Apache Subversion (SVN)

Michel Meynard

UM2

Univ. Montpellier 2

Table des matières

Introduction

- Apache Subversion (SVN) est un gestionnaire de versions successeur de Concurrent Version System (CVS)
- logiciel libre sous licence Apache/BSD développé par CollabNet Inc. depuis 2000
- Un logiciel de gestion de versions permet de stocker une arborescence de fichiers (projet) en :
  - conservant la chronologie de toutes les modifications qui ont été effectuées
  - permettant de retrouver les différentes versions d'un lot de fichiers connexes
- l'architecture d'un gestionnaire de versions peut être locale (SCCS), ou client-serveur (CVS, **SVN**), ou distribuée (Git, SVK)

Avantages d'un gestionnaire de versions distant

- travail collaboratif sur le même projet ;
- gestion des conflits de révisions ;
- sauvegarde distante et locale du projet (même si un seul développeur) ;
- récupération possible après des modifications désastreuses !

Numérotation des versions

Généralement, un numéro de version est composé d'une suite de (souvent 3 : majeur, mineur et micro (major, minor, micro)) nombres séparés par des points.

- les nombres sont ordonnés du plus significatif au moins significatif
- le premier nombre correspond à une refonte (relative) du logiciel
- le second indique une évolution **majeure** (nouvelles fonctionnalités)
- le dernier correspond à une évolution **mineure** : corrections de bugs ou fonctionnalités secondaires (e.g. ajout d'un bouton de raccourci)

Une version nommée "2.5.21" peut avoir le sens suivant :

- 2ème version publiée
- 5ème ajout de fonctionnalité dans la version 2
- 21ème révision de la version 2.5.

Versions et SVN

- première version fonctionnelle : version 1.0
- versions notées 0.x ou 0.x.x indiquent une version bêta (non aboutie)
- Un dépôt (repository) local ou distant est géré par le **serveur svn**
- un **client svn** permet de récupérer (checkout) la version la plus récente ou de valider (commit) ses propres modifications
- les commandes SVN sont réalisées **côté client** afin de réaliser une action en lien avec le serveur
- on peut utiliser le client svn en ligne de commande : `svn commit`
- `Tortoise SVN` est un client graphique pour windows qui s'intègre à l'explorateur de fichier
- `Subclipse` est un plugin `Eclipse` pour intégrer SVN à l'Environnement de Développement Intégré

Vocabulaire et spécificités

- Le **dépôt** est un serveur d'arborescence de fichiers se souvenant de toutes les modifications apportées ;
- un **commit**, ou publication des modifications est **atomique** grâce à des transactions (Berkeley DB) côté serveur ;
- SVN permet le renommage et le déplacement de fichiers sans en perdre l'historique ;
- les numéros de révision sont **globaux** (pour l'ensemble du dépôt) et non pas par fichier (CVS) : chaque patch a un numéro de révision unique, quels que soient les fichiers touchés.
- les répertoires et métadonnées sont versionnés.
- chaque action `update` ou `commit` est indépendante, il n'y a pas de d'action pousser-tirer atomique ;

Modèle copier-modifier-fusionner

- Un modèle simpliste de gestion de versions consiste à verrouiller après la lecture d'un fichier par un utilisateur puis à déverrouiller après l'écriture !
- Cela réduit la concurrence d'accès ;
- Svn gère les conflits d'écriture par **fusion** :
  1. A et B lisent la même version 13 d'un fichier F l'un après l'autre ;
  2. A modifie sa version locale et publie (commit) sur le dépôt la version 14 ;
  3. B modifie sa version locale et tente de publier mais ce n'est pas possible, aussi le dépôt lui retourne un nouveau fichier incluant les modifications de A (14) mais aussi les siennes ; ainsi B publiera la version 15.
- Svn fournit quand même le verrouillage pour les fichiers impossibles à fusionner (binaires).

URL d'Accès au dépôt

L'emplacement d'un dépôt est toujours une URL correspondant à différentes méthodes d'accès :

- `file:///X:/var/svn/depot` : accès direct au dépôt (sur un disque local Windows) ;
- `http://projetXX.googlecode.com/svn/trunk/` : accès à un serveur Apache configuré pour Subversion ;
- `https://...` : identique à http avec chiffrement SSL ;
- `svn://...` : accès à un serveur svnserve ;
- `svn+ssh://...` : accès à un serveur svnserve à travers un tunnel SSH ;

Créer un projet dans un dépôt

Après l'installation et la configuration du serveur, il faut lui confier l'arborescence initiale :

```
svn import [CHEMIN] URL
```

- le chemin est optionel (répertoire courant par défaut) ;
- l'arborescence complète va être ajoutée (répertoires et fichiers) ;
- option : `-m "importation initiale"` ; permet de conserver un message sur la version initiale ;
- on peut importer un nouveau répertoire dans un répertoire du dépôt ;
- après importation, le répertoire local n'est pas sous contrôle de SVN : il faudra un `checkout` qui créera les `.svn` !

Exemple :

```
svn import https://tccp.googlecode.com/svn/trunk/ -m "importation
  initiale" --username meynard.michel@gmail.com
svn checkout https://tccp.googlecode.com/svn/trunk/
```


Lire une copie de travail

Pour recopier localement l'arborescence courante du projet, il suffit de réaliser un `checkout` (extraction) depuis le répertoire local où vous désirez installer votre copie :

```
mkdir exemplesvn
cd exemplesvn/
svn checkout https://tccp.googlecode.com/svn/trunk/
```

Cette copie dans `exemplesvn/trunk` crée un répertoire caché `.svn` qui contient de nombreuses informations d'administration.

Authentification et options

Authentification :

- Certaines opérations Subversion nécessitent une authentification : `import`, `commit`, ... ;
- La mise en cache du nom et du mot de passe sont effectués automatiquement côté client (`~/.subversion/auth/`) ;
- la première fois, il faut utiliser l'option `--username jdupont` ;
- Pour le serveur `Google Code`, il y a un mot de passe unique pour tous les contributeurs ;

Messages :

- Certaines commandes nécessite un message d'information qui restera stocké sur le dépôt dans le journal de propagation ;
- ce msg peut être saisi :
  - soit en option de la ligne de cmd : `-m "importation initiale"` ;
  - soit avec un éditeur désigné dans la variable d'environnement `EDITOR` : `export EDITOR=emacs` ; (`SVN_EDITOR`)

Ajouter un nouveau fichier

- Après l'édition d'un **nouveau** fichier `toto.c` :

  `svn add toto.c`
- Vous pouvez modifier d'autres fichiers du projet ;
- Pour **valider** vos modifications sur le serveur, il faut faire un `commit` :

  `svn commit -m "validation de toto et autre"`

Etats d'un fichier local

Contenues dans le répertoire `.svn`, pour chaque fichier :

- le numéro de révision de travail du fichier : le dépôt a un numéro de révision **unique** mais localement chaque fichier a son **propre** numéro ;
- la date et l'heure de la dernière mise à jour de la copie locale depuis le dépôt ;

Ainsi, chaque fichier se trouve dans l'un des états suivants :

- Inchangé et à jour ;
- Modifié localement et à jour : un appel à `svn commit` sur le fichier permettra de publier vos modifications, un appel à `svn update` ne fera rien ;
- Inchangé et périmé : un appel à `svn commit` ne fera rien et un appel à `svn update` incorporera les dernières modifications ;
- Modifié localement et périmé : un appel à `svn commit` échoue (out-of-date) ; un appel à `svn update` va tenter de fusionner.

Commandes de base

- `svn help commit` : aide sur la commande `commit` ;
- `svn import [CHEMIN] URL -m "Import initial"` : importation dans le dépôt du projet initial ou d'un répertoire (non encore versionné) ;
- `svn list` : liste des fichiers du dépôt ;
- `svn info` : informations sur la révision l'URL du dépôt, ... ;
- `svn status [CHEMIN...]` : affiche l'état des fichiers et des répertoires de la copie de travail ; A (add), D (del), M (modif), C (conflit), I (Ignore) ;
- `svn checkout URL [-r 2] [CHEMIN]` : extraction de fichiers d'une certaine version (option -r) ;
- `svn commit [CHEMIN] -m "msg"` : propagation des modifs locales vers le dépôt ;
- `svn add CHEMIN...` ajoute un ou des nouveaux fichiers ;
- `svn del CHEMIN...` supprime les fichiers (ou `rm`) ;
- `svn move SRC... DST` déplace des fichiers ;

Commandes de base - suite

- `svn update [CHEMIN]` : met à jour la copie locale grâce à la version la plus récente du dépôt ; Indique chaque modification par :
  - A Ajouté ;
  - D supprimé (*deleted*) ;
  - U mis à jour (*updated*) ;
  - G Fusionné (*Merged*) ;
  - C en Conflit ;
- `diff [-c M | -r N[:M]] [CIBLE[@REV]...]` : liste les différences entre deux révisions ou chemin ;
- `svn export [-r N] URL [CHEMIN]` : crée une copie non versionnée d'une arborescence ;
- `svn log [CHEMIN]` : affiche les entrées du journal de propagation ;
- `svn lock CIBLE...` : verrouille des fichiers afin qu'aucun autre utilisateur ne puisse propager (commit) des modifications les concernant ;
- `svn unlock CIBLE...` : déverrouille

Résoudre les conflits (fusionner des modifications)

Après mise à jour, les fichiers marqués G (merGed) ont été fusionnés automatiquement. La **résolution interactive** des conflits vous permet d'éditer le fichier **marqué** sur les différences (<=>)

```
debut commun
<<<<<<< .mine
mes modifs personnelles
=======
la version 2 du depot
>>>>>>> .r2
suite commune
```

Après décision avec l'autre contributeur, il faut supprimer les marqueurs puis :

```
$svn resolve --accept working fic.c
Conflit sur 'fic.c' résolu
$ svn commit -m "fic.c OK après résol conflit"
```


Gestion des branches

Une **branche** est une ligne de développement qui existe indépendamment d'une autre ligne, mais partage cependant une histoire commune avec elle (fichiers communs). Plusieurs branches parallèles peuvent être gérées par Subversion.

- la ligne principale s'appelle **trunk** ;
- pour créer une branche, il suffit de copier le tronc :

```
svn copy  https://tccp.googlecode.com/svn/trunk/ \
          https://tccp.googlecode.com/svn/branches/ma-branche \
      -m "Création d'une branche privée à partir du tronc."
```

- la copie est peu couteuse (temps constant car lien symbolique) ;
- un checkout pour extraire cette nouvelle branche !

```
svn checkout https://tccp.googlecode.com/svn/branches/ma-branche
/calc/trunk
```

La fusion (`svn merge`) de branches est un sujet pointu ... (voir doc.)

Références

Le site officiel `http://subversion.apache.org`
svnbook `http://svnbook.red-bean.com`
Wikipedia `http://fr.wikipedia.org/wiki/Apache_Subversion`
Tortoise `http://tortoisesvn.tigris.org/`
subclipse `http://subclipse.tigris.org/`