changeset id (integer).

ID : integer.

Campos de replicación. Consulte la sección "campos relacionados con la réplica"

en esta guía para más información sobre:

ReplLogId

ReplSrcDate

ReplSrcId

ReplSrcRepository

ReplSrcServer

label:

Se puede buscar etiquetas filtrando por los siguientes campos:

Name : string.

Attribute : string.

AttrValue : string.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para más

info.

Owner : usuario.

Admite el usuario especial 'me'.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Branch : branch spec.

Branchid : integer.

Changeset : changeset id (integer).

Comment : string.

ID : integer.

Campos de replicación. Consulte la sección "campos relacionados con la réplica"

en esta guía para más información sobre:

ReplLogId

ReplSrcDate

ReplSrcId

ReplSrcRepository

ReplSrcServer

merge:

Se puede buscar merges filtrando por los siguientes campos:

SrcBranch : branch spec.

SrcChangeset : changeset id (integer).

DstBranch : branch spec.

DstChangeset : changeset id (integer).

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para

más info.

Owner : usuario.

Admite el usuario especial 'me'.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Type : string.

Los posibles valores son 'merge', 'cherrypick',

'cherrypicksubstractive', 'interval', 'intervalcherrypick'

e 'intervalcherrypicksubstractive'

ID : integer.

replicationlog:

Se puede buscar replication log filtrando por los siguientes campos:

Branch : branch spec.

RepositoryName : string.

Owner : usuario.

Admite el usuario especial 'me'.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para

más info.

Server : string.

Package : boolean.

ID : integer.

review:

Se puede buscar code reviews filtrando por los siguientes campos:

Status : string.

Assignee : string.

Title : string.

Target : object spec.

TargetId : integer.

TargetType : string.

Los posibles valores son 'branch' y 'changeset'.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para más

info.

Owner : usuario.

Admite el usuario especial 'me'.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Id : integer.

revision:

Se puede buscar revisiones filtrando por los siguientes campos:

Branch : branch spec.

Changeset : changeset id (integer).

Item : string o integer.

ItemId : integer.

Attribute : string.

AttrValue : string.

Archived : boolean.

Comment : string.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía

para más info.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Owner : usuario.

Admite el usuario especial 'me'.

Parent : revision id (integer).

ReturnParent : boolean.

Shelve : shelve id (integer).

Size : integer (en bytes).

Type : string.

Los posibles valores son 'dir', 'bin' y 'txt'.

WorkspaceCheckoutId : integer.

ID : integer.

Campos de replicación. Consulte la sección "campos relacionados con la réplica"

en esta guía para más información sobre:

ReplLogId

ReplSrcDate

ReplSrcId

ReplSrcRepository

ReplSrcServer

shelve:

Se puede buscar shelves filtrando por los siguientes campos:

Owner : usuario.

Admite el usuario especial 'me'.

Date : fecha.

Consulte la sección "constantes de fecha" en esta guía para más

info.

Attribute : string.

AttrValue : string.

Comment : string.

GUID : Identificador Global Único.

Id hexadecimal con el formato xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Parent : integer.

ShelveId : integer.

ID : integer.

Campos de replicación. Consulte la sección "campos relacionados con la réplica"

en esta guía para más información sobre:

ReplLogId

ReplSrcDate

ReplSrcId

ReplSrcRepository

ReplSrcServer

Campos relacionados con la réplica:

Muchos objetos mantienen información de replicación, lo que significa que

Plastic mantiene el rastro de dónde se crearon originalmente.

Los campos que se pueden utilizar son:

ReplSrcServer : Significa "servidor origen de la réplica", y es el

servidor de donde se trajeron los objetos.

Ejemplo:

cm find branch "where replsrcserver='skull.codicefactory.com:9095'"

ReplSrcRepository : string. Significa "repositorio origen de la réplica",

y es el repositorio de donde se trajeron los objetos.

Ejemplo:

cm find branch "where \

replsrcserver = 'skull.codicefactory.com:9095' \

and replsrcrepository = 'codice'"

ReplLogId : integer. El identificador de la operación de réplica. En

Plastic, cada vez que un objeto nuevo se crea de una réplica,

también se crea un nuevo 'replicationlog'.

Ejemplo:

cm find replicationlog

324054 9/13/2018 02:00:15 /main/scm23064 sluisp plasticscm.com T

327255 9/11/2018 12:33:38 /main/scm23042 maria plasticscm.com T

329631 9/17/2018 13:06:49 /main/scm23099 sluisp plasticscm.com T

Ahora se pueden buscar las ramas creadas en la réplica 327255, en

caso de haber alguna:

cm find branch "where replogid=327255"

ReplSrcDate: fecha. Es la fecha en la que la operación de réplica tuvo

lugar. Los objetos replicados mantienen la fecha en la que

fueron creados originalmente, por lo que este campo es útil

si se necesita buscar objetos que fueron replicados en un

determinado marco temporal.

Ejemplo:

cm find replicationlog "where date > 'one week ago'"

8780433 27/09/2018 8:49:38 codice@BACKYARD:8087 F mbarriosc

Ahora se puede verificar que la rama replicada se creó antes de la

propia operación de réplica:

cm find branch "where repllogid = 8780433"

8780443 26/09/2018 12:20:55 /main/scm23078 maria codice T

ReplSrcId: integer. Es el ID del servidor origen de la réplica

(ReplSrcServer). Este ID se puede descubrir buscando objetos

de tipo 'replicationsource' con el comando 'cm find'.

Ejemplo:

cm find replicationsource

7860739 codice@AFRODITA:8087 d9c4372a-dc55-4fdc-ad3d-baeb2e975f27

8175854 codice@BACKYARD:8087 66700d3a-036b-4b9a-a26f-adfc336b14f9

Ahora se pueden buscar los changesets replicados desde

codice@AFRODITA:8087 de la siguiente forma:

cm find changesets "where replsrcid = 7860739"

Constantes de fecha:

Se pueden usar fechas formateadas siguiendo la configuración de localización

del Sistema Operativo. Por ejemplo, si el SO muestra fechas en el formato

'dd/MM/yyyy', se pueden utilizar fechas como '31/12/2018' en las consultas.

Sin embargo, también se pueden utilizar las siguientes constantes para

facilitar la escritura de las queries:

'today' : la fecha de hoy.

'yesterday' : la fecha de ayer.

'this week' : la fecha del lunes de la semana actual.

'this month' : la fecha del primer día del mes actual.

'this year' : la fecha del 1 de enero del año actual.

'one day ago' : un día antes de la fecha actual.

'one week ago' : una semana antes de la fecha actual.

'one month ago' : un mes antes de la fecha actual.

'n days ago' : 'n' días antes de la fecha actual.

'n months ago' : 'n' meses antes de la fecha actual.

'n years ago' : 'n' años antes de la fecha actual.

Las siguientes cláusulas 'where' son válidas para campos que admitan fechas:

'(...) where date > 'today' (...)'

'(...) where date < 'yesterday' (...)'

'(...) where date > 'this week' (...)'

'(...) where date > 'this month' (...)'

'(...) where date < 'one day ago' and date > '3 days ago' (...)'

'(...) where date < 'one week ago' and date > '3 weeks ago' (...)'

'(...) where date < 'one month ago' and date > '3 months ago' (...)'

'(...) where date > '1 year ago' (...)'

También se puede forzar un determinado formato de fecha en el comando

'cm find' utilizando el flag --dateformat. Lee 'cm find --help' para más

información.

== CMD\_DESCRIPTION\_TRIGGER\_SHOWTYPES ==

Muestra la lista de los tipos de triggers disponibles.

== CMD\_USAGE\_TRIGGER\_SHOWTYPES ==

Sintaxis:

cm trigger showtypes

== CMD\_DESCRIPTION\_SHOWACL ==

Muestra la ACL (Lista de Control de Acceso) de un elemento.

== CMD\_USAGE\_SHOWACL ==

Sintaxis:

cm showacl [opciones] spec\_objeto

spec\_objeto Especificación del objeto sobre el que mostrar el ACL.

Objetos válidos para este comando son repserver,

repositorio, rama, changeset, etiqueta, item y atributo.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de objetos.

Opciones:

--extended Muestra información detallada.

--xml Permite volcar la salida del comando en formato xml

a la salida estándar. También se permite especificar

un fichero de salida (--xml=output.xml).

--encoding Si se utiliza con la opción --xml, permite especificar

el encoding con el que se guardará la salida del

comando. Por ejemplo: --encoding=utf-8

Consulte la documentación de la MSDN para obtener la

tabla completa de codificaciones soportadas y su formato;

al final de la página en la columna "Name":

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

== CMD\_HELP\_SHOWACL ==

Notas:

Este comando se utiliza para obtener una lista de permisos de un usuario o

un grupo sobre un elemento del repositorio.

Ejemplos:

cm showacl repserver:PlasticServer:8084

cm showacl br:/main --extended

== CMD\_DESCRIPTION\_SHOWCOMMANDS ==

Muestra todos los comandos disponibles.

== CMD\_USAGE\_SHOWCOMMANDS ==

Sintaxis:

cm showcommands

== CMD\_HELP\_SHOWCOMMANDS ==

== CMD\_DESCRIPTION\_SHOWOWNER ==

Muestra el propietario de un elemento del repositorio.

== CMD\_USAGE\_SHOWOWNER ==

Sintaxis:

cm showowner spec\_objeto

spec\_objeto Especificación del objeto para mostar su propietario.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de objetos.

El objeto debe ser uno de los siguientes: repserver,

repositorio, rama, changeset, etiqueta, atributo,

revisión e item.

== CMD\_HELP\_SHOWOWNER ==

Notas:

Este comando muestra el propietario de un elemento del repositorio.

Puede ser un usuario o un grupo.

El propietario se puede modificar con el comando 'cm setowner'.

Ejemplos:

cm showowner repserver:PlasticServer:8084

cm showowner rev:fichero.cs#br:/main#LAST

== CMD\_DESCRIPTION\_SHOWPERMISSIONS ==

Muestra la lista de los permisos disponibles.

== CMD\_USAGE\_SHOWPERMISSIONS ==

Sintaxis:

cm showpermissions

== CMD\_HELP\_SHOWPERMISSIONS ==

Ejemplos:

cm showpermissions

== CMD\_DESCRIPTION\_SHOWSELECTOR ==

Muestra el selector del espacio de trabajo actual.

== CMD\_USAGE\_SHOWSELECTOR ==

Sintaxis:

cm showselector [wk\_ruta | wk\_spec]

wk\_ruta Ruta del espacio de trabajo para mostrar su selector

wk\_spec Especificación del espacio de trabajo para mostrar su

selector.

Usa 'cm help objectspec' para obtener más información sobre

especificaciones de espacios de trabajo.

== CMD\_HELP\_SHOWSELECTOR ==

Notas:

Si no se especifica una ruta o un especificador de espacio de trabajo,

el comando tomará como ruta el directorio actual.

Ejemplos:

cm showselector c:\workspace

cm showselector

cm showselector > miSelector.txt

cm showselector wk:build\_wk@BUILDER

== CMD\_DESCRIPTION\_SUPPORT ==

Este comando permite al usuario efectuar operaciones de soporte.

== CMD\_USAGE\_SUPPORT ==

Sintaxis:

cm support comando [opciones]

Comandos:

bundle

Para obtener más información sobre cada uno de los comandos ejecute:

cm support comando --usage

cm support comando --help

== CMD\_HELP\_SUPPORT ==

Ejemplos:

cm support

cm support bundle

cm support bundle c:\outputfile.zip

== CMD\_DESCRIPTION\_SUPPORT\_BUNDLE ==

Crea el paquete de soporte con los ficheros de log relevantes.

Este paquete se puede adjuntar cuando se solicita la ayuda, cuando

se pregunte por información extra o cuando se envíe un error.

== CMD\_USAGE\_SUPPORT\_BUNDLE ==

Sintaxis:

cm support bundle [outputfile]

Opciones:

outputfile Crea el paquete de soporte en la ruta especificado.

== CMD\_HELP\_SUPPORT\_BUNDLE ==

Notas:

Este comando permite al usuario crear un paquete de soporte con los

ficheros de log relevantes. Es posible especificar un fichero de salida.

Ejemplos:

cm support bundle

(Crea el paquete de soporte en el directorio temporal)

cm support bundle c:\outputfile.zip

(Crea el paquete de soporte en la ruta especificado)

== CMD\_DESCRIPTION\_SWITCH ==

Actualiza el espacio de trabajo a la rama, etiqueta, changeset o shelve especificado.

== CMD\_USAGE\_SWITCH ==

Sintaxis:

cm switch brspec | csetspec | lbspec | shspec [--repository=name]

[--workspace=path]

Opciones:

--workspace Ruta del espacio de trabajo a cambiar.

--repository Repositorio al que se va a cambiar.

Usa 'cm help objectspec' para aprender más sobre cómo indicar

especificaciones de changesets, ramas y etiquetas.

== CMD\_HELP\_SWITCH ==

Notas:

Este comando permite a los usuarios actualizar el espacio de trabajo

con el contenido del objeto especificado (rama, etiqueta o changeset).

Ejemplos:

cm switch br:/main

cm switch lb:Rel1.1

cm switch Rel2.0

cm switch cs:4375

cm switch 5632

cm switch sh:2

== CMD\_DESCRIPTION\_SWITCH\_TO\_BRANCH ==

Establece la rama especificada como rama de trabajo.

== CMD\_USAGE\_SWITCH\_TO\_BRANCH ==

Sintaxis:

cm switchtobranch [opciones] [spec\_rama]

spec\_rama: Especificación de rama.

Opciones:

--label=nombre | --changeset=número: Carga las revisiones de la etiqueta

o changeset especificado. Si no se especifica spec\_rama, una de

estas opciones es obligatoria.

--workspace | -wk=path: Ruta del espacio de trabajo a cambiar.

== CMD\_HELP\_SWITCH\_TO\_BRANCH ==

Notas:

Este comando permite a los usuarios especificar una rama, como rama de

trabajo. Adicionalmente se puede especificar una etiqueta o un changeset,

desde la cual se cargarán las revisiones.

Si no se especifica rama, se ha de especificar una etiqueta o un changeset

obligatoriamente. Consulte el manual para obtener más información.

Si no se especifica repositorio, la rama se fija sobre el repositorio

actual.

Ejemplos:

cm switchtobranch br:/main

cm switchtobranch br:/main/tarea001

cm switchtobranch --label=BL050

(Configuración de sólo lectura. Se cargan revisiones del changeset etiquetado)

== CMD\_DESCRIPTION\_SYNC ==

Sincroniza con Git

== CMD\_USAGE\_SYNC ==

Sintaxis:

cm synchronize | sync repspec git [url [--user=usr\_name --pwd=pwd]] [--author]

[[--txtsimilaritypercent | --binsimilaritypercent | --dirsimilaritypercent]=value]

repspec Especificación del repositorio.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de repositorios.

git (Default).

Opciones:

url URL del repositorio remoto

(http(s):// o git:// o una URL ssh).

--user Nombre de usuario para la URL especificada.

--pwd Contraseña para la URL especificada.

--txtsimilaritypercent | --binsimilaritypercent | --dirsimilaritypercent

Opciones para detectar ítems movidos, del mismo modo que

lo hace la GUI de Plastic SCM.

--author Usa los valores de nombre y fecha (timestamp) del autor

en git (git committer by default).

--skipgitlfs Ignora la configuración de Git LFS que haya en el

fichero '.gitattributes'. Se comporta como si no

tuviese soporte de Git LFS.

== CMD\_HELP\_SYNC ==

Notas:

En caso de sincronizar con un servidor que no requiera credenciales, tras

la primera sincronización no es necesario introducir el parámetro URL.

Este comando no soporta repositorios locales.

En el caso de usar el protocolo SSH, es necesario tener añadido al PATH

el cliente por línea de comandos 'ssh' y que esté correctamente configurado

para conectarse al servidor remoto (claves públicas/privadas).

Ejemplos:

cm sync default@localhost:8087 git git://localhost/repository

== CMD\_DESCRIPTION\_TRIGGER ==

Use este comando para administrar triggers.

== CMD\_USAGE\_TRIGGER ==

Sintaxis:

cm trigger | tr command [opciones]

Comandos:

create | mk

delete | rm

edit

list | ls

showtypes

Para obtener más información sobre cada comando:

cm trigger command --usage

cm trigger command --help

== CMD\_HELP\_TRIGGER ==

Ejemplos:

cm tr mk before-mklabel new "/path/to/script" --server=myserver:8084

cm tr edit before-mklabel 7 --position=4 --server=myserver:8084

cm tr ls before-mkbranch --server=myserver:8084

cm tr rm after-setselector 4

cm tr showtypes

== CMD\_DESCRIPTION\_TUBE ==

Ejecuta comandos relacionados con Plastic Tube.

== CMD\_USAGE\_TUBE ==

Sintaxis:

cm tube config -u=user -p=password

cm tube create remoteuser

cm tube remove remoteuser

cm tube local

cm tube remote

cm tube share <repository\_spec>+ -u=remoteuser -a=access\_mode

cm tube unshare <repository\_spec>+ -u=remoteuser

cm tube connect

cm tube disconnect

cm tube status

Opciones:

-u Usuario de Tube (el usuario de plasticscm.com)

-p Password del usuario

-a Acceso: pull, push o ambos (pull | push | pull,push)

rep\_spec Especificación del repositorio.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de repositorios.

== CMD\_HELP\_TUBE ==

Notas:

config

Configura Plastic SCM para usar Plastic Tube con el usuario y password

especificados.

create

Crea el Tube remoteuser -> myuser.

El usuario 'myuser' permite a 'remoteuser' conectarse al servidor

de 'myuser'.

Pueden establecerse conexiones de 'remoteuser' a 'myuser'.

Sólo pueden crearse un Tube de otro usuario al usuario actual de

Plastic Tube.

remove

Elimina el Tube remoteuser -> myuser.

local

Lista los repositorios locales compartidos en el servidor local y los

usuarios con quien son compartidos.

remote

Lista los repositorios remotamente compartidos que son compartidos con

el usuario actual de Plastic Tube.

share

Comparte el repositorio(s) local con el usuario y remoto y los permisos

especificados.

unshare

Deja de compartir el repositorio local con el usuario remoto.

connect

Conecta el servidor de Plastic SCM a Plastic Tube.

disconnect

Desconecta el servidor de Plastic SCM de Plastic Tube.

status

Muestra si el servidor de Plastic SCM está conectado a Plastic Tube.

Ejemplos:

cm tube config -u=ruben@codicesoftware.com -p=rubenpassword

cm tube create pablo@codicesoftware.com

('pablo@codicesoftware.com' puede conectar con el usuario actual del Tube)

cm tube remove pablo@codicesoftware.com

cm tube local

cm tube remote

cm tube share repo@server:8087 -u=pablo@codicesoftware.com -a=pull,push

cm tube share repo@server:8087 doc@server:8087 -u=pablo@codicesoftware.com -a=push

cm tube unshare repo@server:8087 -u=pablo@codicesoftware.com

cm tube connect

cm tube disconnect

cm tube status

== CMD\_DESCRIPTION\_UNCO ==

Deshace la desprotección de un ítem.

== CMD\_USAGE\_UNCO ==

Sintaxis:

cm undocheckout | unco <item\_path>+ [--all] [--symlink] [--silent]

item\_path Especificación de los ítems para deshacer la desprotección,

separados por comas.

Se pueden usar comillas (") para especificar rutas que

contengan espacios.

Use . para aplicar la operación al directorio actual.

Opciones:

--all (-a) Los ítems especificados que han sido cambiados, movidos y

borrados localmente también serán deshechos.

--symlink Deshace la protección del fichero de link y no al elemento

al que apunta.

--silent No muestra salida.

== CMD\_HELP\_UNCO ==

Notas:

Si se desean descartar los cambios realizados en un ítem desprotegido

previamente, se utiliza este comando.

El ítem se cambia a la revisión justamente anterior.

Requisitos para deshacer la desprotección:

El ítem ha de estar controlado por Plastic

El ítem ha de estar desprotegido

Ejemplos:

cm undocheckout .

(Desprotege el directorio actual)

cm undocheckout fichero1.txt fichero2.txt

cm unco c:\workspace\fichero.txt

cm undocheckout -R c:\workspace\src

(Deshace la desprotección de 'src' recursivamente)

cm unco link --symlink

(Efectúa la desprotección sobre el fichero de link y no del que apunta,

disponible en entornos UNIX)

cm unco code\cgame\cg\_main.c --all

(Deshace la desprotección del fichero cambiado)

Deshacer 'changelist'. Ejemplo:

cm status --short --changelist=pending\_to\_review | cm undocheckout -

(El comando listará los paths en el changelist llamado 'pending\_to\_review'

y dicha lista será redirigida a la entrada del comando 'uncheckout').

== CMD\_DESCRIPTION\_UNCOUNCHANGED ==

Deshace la desprotección en elementos que no han cambiado.

== CMD\_USAGE\_UNCOUNCHANGED ==

Sintaxis:

cm uncounchanged [opciones]

Opciones:

-R|-r|--recursive Descender recursivamente dentro de directorios.

== CMD\_HELP\_UNCOUNCHANGED ==

Notas:

Este comando se aplica desde la raíz del workspace recursivamente.

Ejemplos:

cm uncounchanged

== CMD\_DESCRIPTION\_UNDELETE ==

Recupera en el espacio de trabajo la revisión especificada en la ruta dada.

== CMD\_USAGE\_UNDELETE ==

Sintaxis:

cm undelete rev\_spec ruta

rev\_spec: Especificación de la revisión del ítem cuyo contenido se desea

cargar en el espacio de trabajo.

path: Ruta donde se va a restaurar.

== CMD\_HELP\_UNDELETE ==

Notas:

El elemento a recuperar no debe de estar ya cargado en el workspace.

La operación de recuperar no esta soportada para xlinks.

Ejemplo:

cm undelete revid:756 C:\mywks\src\foo.c

cm undelete itemid:68#cs:2 C:\mywks\dir\myfile.pdf

cm undelete serverpath:/src#br:/main C:\mywks\Dir

== CMD\_DESCRIPTION\_UNDOCHANGE ==

Deshace los cambios hechos en un fichero.

== CMD\_USAGE\_UNDOCHANGE ==

Notas:

Si se desean descartar los cambios en un ítem, independientemente de su

estado (desprotegido o modificado sin hacer desprotección), este comando

vuelve a la revisión anterior. Si se ordena realizar la operación sobre

un directorio en vez de un fichero la acción se realiza recursivamente por

defecto.

Ejemplo:

cm unc . (Descarta los cambios de los ítems del directorio actual).

cm unc fichero1.txt fichero2.txt

cm unc c:\workspace\fichero1.txt

== CMD\_HELP\_UNDOCHANGE ==

Ejemplo:

cm unc . (Descarta los cambios de los ítems del directorio actual).

cm unc fichero1.txt fichero2.txt

cm unc c:\workspace\fichero1.txt

== CMD\_DESCRIPTION\_LOCK\_UNLOCK ==

Deshace los bloqueos en los elementos del servidor de bloqueos especificado

== CMD\_USAGE\_LOCK\_UNLOCK ==

Sintaxis:

cm lock unlock [repserverspec] <guid>+

repserverspec Especificación del servidor de repositorio.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de servidores.

guid Lista de GUIDs de ítems a desbloquear, separados por

espacios.

== CMD\_DESCRIPTION\_UNDO ==

Deshace cambios en un workspace.

== CMD\_USAGE\_UNDO ==

Syntaxis:

cm undo [path]+ [--symlink] [-r | --recursive] [Filtros]+

[Opciones de salida]

path Rutas de los ficheros o carpetas a los que aplicar

la operación, separados por espacios. Se pueden

utilizar comillas (") para especificar rutas con

espacios. Si no se especifica ninguna ruta, por

defecto la operación se ejecutará sobre los ficheros

en el directorio actual.

Opciones:

--symlink Aplica la operación de undo al enlace simbólico y

no a su destino.

-r Ejecuta el undo de manera recursiva.

Filtros:

Si ninguna de estas opciones está presente, por defecto se deshacen todos

los tipos de cambio, pero los ficheros pueden filtrarse utilizando estas

opciones.

Si un fichero o directorio tiene uno o más de los tipos de cambio

especificados, todos los cambios en ese fichero o directorio serán

deshechos.

Por ejemplo, si se especifica tanto --checkedout como --moved, si un fichero

está tanto movido como checkedout, ambos cambios se desharán.

--checkedout Selecciona ficheros y directorios marcados para

cambios.

--unchanged Selecciona ficheros sin modificaciones en su

contenido.

--changed Selecciona ficheros y directorios cambiados o

marcados para cambios.

--deleted Selecciona ficheros y directorios eliminados.

--moved Selecciona ficheros y directorios movidos.

--added Selecciona ficheros y directorios añadidos.

Opciones de salida:

--silent | --machinereadable [--startlineseparator=sep]

[--endlineseparator=sep] [--fieldseparator=sep]

--silent No muestra ninguna salida.

--machinereadable Muestra la salida en un formato fácil de tratar.

--startlineseparator Usado en conjunto con la opción '--machinereadable',

especifica cómo deberían comenzar las líneas.

--endlineseparator Usado en conjunto con la opción '--machinereadable',

especifica cómo deberían terminar las líneas.

--fieldseparator Usado en conjunto con la opción '--machinereadable',

especifica cómo deberían separarse las líneas.

== CMD\_HELP\_UNDO ==

Notas:

El comand undo es peligroso - deshace trabajo de una manera irreversible.

Una vez que el undo termina, no hay manera posible de recuperar el estado

anterior de los ficheros y directorios afectados. Si no se especifica

ninguna ruta en los argumentos, por defecto se desharán los cambios en todos

los elementos del directorio actual, pero NO de manera recursiva.

Estos comandos son equivalentes ejecutados desde el directorio /src:

/src

|- file.txt

|- code.cs

\- /test

|- test\_a.py

\- test\_b.py

$ cm undo

$ cm undo \*

$ cm undo file.txt code.cs /test

$ cm undo .

$ cm undo /src file.txt code.cs

Si se quiere que la operación sea recursiva, se necesita especificar la

opción -r.

Para deshacer todos los cambios debajo de un directorio (incluyendo los

cambios sobre el propio directorio):

$ cm undo dirpath -r

Si dirpath es la ruta de un workspace, todos los cambios dentro del

workspace serán deshechos.

Ficheros borrados:

Para deshacer borrados de fichero o directorios es necesario especificar la

ruta completa del fichero o directorio, o el directorio padre más la opción

-r (recursivo).

Por ejemplo:

$ cm undo .

(NO deshace los borrados en el directorio actual.)

$ cm undo . -r

(Deshace todos los borrados (y otros cambios) recursivamente del directorio

actual.)

$ cm undo src/file.txt

(Deshace el borrado (u otro cambio) del fichero src/file.txt.)

Ejemplos:

$ cm undo . -r

(Deshace todos los cambios en el directorio actual de manera recursiva. Si

se ejecuta desde la raíz del workspace, deshace todos los cambios en el

workspace completo.)

$ cm co file.txt

$ cm undo file.txt

(Deshace el checkout en file.txt.)

$ echo content >> file.txt

$ cm undo file.txt

(Deshace el cambio local en file.txt.)

$ cm undo src

(Deshace los cambios en el directorio src y en todos los ficheros

controlados que contenga)

$ cm undo src/\*

(Deshace los cambios en todos los elementos contenidos en src, sin afectar

al propio directorio. Por la expansión del wildcard, es equivalente a

'cm undo src/file1.txt src/file2.txt').

$ cm undo \*.cs

(Deshace cambios en cada elemento que encaje con el patrón \*.cs en el

directorio actual.)

$ cm undo \*.cs -r

(Deshace cambios en cada elemento que encaje con el patrón \*.cs en el

directorio actual, y en cada directorio por debajo de una manera recursiva.)

$ cm co file1.txt file2.txt

$ echo content >> file1.txt

$ cm undo --unchanged

(Deshace el checkout en file2.txt por no estar modificado, ignorando

file1.txt ya que tiene cambios locales.)

$ echo content >> file1.txt

$ echo content >> file2.txt

$ cm co file1.txt

$ cm undo --checkedout

(Deshace el cambio en el fichero en checkout file1.txt, ignorando file2.txt

ya que no está en checkout.)

$ cm add file.txt

$ cm undo file.txt

(Deshace el añadido de file.txt, dejándolo de nuevo como privado.)

$ rm file1.txt

$ echo content >> file2.txt

$ cm add file3.txt

$ cm undo --deleted --added \*

(Deshace el borrado de file1.txt y el añadido de file3.txt, ignorando

el cambio en file2.txt.)

== CMD\_HELP\_LOCK\_UNLOCK ==

Ejemplos:

cm lock unlock 91961b14-3dfe-4062-8c4c-f33a81d201f5

cm lock unlock DIGITALIS:8084 2340b4fa-47aa-4d0e-bb00-0311af847865 bcb98a61-2f62-4309-9a26-e21a2685e075

Notas:

El comando utilizará el servidor especificado para realizar los desbloqueos.

Si no se especifica ningún servidor, el comando intentará obtener un

servidor válido a través del espacio de trabajo actual.

Si con los pasos anteriores no se pudo calcular ningún servidor, el comando

utilizará el servidor especificado en la configuración del cliente.

Sólo el usuario administrador del servidor de repositorios podrá ejecutar

el comando "cm unlock".

Para la especificación de los GUID's de los elementos, el formato ha de ser

de 32 dígitos separados por guiones (y opcionalmente, rodeado por llaves):

{00000000-0000-0000-0000-000000000000} ó 00000000-0000-0000-0000-000000000000

== CMD\_DESCRIPTION\_UPDATE ==

Actualiza el espacio de trabajo.

== CMD\_USAGE\_UPDATE ==

Sintaxis:

cm update [<ruta> | --last]

[--changeset=csetspec] [--cloaked] [--dontmerge] [--forced]

[--ignorechanges] [--override] [--recursewk] [--skipchangedcheck]

[--silent] [--verbose] [--xml[=output\_file]] [--encoding=name]

ruta Ruta a actualizar.

Usa . para aplicar la actualización al directorio actual.

Omita la ruta para actualizar todo el espacio de trabajo

Si no se especifica ninguna ruta, entonces se actualiza

todo el espacio de trabajo parcial.

--last Antes de la actualización, cambia el selector del

espacio de trabajo de una configuración de changeset o

de etiqueta a una configuración de rama.

El selector se cambia a la rama a la que pertenece el

changeset o la etiqueta.

Opciones:

--changeset Actualiza el espacio de trabajo a un changeset concreto.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de changesets.

--cloaked Incluye los ítems cloaked en la operación de update. Si

no se utiliza esta opción, aquellos ítems marcados como

cloaked serán ignorados en la operación.

--dontmerge En el caso en que se necesite una operación de update

merge, esta operación no se realizará.

--forced Fuerza la escritura de todas las revisiones.

--ignorechanges Ignora el mensaje de aviso de "cambios pendientes" si

existen cambios pendientes durante la actualización del

espacio de trabajo.

--override Se ignoran los ficheros cambiados fuera del control de

Plastic SCM, sobrescribiendo su contenido con el del

control de versiones.

--recursewk Actualiza todos los espacios de trabajo encontrados en

la actual ruta. Esto es útil para actualizar todos los

espacios de trabajo contenidos en la ruta especificada.

--skipchangedcheck El update comprueba si hay cambios locales en el

workspace antes de empezar. Si siempre haces co antes

de modificar un fichero, puedes usar esta opción para

acelarar la operación.

--silent No muestra ninguna salida.

--verbose Muestra información adicional.

--xml Permite volcar la salida del comando en formato xml a la

salida estándar. También se permite especificar un

fichero de salida (--xml=output.xml).

--forcedetailedprogress Fuerza mostrar progreso detallado incluso cuando se

redirige la salida estándar.

--encoding Si se utiliza con la opción --xml, permite especificar

el encoding con el que se guardará la salida del

comando. Consulte la documentación de MSDN para obtener

la tabla completa de codificaciones soportadas y su

formato (al final de la página, en la columna "Name"):

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

== CMD\_HELP\_UPDATE ==

Notas:

Cuando se trabaja con varias ramas o cuando varias personas trabajan contra

una rama, la información que tenemos en el espacio de trabajo puede quedar

desactualizada. El caso más sencillo se da cuando cambiamos de rama. Al

cambiar el selector apuntando a otra rama, debemos actualizar el espacio de

trabajo con las revisiones de esa rama.

Cuando varias personas trabajan contra una rama, crean revisiones nuevas

cada poco tiempo y es fundamental poder actualizar el espacio de trabajo

para obtener las últimas revisiones.

Para ello utilizaremos el comando update. Este comando actualiza el espacio

de trabajo con las revisiones que se le indiquen en el selector.

Solamente obtiene los ficheros necesarios. Si la revisión que hay en el

espacio de trabajo coincide con la que hay en el repositorio no se

actualiza, a no ser que se utilice la opción -–forced. Si la revisión del

espacio de trabajo es más nueva que la que hay en el repositorio tampoco

se actualiza. En este caso no nos valdrá la opción -–forced.

NOTA: El comando update siempre asume recursividad.

Cuando se especifica la opción --last, no es necesario especificar una ruta.

En ese caso, el espacio de trabajo del directorio actual es actualizado.

(Recuerde que al especificar el flag --last puede causar que el selector

del espacio de trabajo cambie a una rama si previamente apuntaba a un

changeset o una etiqueta).

Ejemplos:

cm update (actualiza el espacio de trabajo actual)

cm update . (obtiene el directorio actual y todos sus hijos)

cm update . --forced --verbose (fuerza a escibir todas las revisiones,

y muestra más información)

cm udpate --last

== CMD\_DESCRIPTION\_VERSION ==

Muestra el número de versión del cliente.

== CMD\_USAGE\_VERSION ==

Sintaxis:

cm version

== CMD\_HELP\_VERSION ==

== CMD\_DESCRIPTION\_WHOAMI ==

Muestra el usuario actual de Plastic SCM.

== CMD\_USAGE\_WHOAMI ==

Sintaxis:

cm whoami

== CMD\_HELP\_WHOAMI ==

== CMD\_USAGE\_WKTREENODESTATUS ==

Uso:

cm wktreenodestatus path1, path2, ...

== CMD\_DESCRIPTION\_WORKSPACE ==

Use este comando para administrar workspaces.

== CMD\_USAGE\_WORKSPACE ==

Sintaxis:

cm workspace | wk comando [options]

Comandos:

list | ls

create | mk

delete | rm

move | mv

rename

Para obtener más información sobre cada comando:

cm workspace comando --usage

cm workspace comando --help

== CMD\_HELP\_WORKSPACE ==

Ejemplos:

cm workspace create myWorkspace wk\_path

cm workspace list

cm workspace delete myWorkspace

== CMD\_DESCRIPTION\_WORKSPACE\_CREATE ==

Crea un nuevo workspace.

== CMD\_USAGE\_WORKSPACE\_CREATE ==

Sintaxis:

cm workspace | wk [create | mk] nombre ruta [rep\_spec] [--selector[=selector\_file]

nombre ruta --dynamic --tree=[tree]

nombre Nombre del nuevo workspace.

ruta Ruta de acceso del workspace.

rep\_spec Crea el nuevo workspace con el repositorio especificado.

Use 'cm help objectspec' para obtener más

información sobre especificaciones de repositorios.

Opciones:

--selector Edita el selector para el nuevo workspace.

Si se especifica selector-file, entonces el selector

para el nuevo workspace se fiha desde el fichero

especificado.

--dynamic Crea un workspace dinámico. Esta funcionalidad es

todavía experimental, y solo está disponible para

Windows. Si se usa esta opción, es obligatorio usar el

parámetro --tree.

--tree Especifica el punto inicial al que apuntará el

workspace dinámico. Puede ser una especificación de

rama, changeset o etiqueta. El workspace continuará

usando el repositorio de la especificación. Use

'cm help objectspec' para obtener más información sobre

la especificación.

== CMD\_HELP\_WORKSPACE\_CREATE ==

Notas:

- Un workspace es una vista del repositorio mapeado en el sistema de

ficheros local, en un directorio especificado. Utilice los comandos

'cm showselector' y 'cm setselector' para ver y modificar el selector.

- Si no se especifica rep\_spec ni --selector, el workspace cargará

automáticamente el contenido del primer repositorio (alfabéticamente) del

servidor configurado en el fichero de configuración client.conf

- Los workspaces dinámicos son una funcionalidad experimental (sólo Windows),

y requiere que el programa plasticfs.exe se esté ejecutando en segundo

plano.

Ejemplos:

cm workspace create myworkspace c:\workspace

cm wk mk myworkspace /home/john/wkspace

(Crea el workspace 'myworkspace' en 'c:\workspace' y '/home/john/wkspace'

respectivamente.)

cm wk mywktest c:\wks\wktest --selector=myselector.txt

(Crea 'mywktest' utilizando como selector 'myselector.txt'.)

cm wk mywkprj c:\wks\wkprj myrep@repserver:localhost:8084

(Crea 'mywkprj' en 'c:\wks\wkprj' con el repositorio seleccionado.)

cm wk mywkprj c:\dynwks\mywkprj --dynamic --tree=br:/main@myrep@localhost:8084

(Crea el workspace dinámico 'mywkprj' en 'c:\dynwks\mywkprj' con el

repositorio 'myrep@localhost:8084', apuntando a la rama '/main' la primera

vez que se monte.)

== CMD\_DESCRIPTION\_WORKSPACE\_DELETE ==

Borra un workspace.

== CMD\_USAGE\_WORKSPACE\_DELETE ==

Sintaxis:

cm workspace | wk delete | rm [ruta | wkspec] [--keepmetadata]

ruta Ruta del workspace.

wkspec Especificación del workspace a borrar.

Use 'cm help objectspec' para obtener más información sobre la

especificación de espacios de trabajo.

Opciones:

--keepmetadata: No elimina los archivos de metadatos del directorio

.plastic.

== CMD\_HELP\_WORKSPACE\_DELETE ==

Notas:

Si no se especifican parámetros, se borrará el workspace actual.

Ejemplos:

cm workspace delete

(Elimina el workspace actual)

cm wk delete c:\workspace

cm workspace rm /home/danipen/wks

cm wk rm wk:MiWorkspace

cm wk rm wk:MiWorkspace@DIGITALIS

== CMD\_DESCRIPTION\_WORKSPACE\_LIST ==

Muestra la lista de workspaces.

== CMD\_USAGE\_WORKSPACE\_LIST ==

Sintaxis:

cm workspace | wk [list | ls] [--format=str\_format]

Opciones:

--format Devuelve el mensaje de respuesta con el formato

especificado. Consulte la ayuda para más información.

== CMD\_HELP\_WORKSPACE\_LIST ==

Notas:

Este comando acepta una cadena de texto formateada para mostrar la

salida.

Los parámetros de salida del comando son los siguientes:

{0} | {wkname} Nombre del workspace.

{1} | {machine} Nombre de la máquina.

{2} | {path} Ruta del workspace.

{3} | {wkid} Identificador único del workspace.

{4} | {wkspec} Especificación del workspace en el formato:

'wkname@machine'.

{tab} Inserta un tabulador.

{newline} Inserta una nueva línea.

Ejemplos:

cm wk

(Lista todos los workspaces.)

cm workspace list --format={0}#{3,40}

cm workspace list --format={wkname}#{wkid,40}

(Lista todos los workspaces mostrando el nombre del workspaces, el

símbolo '#' y el identificador del workspace utilizando 40 espacios y

alineado a la izquierda.)

cm wk --format="Workspace {0} in path {2}"

cm wk --format="Workspace {wkname} in path {path}"

(Lista todos los workspaces, mostrando los resultados en cadenas

formateadas.)

== CMD\_DESCRIPTION\_WORKSPACE\_MOVE ==

Mueve un workspace.

== CMD\_USAGE\_WORKSPACE\_MOVE ==

Sintaxis:

cm workspace | wk move | mv [wkspec] ruta

Opciones:

wkspec Especificación del workspace a ser movido.

Use 'cm help objectspec' para obtener más información sobre la

especificación de espacios de trabajo.

ruta La ruta destino a la que mover el workspace.

== CMD\_HELP\_WORKSPACE\_MOVE ==

Notas:

Este comando permite al usuario mover workspace a otro lugar en disco.

Ejemplos:

cm workspace move myWorkspace \new\workspaceDirectory

(Mueve el workspace a la ruta especificada.)

cm wk mv C:\wkspaces\newpath

(Mueve el workspace de trabajo a 'C:\wkspaces\newpath')

== CMD\_DESCRIPTION\_WORKSPACE\_RENAME ==

Cambia el nombre de un workspace.

== CMD\_USAGE\_WORKSPACE\_RENAME ==

Sintaxis:

cm workspace | wk rename [nombre\_antiguo] nombre\_nuevo

nombre\_antiguo Nombre antiguo del workspace.

nombre\_nuevo Nombre nuevo del workspace.

== CMD\_HELP\_WORKSPACE\_RENAME ==

Notas:

Este comando renombra un workspace. Si no se especifica nombre\_antiguo,

el sistema tomará el nombre del workspace actual.

Ejemplos:

cm workspace rename miworkspace wk2

(El workspace 'miworkspace' se renombra a 'wk2'.)

cm wk rename nuevoNombre

(El workspace actual se renombra a 'nuevoNombre'.)

== CMD\_DESCRIPTION\_WORKSPACESTATUS ==

Muestra el changeset cargado en un workspace y si los hay los elementos cambios que contiene.

== CMD\_USAGE\_WORKSPACESTATUS ==

Sintaxis:

cm status [workspacepath] [opciones] [tipos\_de\_búsqueda]

workspacepath: El path de un workspace determinado o una ruta por debajo

del mismo. La búsqueda de cambiados se realizará a partir del path

especificado y de manera recursiva.

Opciones generales:

--changelist=NAME: Muestra los cambios en el 'changelist' llamado "NAME".

--changelists: Muestra los cambios agrupados en 'changelists'.

--cutignored: Evita mostrar los contenidos de los directorios ignorados.

Requiere el tipo de búsqueda --ignored.

--header: Solo imprimirá la información del changeset cargado en el

workspace, no realizará la búsqueda de elementos cambiados.

--noheader: El comando no imprimirá el estado del workspace, sólo

realizará e imprimirá la búsqueda de elementos cambiados.

--nomergesinfo: La información de los merges de los cambios no se imprimirá.

--head: El comando imprimirá el estado del último changeset de la rama.

--short: Muestra únicamente rutas de elementos que tienen cambios.

--symlink: Aplica la operación a los enlaces simbólicos en lugar de

aplicarla a sus destinos.

--dirwithchanges: Muestra directorios que contienen cambios (ítems

añadidos, movidos o eliminados en su interior).

--xml: Permite volcar la salida del comando en formato xml. Por defecto la

salida sale en la pantalla, pero si se especifica un fichero

(--xml=output.xml) la salida se vuelca en dicho fichero.

--encoding: si se utiliza con la opción --xml, permite especificar el

encoding con el que se guardará la salida del comando. Consulte la

documentación de la MSDN para obtener la tabla completa de

codificaciones soportadas y su formato; al final de la página en la

columna "Name":

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

--wkrootrelativepaths, --wrp: Muestra rutas relativas a la raíz del

workspace en vez de rutas relativas al directorio actual.

--fullpaths, --fp: Fuerza al comando a imprimir rutas absolutas,

sobreescribiendo cualquier otra configuración acerca de impresión

de rutas.

Opciones del viejo formato:

--cset: El comando imprimirá el estado del workspace en el viejo formato.

--compact: El comando imprimirá el estado del workspace y la búsqueda de

elementos cambiados en el viejo formato.

--noheaders: Usado junto con --compact, la información de cada grupo de

cambiós no se imprimirá. (No aplica al nuevo formato.)

Tipos de búsqueda:

--added: El comando busca elementos añadidos.

--checkout: El comando busca elementos en checkout.

--changed: El comando busca elementos cambiados.

--copied: El comando busca elementos copiados.

--replaced: El comando busca elementos reemplazados.

--deleted: El comando busca elementos borrados.

--localdeleted: El comando busca elementos borrados localmente.

--moved: El comando busca elementos movidos.

--localmoved: El comando busca elementos movidos localmente.

--percentofsimilarity (-p)=valor: Porcentaje de similitud entre dos

ficheros para considerarlos el mismo en la búsqueda de movidos

locales, su valor por defecto es un 20%.

--txtsameext: Sólo aquellos ficheros de texto que tengan la misma

extensión serán tenidos en cuenta para la búsqueda de movidos basada

en la similitud del contenido. Por defecto todos los ficheros de texto

son procesados.

--binanyext: Cualquier fichero binario es tenido en cuenta para la

búsqueda de movidos basada en la similitud del contenido. Por defecto

sólo aquellos ficheros binarios que tengan la misma extensión

son procesados.

--private: El comando busca elementos no controlados.

--ignored: El comando busca elementos ignorados.

--hiddenchanged: El comando busca elementos cambiados marcados como

ocultos (incluye '--changed').

--cloaked: El comando busca elementos marcados como cloaked.

--controlledchanged: Este modificador engloba los siguientes parámetros:

'--added' '--checkout' '--copied' '--replaced' '--deleted' '--moved'.

--all (-a): Este modificador engloba los siguientes parámetros:

'--controlledchanged' '--changed' '--localdeleted' '--localmoved'

'--private'.

== CMD\_HELP\_WORKSPACESTATUS ==

Notas:

El comando status muestra el changeset cargado en un workspace

y si los hay, los cambios que contiene.

Se puede utilizar el comando para saber cuál es el changeset con el que se

está trabajando en un momento dado, y poder utilizarlo, por ejemplo, para

marcar configuraciones concretas mediante etiquetas de tipo changeset.

También es útil para conocer los cambios pendientes que existen dentro

del workspace, los tipos de cambios que deseamos obtener del workspace pueden

ser modificados gracias a los parámetros que el comando admite. Por defecto

se muestran todos los cambios, controlados y locales.

El porcentaje de similitud entre dos ficheros usado por el algoritmo de

cálculo de movidos locales se puede especicifar mediante la option '--percentofsimilarity'

o '-p' y su rango va desde 0 hasta 100. Si este parámetro no es especidicado

su valor por por defecto es del 20%.

A través de éste comando, también es posible mostrar los cambios agrupados

en 'changelists' de cliente. El changelist por defecto del sistema (llamado

'default'), agrupa los cambios que no están contenidos en el resto de

'changelists', por lo que los cambios que se muestran especificando éste

'changelist', dependerán del tipo de búsquedas especificados en el comando.

La opción de mostrar cambios agrupados en 'changelists' requiere mostrar cambios

controlados (en estado 'added', 'checkout', 'copied', 'replaced', 'deleted' o

'moved'), por lo que la opción '--controlledchanged' se activará automáticamente

cuando se muestren 'changelists'.

El formato por defecto para la salida en XML es UTF-8. Es posible cambiar

dicho encoding usando la opción --encoding=otro\_encoding.

El comando 'status' imprime por defecto rutas relativas al directorio de trabajo,

a menos que se especifiquen las opciones '--machinereadable' o '--short'

(en cuyo caso, se imprimen rutas absolutas).

Si se usa la opción '--xml', el comando 'status' imprimirá por defecto rutas

relativas a la raíz del workspace, a menos que se especifique adicionalmente

la opción '--fp' (en cuyo caso, se imprimen rutas absolutas).

Ejemplos:

cm status

(Ejecutándolo dentro de un workspace el comando imprime el changeset cargado

y los elementos en checkout, añadidos, copiados, reemplazados, borrados, borrados locales,

movidos, movidos locales, y privados. La busqueda de los elementos se realiza

sobre el workspace de forma recursiva)

cm status --controlledchanged

(Ejecutándolo dentro de un workspace el comando imprime el changeset cargado

y los elementos del workspace en checkout, añadidos, copiados, reemplazados,

borrados, y movidos)

cm status --added

(Ejecutándolo dentro de workspace el comando imprime el changeset cargado

y solamente los elementos añadidos. La búsqueda de los elementos se realiza

sobre el workspace de forma recursiva)

cm status c:\workspaceLocation\code\client --added

(El comando imprime el changeset cargado y solamente los elementos añadidos.

La búsqueda de los elementos se realiza sobre el path proporcionado de forma recursiva)

cm status --changelists

cm status --changelist

(El comando muestra todos los cambios en el espacio de trabajo, agrupados

por 'changelists').

cm status --changelist=pending\_to\_review

(El comando muestra los cambios que contiene el changelist 'pending\_to\_review')

cm status --changelist=default --private

(El comando muestra los cambios que contiene el changelist 'default',

el changelist por defecto del sistema, mostrando elementos privados,

además de elementos con cambios controlados si los hubiese).

cm status --short --changelist=pending\_to\_review | cm checkin -

(Se protegen los cambios en el changelist llamado 'pending\_to\_review').

cm status c:\workspaceLocation\code\client --xml=output.xml

(Guarda la salida del status en el fichero output.xml, en formato XML y

usando UTF-8.)

cm status --ignored

(Muestra todos los elementos ignorados)

Salida:

IG src

IG src/version.c

IG out/app.exe

cm status --ignored --cutignored

(Muestra ficheros ignorados cuyo directorio padre no está ignorado y

directorios ignorados pero no el árbol de sus contenidos)

Salida:

IG src

IG out/app.exe

== CMD\_DESCRIPTION\_XLINK ==

Crea, edita o muestra un xlink

== CMD\_USAGE\_XLINK ==

Sintaxis:

cm xlink [-w] [-rs] xlink\_path / <csetspec | lbspec | brspec> [<expansion\_rules>+]

(Crea un xlink).

cm xlink [-rs] xlink\_path /relative\_path <csetspec | lbspec | brspec> [<expansion\_rules>+]

(Crea un xlink parcial de solo lectura apuntando a /relative\_path en lugar

de apuntar a la raíz por defecto / ).

cm xlink -e xlink\_path <csetspec | lbspec | brspec>

(Edita un xlink para cambiar la especificación destino).

cm xlink -s|--show xlink\_path

(Muestra la información de un xlink incluyendo las reglas de expansión).

cm xlink -ar|--addrules xlink\_path <expansion\_rules>+

(Añade al xlink las reglas de expansión proporcionadas).

cm xlink -dr|--deleterules xlink\_path <expansion\_rules>+

(Borra del xlink las reglas de expansión especificadas).

xlink\_path Ruta en el espacio de trabajo actual donde se cargan

los contenidos del repositorio que se va a enlazar

(cuando se crea un xlink) o que ya está enlazado

(cuando se edita un xlink).

csetspec Especificación del changeset que se va a enlazar.

Determina qué versión y rama se carga en el espacio de

trabajo para el repositorio enlazado.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de changesets.

lbspec Especificación de la etiqueta que se va a enlazar.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de etiquetas.

brspec Especificación de la rama que se va a enlazar.

Usa el changeset actual a donde la rama especificada

está apuntando.

Usa 'cm help objectspec' para obtener más información

sobre la especificación de ramas.

-e Indica que el comando va a editar un xlink existente

para cambiar la especificación del changeset enlazado.

-s | --show Muestra información sobre el xlink seleccionado.

-ar | --addrules Añade una o más reglas de expansión al xlink indicado.

-dr | --deleterules Borra una o más reglas de expansión del xlink indicado.

expansion\_rules Especifica una o más reglas de expansión. Cada regla de

expansión es un par (rama)-(rama a enlazar)

br:/main/fix-br:/main/develop/fix

Opciones:

-w Crea un xlink de escritura. Esto significa que los

contenidos bajo el xlink se podrán editar mediante el

uso de autoexpansión de ramas.

-rs Relative server. Permite crear un xlink a un repositorio

que es una réplica de otro que existe en un servidor

central. De este modo se reconcilian automáticamente los

repositorios replicados con el original.

Ejemplos:

cm xlink codigo\segundorepositorio / 1@segundo@localhost:8084

cm xlink codigo\segundorepositorio / lb:LB001@segundo@localhost:8084

cm xlink -s codigo\segundorepositorio

cm xlink -ar codigo\segundorepositorio br:/main-br:/main/develop br:/main/fix-br:/main/develop/fix

cm xlink -dr codigo\segundorepositorio br:/main/fix-br:/main/develop/fix

== CMD\_HELP\_XLINK ==

Notas:

Este comando crea un xlink al changeset especificado. Por defecto se

creará como un xlink de sólo lectura. Esto significa que los contenidos

correspondientes a otro repositorio cargados por debajo del xlink no

podrán ser modificados.

Para la edición del changeset de un xlink se puede utilizar una sintaxis

reducida en la que no hay que especificar el resto de parámetros del

xlink. Solamente es necesario en nuevo changeset al que apuntará.

Auto-expansión de ramas:

Cuando se realiza un cambio en uno de los repositorios enlazados en modo

de lectura y escritura (opción -e) es necesario crear una nueva rama en

dicho repositorio basándose en la rama en la que se está trabajando en el

repositorio principal. Para determinar el nombre de dicha rama, se aplican

las siguientes reglas:

1) Se comprueba si el nombre completo de la rama existe en el

repositorio enlazado:

- Si existe, se utiliza dicha rama.

- Si no existe, Plastic construye el nombre de la rama de esta forma:

- Nombre de la rama que contiene el changeset enlazado + nombre

de la rama del repositorio principal.

- Si existe una rama con dicho nombre, se utiliza como rama de

trabajo.

- Si no, se crea una rama nueva con el nombre así construido y

se establece como base de la rama el changeset enlazado.

2) Después, se crea una nueva versión del xlink dentro de la nueva rama

apuntando al nuevo changeset.

De este modo, la estructura completa del xlink se mantiene actualizada con

los últimos cambios en las versiones correctas.

Ejemplos:

cm xlink code\firstrepo / 1@first@localhost:8084

(Crea un xlink en la carpeta 'firstrepo' en el espacio de trabajo actual

donde el changeset '1' en el repositorio 'first' se enlazará).

cm xlink opengl\include /includes/opengl 1627@includes@localhost:8087

(Crea un xlink parcial de solo lectura en la ruta 'opengl\include' en el

espacio de trabajo actual donde la ruta '/includes/opengl' en el changeset

'1627' del repositorio 'includes' se enlazará como raíz. Esto significa

que cualquier ítem dentro de '/includes/opengl' se enlazará en

'opengl\include' mientras que el resto del repositorio se ignorará).

cm xlink -w -rs code\secondrepo / lb:LB001@second@localhost:8084

(Crea un xlink de escritura y relativo en la ruta 'secondrepo' en el

espacio de trabajo actual donde la etiqueta'LB001' en el repositorio

'second' se enlazará).

cm xlink code\thirdrepo / 3@third@localhost:8087 br:/main-br:/main/scm003

(Crea un xlink en la ruta 'thirdrepo' dentro del espacio de trabajo actual

donde el changeset '3' en el repositorio 'third' se enlazará).

cm xlink -e code\secondrepo br:/main/task1234@second@localhost:8084

(Edita el xlink 'code\secondrepo' para cambiar el repositorio destino

enlazando la rama 'main/task1234' en el repositorio 'second').

cm xlink --show code\thirdrepo

(Muestra información del xlink 'code\thirdrepo' incluyendo las reglas de

expansión si éstas existen).

cm xlink -ar code\secondrepo br:/main-br:/main/develop br:/main/fix-br:/main/develop/fix

(Añade dos reglas de expansión al xlink 'code\secondrepo').

cm xlink -dr code\secondrepo br:/main/fix-br:/main/develop/fix

(Borra la regla de expansión del xlink 'code\secondrepo').

cm xlink code\secondrepo / 1@second@localhost:8084

cm xlink code\secondrepo / lb:LB001@second@localhost:8084

== CMD\_USAGE\_AUTOCOMPLETE ==

Sintaxis:

cm autocomplete install

Instala el autocompletado de comandos para 'cm' en la shell.

cm autocomplete uninstall

Desinstala el autocompletado de comandos para 'cm' de la shell.

cm autocomplete --line <shell\_line> --position <cursor\_position>

Devuelve sugerencias de autocompletado para 'shell\_line', que deben ser

insertadas en 'cursor\_position'. Este comando no está pensado para ser

ejecutado por el usuario final, pero está documentado en caso de que se

quiera extender el soporte de autocompletado a otras shells.

shell\_line La linea que el usuario ha escrito en la shell cuando se

pidió el autocompletado.

En Bash, encuentra en la variable de entorno COMP\_LINE.

En PowerShell, se encuentra en la variable $wordToComplete.

cursor\_position La posición del cursor cuando se pidió el autocompletado.

En Bash, se encuentra en la variable de entorno COMP\_POINT.

En PowerShell, se encuentra en la variable $cursorPoint.