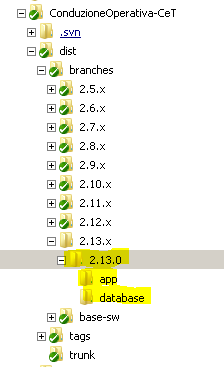
**Guida al rilascio di una release**

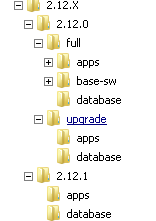
**del software C&T**

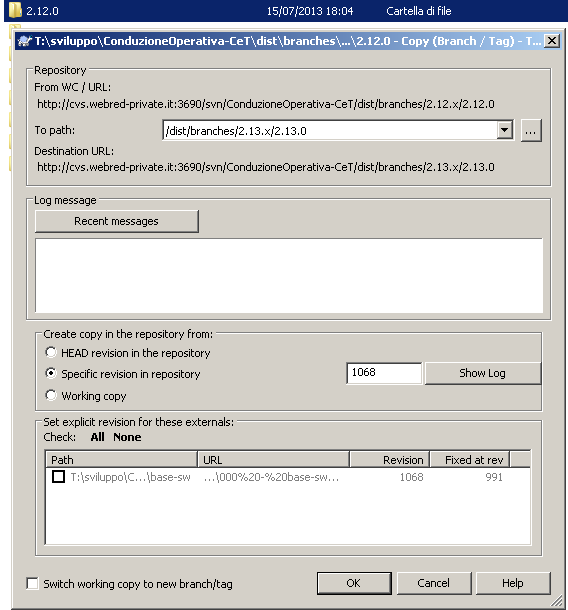
1. creazione delle folder di rilascio nel repository svn ConduzioneOperativa-CeT (in branches)



Se il rilascio è di tipo x.y.0 allora la folder avrà una struttura comprendente anche una cartella base-sw che andremo dopo a collegare al branch del software di base.

Altrimenti , per un rilascio x.y.z (a partire dunque da modifiche al branch n.m.x ) la struttura delle folder sarà più semplice.





Per poi scaricare in locale il nuovo branch creato e operare una sovrascrittura dei moduli e dei file necessari, in sintesi

* scaricare il branch appena creato
* cancellare full\database\GIT.dmp

1. Verifica delle variazioni alla banca dati e aggiornamento degli schemi di riferimento
   1. Collegamento a RE\_GIT , schema di riferimento RE
      1. Verifica dei nuovi comandi inseriti dalla release precedente attraverso esecuzione della stored procedure RE\_GIT.AGGIORNA\_DB\_RE\_RIF.

BEGIN

RE\_GIT.AGGIORNA\_DB\_RE\_RIF;

COMMIT;

END;

* + 1. Controllo del dbms\_output e copia nella clipboard
    2. Esecuzione degli sql nel database RE\_GIT per l’inserimento dei nuovi dati di configurazione
  1. Collegamento a AM\_GIT, schema di riferimento AM
     1. Esecuzione della procedura AM\_GIT.AGGIORNA\_DB\_AM\_RIF : rieseguire più volte fino a quando non ci sono insert da eseguire tranne le due finali e commentate.
     2. Eseguire le due insert finali del dbms\_output
  2. Verifica delle variazioni agli schemi di sviluppo rispetto agli schemi di riferimento AM\_GIT, RE\_GIT, ARGO\_GIT, ARGOBI\_GIT, SOCIOSAN\_REPO\_GIT, CS\_SISO\_GIT [[1]](#footnote-1) , DIOGENE\_GIT, VIRGILIO\_GIT, DBTOTALE\_GIT, GIT\_LAND\_GIT confrontando i rispettivi schemi attraverso uno strumento di comparazione oracle.
  3. Esecuzione degli script di riallineamento prodotti dal punto precedente.
     1. Editing degli script in modo da inserire su ogni dml il prefisso dinamico dello schema oracle(&DIOGENEGIT..,&AMGIT..,&DBTOTALEGIT.. ecc) in modo da poter rilasciare uno script generico valido per ogni nome utente oracle. Inserire all’inizio dello script sql la parola ACCEPT seguita dal nome dello schema: "**ACCEPT DIOGENEGIT;"**.
     2. Se necessario passare come parametro il codice belfiore dell'ente, scrivere "**ACCEPT ENTE;**".
     3. Esecuzione degli script negli schemi di riferimento al fine di allinearli all’ultima rel del database
  4. Salvataggio script nella folder di rilascio
     1. Salvataggio degli script nella sottocartella “database” della versione in rilascio : la nomenclatura dovrà essere:  
        n – GLOBALE.sql (es. 1 – GLOBALE.sql) per gli script degli schemi AM e RE e GIT\_LAND

n – ARGO.sql (es. 1-ARGO.sql) per gli schemi generali come ARGO, ARGOBI, SOCIOSAN\_REPO (che fanno riferimento a più installazioni JBoss )

n – ENTE.sql (es. 1 – ente.SQL) per gli altri schemi

n – CSOCIALE.sql (es. 1 – CSOCIALE.SQL) per gli script relativi alla cartella sociale

* + 1. Inserimento di uno script denominato 99 - VERSIONE\_PIATTAFORMA.sql con all’interno una update contenente il numero di versione in rilascio:

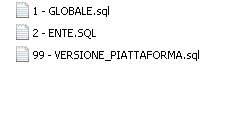
ACCEPT AMGIT;

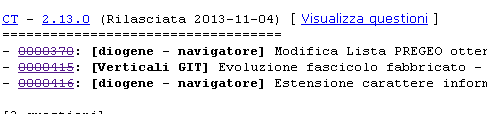
DELETE FROM &AMGIT..AM\_KEY\_VALUE\_EXT WHERE KEY\_CONF = 'versione.piattaforma';

INSERT INTO &AMGIT..AM\_KEY\_VALUE\_EXT (KEY\_CONF, VALUE\_CONF, SECTION\_NAME, ID)

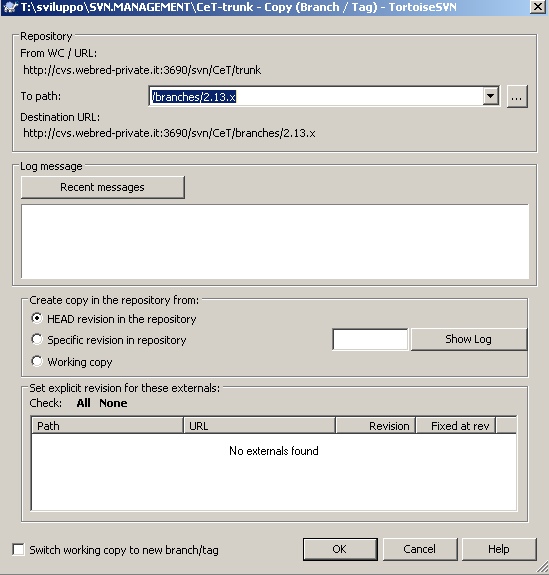
VALUES ('versione.piattaforma', '2.13.0' , 'param.globali', (SELECT nvl(MAX(ID),0)+1 FROM &AMGIT..AM\_KEY\_VALUE\_EXT));

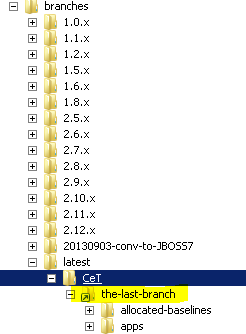
COMMIT;

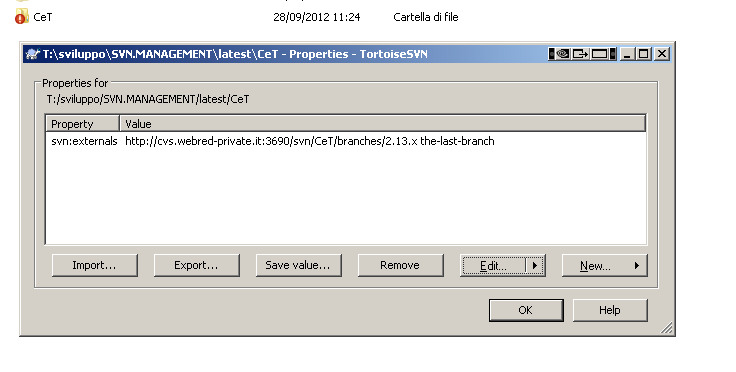


* 1. Verifica nel registro delle questioni della presenza di eventuali script sql allegati alle questioni in rilascio. Esecuzione , se necessario, degli script nei db di riferimento e salvataggio nel folder di rilascio.

1. STOP delle commit sulla baseline da testare!
2. Testing delle questioni da rilasciare in un ambiente aggiornato secondo le checklist di test allegate alle stesse
3. Branching regole (solo nel caso di rilascio x.y) del sorgente applicativo secondo le seguenti con baseline di partenza “trunk”:
   1. Checkout in locale del trunk del repository CeT
   2. Creazione del branch nel repository del sorgente

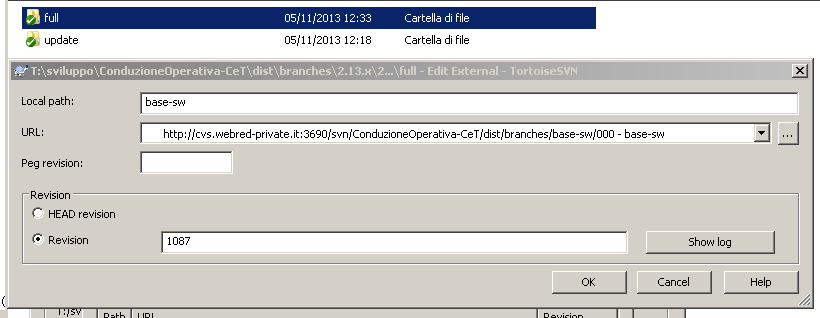


* 1. Switch del branch “latest” sul nuovo branch creato (“latest” è un branch contenente come risorsa esterna l’ultimo branch rilasciato )

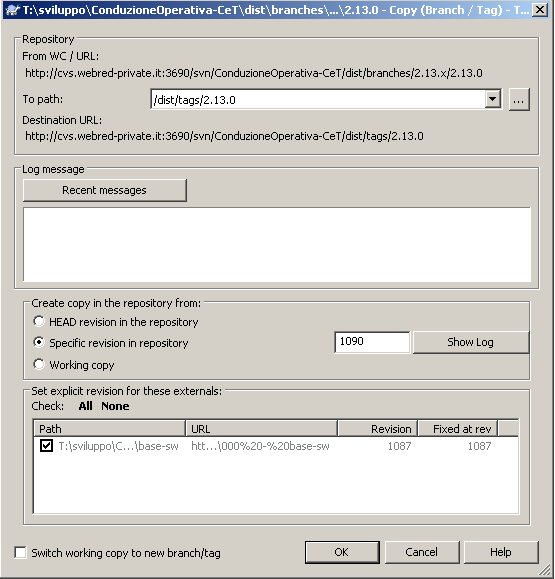


Dopo il setting della property svn:externals nella cartella CeT del branch “latest” ricordarsi di fare commit per rendere la modifica effettiva!!!

1. Eseguire l’update della cartella “latest” nel workspace locale per realizzare in locale lo switch al nuovo branch.
2. Esportare dal workspace al punto precedente i moduli applicativi ear, war ecc. , mettere i moduli modificati dalla rel precedente nella cartella update/app. nel caso di rilascio x.y.0 mettere tutti i moduli in full/app.
3. Esportare il database di riferimento (solo per release x.y.0) e mettere il file nella cartella di distribuzione x.y.0/full/database
4. Aggiornamento del branch base-sw con tutti gli item documentali e di configurazione scaturiti dalle attività di sviluppo
5. Collegare la cartella full\base-sw (se rilascio x.y.0) del rilascio il reference (svn:external) al branch “base-sw\00 – base-sw”, l’external farà riferimento alla reviasion di base-sw al momento della creazione del riferimento.



1. Creare il documento VDD.doc nella root della folder di rilascio
2. Dalla cartella del rilascio branches/x.y.z creare un tag nel repository tags/x.y.z



1. Inviare comunicazione email dell’avvenuto rilascio nel repository della nuova release.

1. Gestione dei rilasci separata CS\_GIT è lo schema di riferimento per SmartWelfare, CS\_SISO\_GIT per Siso [↑](#footnote-ref-1)