Les Framework Java

SubVersion Network

Claude Duvallet

Université du Havre
UFR Sciences et Techniques
25 rue Philippe Lebon - BP 540
76058 LE HAVRE CEDEX
Claude.Duvallet@gmail.com
http://litis.univ-lehavre.fr/∼duvallet/

SVN

Présentation de SVN

- Subversion (en abrégé svn) est un système de gestion de versions, distribué sous licence Apache et BSD.
- Il a été conçu pour remplacer CVS.
- Ses auteurs s'appuient volontairement sur les mêmes concepts (notamment sur le principe du dépôt centralisé et unique) et considèrent que le modèle de CVS est le bon, et que seule son implémentation est en cause.
- Le projet a été lancé en février 2000 par CollabNet, avec l'embauche par Jim Blandy de Karl Fogel, qui travaillait déjà sur un nouveau gestionnaire de version.

Les apports de Subversion : point de vue global

Subversion a été écrit afin de combler certains manques de CVS :

- Les commits, ou publications des modifications sont atomiques.
- Un serveur Subversion utilise de façon sous-jacente une base de données capable de gérer les transactions atomiques (le plus souvent Berkeley DB).
- Subversion permet le renommage et le déplacement de fichiers ou de répertoires sans en perdre l'historique.
- Les métadonnées sont versionnées : on peut attacher des propriétés, comme les permissions, à un fichier, par exemple.

Les apports de Subversion : point de vue utilisateur

Du point de vue du simple utilisateur, les principaux changements lors du passage à Subversion, sont :

- Les numéros de révision sont désormais globaux (pour l'ensemble du dépôt) et non plus par fichier : chaque patch a un numéro de révision unique, quels que soient les fichiers touchés.
- Il devient simple de se souvenir d'une version particulière d'un projet, en ne retenant qu'un seul numéro.
- `svn rename` (ou `svn move`) permet de renommer (ou déplacer) un fichier.
- Les répertoires et méta-données sont versionnés.

Troncs, Branches et Tags (1/4)

- Les notions de tronc, de branches et de tags sont assez spécifiques aux logiciels de contrôle de versions.
- C'est ce qui explique que les arborescences des répertoires de projet contiennent souvent comme premier niveau de sous-répertoires les dossiers trunk, branches et tags.
- En général, on définit par « tronc » la version centrale du programme, le développement principal « officiel ».
- Une « branche » est en général créée lorsqu'un développement « secondaire » est mis en route :
  - Soit pour ajouter une nouvelle fonctionnalité.
  - Soit parce que certains développeurs souhaitent essayer de prendre une autre direction pour certains aspects du développement.
  - Une branche peut, au bout d'un certain temps, soit être à nouveau fusionnée dans le « tronc », soit disparaître, soit donner lieu à un nouveau programme.

Troncs, Branches et Tags (2/4)

- La notion de tags correspond en partie à celle de release :
  - C'est à dire de marquage d'une certaine révision du projet comme composant une version du projet.
  - Une fois que le développement a atteint une certaine stabilité, on pourra par exemple créer un tag pour marquer la sortie de la version 1.0.
  - Ceci permettra de revenir facilement à cette version, indépendamment du numéro de révision sous-jacent correspondant.
- On peut juste citer que la création de branches ou de tags ne sont en fait que des copies créées par la commande svn copy.
- La commande svn switch, elle, permet de faire passer la copie de travail d'une branche à une autre.

Troncs, Branches et Tags (3/4)

- Une des particularités de Subversion est qu'il ne fait aucune distinction entre un label, une branche et un répertoire.
- C'est une simple convention de nommage pour ses utilisateurs.
- Il devient ainsi très facile de comparer un label et une branche ou autre croisement.
- Quel que soit le système de gestion de versions, les numéros de révision à plusieurs chiffres sont difficiles à mémoriser.
- Pour cette raison de nombreux systèmes laissent l'utilisateur définir des tags comme des synonymes plus faciles à retenir.
- Mais ce que Subversion recommande d'utiliser comme tag est d'une nature complètement différente : une fois la commande svn copy effectuée, un tag Subversion ne se rappelle absolument plus de quel numéro de révision il provient.

Troncs, Branches et Tags (4/4)

- Alors que les tags des autres systèmes sont des points dans le temps, Subversion recommande de définir les tags comme des points dans l'espace d'un système de fichiers.
- Cette absence de "synonyme vers un numéro de révision" rend certaines opérations un peu moins pratiques dans Subversion.
- Par exemple, retrouver ce qui a changé d'un tag à l'autre dans un fichier est un petit peu plus compliqué que de lancer une simple commande : `svn diff -r tag1:tag2` monfichier dans le répertoire de travail.
- D'autres opérations deviennent impossibles : par exemple une commande telle que `svn log -r tag1:tag2 monfichier` ne fonctionne pas et il n'y a pas d'alternative qui fonctionne.

Les principales commandes (1/3)

- **add :** Déclare l'ajout d'une nouvelle ressource pour le prochain commit.
- **blame :** Permet de savoir quel contributeur a soumis les lignes d'un fichier.
- **checkout (co) :** Récupère en local une révision ainsi que ses méta-données depuis le dépôt.
- **cleanup :** Nettoie la copie locale pour la remettre dans un état stable.
- **commit (ci) :** Enregistre les modifications locales dans le dépôt créant ainsi une nouvelle révision.
- **copy :** Copie des ressources à un autre emplacement (localement ou dans le dépôt).
- **delete :** Déclare la suppression d'une ressource existante pour le prochain commit (ou supprime directement une ressource du dépôt).

Les principales commandes (2/3)

- **diff :** Calcule la différence entre deux révisions (permet de créer un patch à appliquer sur une copie locale).
- **export :** Récupère une version sans métadonnées depuis le dépôt ou la copie locale.
- **import :** Envoie une arborescence locale vers le dépôt.
- **info :** Donne les informations sur l'origine de la copie locale.
- **lock :** Verrouille un fichier.
- **log :** Donne les messages de commit d'une ressource.
- **merge :** Calcule la différence entre deux versions et applique cette différence à la copie locale.
- **move :** Déclare le déplacement d'une ressource.
- **propdel :** Enlève la propriété du fichier.
- **propedit :** Édite la valeur d'une propriété.
- **propget :** Retourne la valeur d'une propriété.

Les principales commandes (3/3)

- **proplist :** Donne une liste des propriétés.
- **propset :** Ajoute une propriété.
- **resolved :** Permet de déclarer qu'un conflit de modifications est résolu.
- **revert :** Revient à une révision donnée d'une ressource. Les modifications locales sont écrasées.
- **status (st) :** Indique les changements qui ont été effectués.
- **switch :** Met à jour la copie du dépôt.
- **update (up) :** Met à jour la copie locale existante depuis la dernière révision disponible sur le dépôt.
- **unlock :** Retire un verrou.

Les clients graphiques

- Les clients graphiques ne permettent pas de faire plus, mais proposent des interfaces plus élaborées que la ligne de commande.
- Ils permettent notamment de naviguer dans le dépôt comme dans un explorateur de fichiers, d'afficher les informations de manière plus structurée, de garder un historique des logs saisis, etc.
- TortoiseSVN, par exemple, est un client particulièrement bien intégré à Windows, puisqu'il s'ajoute directement au menu contextuel de l'explorateur de fichiers.
- Les commandes sont les mêmes que celles décrites précédemment, mais elles sont lancées par un menu et bénéficient d'interfaces graphiques plus conviviales.

Un serveur SVN sous Linux (1/4)

1. Commencer par installer le paquet `subversion` : `sudo apt-get install subversion`.
2. Créer un répertoire pour héberger les dépôts : `sudo mkdir /var/svn`.
3. Créer un dépôt : `sudo svnadmin create /var/svn/projet1`.
4. Configuration du serveur SVN : il faut qu'au minimum le fichier `/var/svn/projet1/conf/svnserve.conf` contienne :

```
[general]
    # Les utilisateurs non authentifiés : none/read/write
    anon-access = none
    # Les utilisateurs authentifiés : none/read/write
    auth-access = write
    # le fichier de mot de passe
    password-db = passwd
    # Le nom du dépôt
    realm = projet1
```

5. Configuration des mots de passe des utilisateurs dans le fichier : `/var/svn/projet1/conf/passwd` :

```
[users]
    # utilisateur   =  mot de passe
    duvallet = duvallet
```


Un serveur SVN sous Linux (2/4)

6. Démarrer le serveur : `sudo svnserve -d -r /var/svn`.
7. Le port par défaut écouté par svnserve est 3690.
8. Le dépôt créé est désormais accessible, via les commandes SVN de base, à l'adresse : `svn://mon_serveur/projet1`.
9. On peut automatiser le démarrage du serveur :
   - Jusqu'ici, le serveur SVN est lancé avec votre utilisateur. Ce qui implique que le serveur a les mêmes droits que vous.
   - Il est donc recommandé de créer un utilisateur pour lancer le serveur et le définir comme propriétaire du dépôt (nous prendrons ici l'utilisateur svn) :

```
sudo addgroup svn --system
sudo adduser svn --system --home /var/svn --no-create-home --ingroup svn
sudo chown -R svn: /var/svn
```


Un serveur SVN sous Linux (3/4)

10. Création d'un script exécutable `/etc/init.d/svnserve` contenant le code :

```
#!/bin/sh
set -e
if [ -x /usr/bin/svnserve ] ; then
    HAVE_SVNSERVE=1
else
    echo "Svnserve n'est pas installé"
    exit 0
fi
    /lib/lsb/init-functions
case "$1" in
  start)
    log_action_begin_msg "Démarrage du serveur SVN"
    start-stop-daemon --start --chuid svn:svn --exec /usr/bin/svnserve -- -d
    log_action_end_msg $?
    ;;
  stop)
    log_action_begin_msg "Arrêt su serveur SVN"
    start-stop-daemon --stop --exec /usr/bin/svnserve
    log_action_end_msg $?
    ;;
```

Un serveur SVN sous Linux (4/4)

```
    force-reload|restart)
      $0 stop
      $0 start
      ;;
    *)
      echo "Usage: /etc/init.d/svnserve {start|stop|restart|force-reload}"
      exit 1
      ;;
  esac
  exit 0
```

11. ne pas oublier de rendre exécutable le script :

```
sudo chmod +x /etc/init.d/svnserve
```

12. Vous pouvez alors respectivement le démarrer, redémarrer et arrêter à l'aide des commandes suivantes :

```
sudo /etc/init.d/svnserve start
sudo /etc/init.d/svnserve restart
sudo /etc/init.d/svnserve stop
```

13. Pour ajouter le serveur SVN au démarrage de la machine :

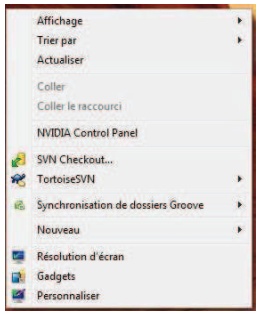
```
sudo update-rc.d svnserve defaults
```


Un client SVN sous Windows : TortoiseSVN

- TortoiseSVN est un des logiciels client de SVN les plus populaires.
- C'est un logiciel libre distribué selon les termes de la licence GNU GPL.
- En s'intégrant dans l'explorateur de Windows, il offre aux utilisateurs de Windows une interface graphique permettant de réaliser la plupart des tâches qu'offre SVN en ligne de commande.
- L'explorateur Windows s'enrichit des fonctionnalités suivantes :
  - Superposition d'icône aux répertoires et fichiers permettant de visualiser instantanément l'état (à jour, modifié, en conflit...).
  - Menu contextuel permettant de committer ou d'updater, à l'échelle d'un fichier, d'une sélection de fichiers ou encore d'un répertoire.
  - Possibilité d'ajouter en mode détails de l'explorateur des colonnes de type numéro de révision, état.
- Il est disponible en version 32 et 64 bits.

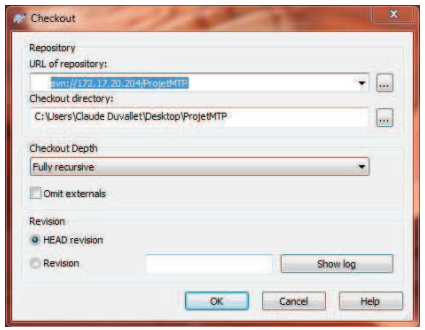
Comment travailler avec TortoiseSVN ?

- On suppose que l'on dispose d'un serveur SVN avec un projet qui s'appelle ProjetMTP.
- On a aussi un utilisateur "mtp" avec pour mot de passe "mtp".
- Nous allons maintenant récupérer les sources du projet (vide) dans notre répertoire de travail :
  - Dans le menu contextuel de Windows, nous avons maintenant des commandes "TortoiseSVN".



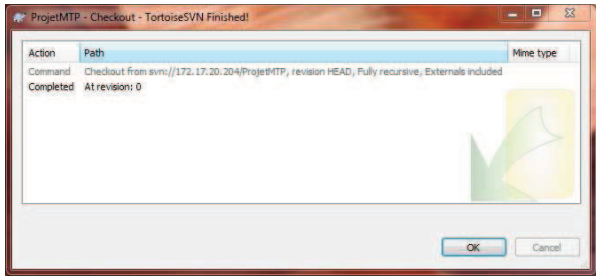
Récupérer mon premier projet (1/2)

- Dans le menu contextuel de Windows, nous choisissons la commande "checkout" et nous renseignons cet écran.

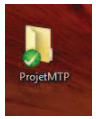


Récupérer mon premier projet (2/2)

- Lorsque la récupération est terminée, vous obtenez un écran avec les résultats des opérations.
- Si des erreurs son survenues elles apparaîtront dans cet écran.

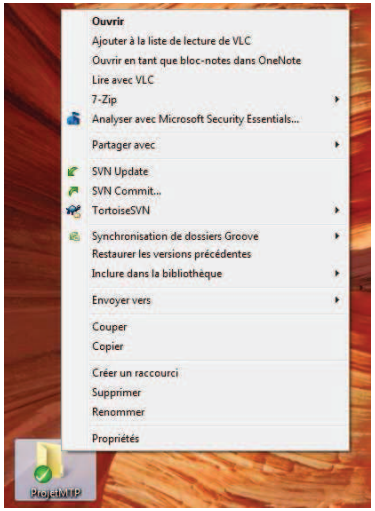


- Votre projet est présent dans un dossier avec une encoche verte qui indique que le projet est à jour par rapport au serveur.



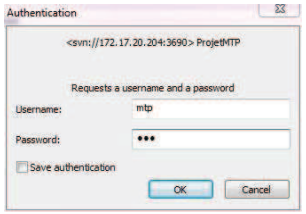
Travailler sur le projet

- Dans le répertoire du projet, vous allez créer un sous-répertoire à votre nom dans lequel vous placerez un fichier vide "Fichier1.java".
- Il faut ensuite valider ces modifications au niveau du serveur.
- Pour cela, vous allez de nouveau vous placer sur le répertoire du projet et faire appel au menu contextuel.



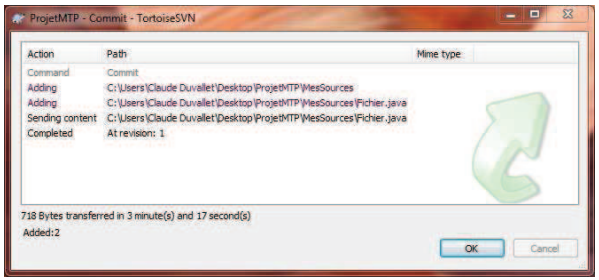
Mettre à jour le projet (1/2)

- Après avoir choisi la commande "commit", une boite de dialogue vous demandera de vous authentifier.
- Il faut donc qu'un utilisateur à votre nom ait été créé sur le serveur.



Mettre à jour le projet (2/2)

- Une fois, le "commit" terminé, un écran affichant le résultat des opérations apparaîtra.
- Si des erreurs son survenues elles apparaîtront dans cet écran.



Connexion en mode sécurisé au serveur

- Vous pouvez aussi vous connecter en mode sécurisé au serveur via le protocole ssh.
- Pour cela, il faut :
  - se connecter au moyen d'un utilisateur système (et non svn).
  - préciser le chemin complet du dépôt.
  - avoir défini des droits permettant à l'utilisateur de se connecter au dépôt (la plupart du temps des droits en écriture).

  ⇒ Cela donne l'url suivante : `svn+ssh://duvallet@monserveur/var/svn/ProjetMTP`.

Accéder par le WEB à vos dépôts grâce à Apache

- Pour installer un serveur SVN utilisant le serveur web Apache, nous aurons besoin d'Apache et de la bibliothèque de Subversion pour Apache.
- Il faut donc installer les paquets apache2, libapache2-svn.
- Ensuite il faut configurer SVN pour Apache :
  - La configuration principale du serveur SVN se situe dans le fichier `/etc/apache2/mods-available/dav_svn.conf`.
- Il y a deux orientations de configuration possible :
  - Configurer tous les dépots un par un, ce qui permet de gérer les autorisations dépot par dépot mais qui oblige à modifier la configuration d'Apache (et donc de le relancer ensuite) à chaque ajout de dépot.
  - Configurer un répertoire parent de tous les dépots, ce qui n'oblige pas à reconfigurer Apache pour chaque ajout mais qui ne permet pas de gérer les autorisations finement.

Configurer les dépôts au cas par cas

- Soit le dépot "ProjetMTP" qui se situe dans le répertoire `/var/svn` et on veut y accéder par l'url `http://mon_serveur/ProjetMTP`.
- Ouvrir le fichier `/etc/apache2/mods-available/dav_svn.conf` et décommenter certaines lignes, et modifier le nom par l'url que vous voulez utiliser :

```
<Location /ProjetMTP>
DAV svn
SVNPath /var/svn/ProjetMTP
</Location>
```

- Redémarrer Apache : `sudo /etc/init.d/apache2 restart`
- Vérifier à l'aide d'un navigateur que `http://mon_serveur/ProjetMTP` réponde.
- Pour ajouter un deuxieme dépôt, il faut recopier le contenu entre les balises `<location>` et `</location>` une deuxième fois dans le fichier, et donner une url différente au deuxième projet.

Configurer les dépôts de façon globale

- Dans ce cas, on a juste besoin de connaitre l'url de la racine des dépots et l'adresse de cette racine sur le disque dur.
- Nous utilisons l'url `http://mon_serveur/svn` et la racine s'appelle `/var/svn`.
- Ouvrir le fichier `/etc/apache2/mods-available/dav_svn.conf` et décommenter la ligne d'ouverture de l'environnement, et modifier le nom par l'url que vous voulez utiliser :

```
<Location /svn>
DAV svn
SVNParentPath /var/svn
SVNListParentPath On
</Location>
```

- Redémarrer Apache : `sudo /etc/init.d/apache2 restart`
- Vérifier à l'aide d'un navigateur que `http://mon_serveur/svn` réponde (liste des dépots si la liste est activée, erreur HTTP 403 sinon).

Authentification sous Apache

- Il faut modifier le fichier `/etc/apache2/mods-available/dav_svn.conf` pour activer l'authentification, en décommentant et renseignant certaines lignes comme ceci :

```
AuthType Basic
AuthName "Depot Subversion"
AuthUserFile /etc/apache2/dav_svn.passwd
Require valid-user
```


Création du fichier des utilisateurs authentifiés

- Créer le fichier `/etc/apache2/dav_svn.passwd` pour un utilisateur, de cette manière :

```
sudo htpasswd -cs /etc/apache2/dav_svn.passwd utilisateur1
```

  - Pour créer d'autres utilisateurs, utiliser la commande :

```
sudo htpasswd -s /etc/apache2/dav_svn.passwd utilisateur2
```

  - Faire appartenir ce fichier à l'utilisateur d'Apache :

```
sudo chown www-data:www-data /etc/apache2/dav_svn.passwd
```

  - Redémarrer à nouveau Apache :

```
sudo /etc/init.d/apache2 restart.
```

- Votre dépôt doit maintenant être accessible via les commandes SVN de base sous l'URL : `http://mon_serveur/ProjetMTP`
- Si vous avez suivi la procédure d'authentification, un nom d'utilisateur et mot de passe (précédement créés) seront demandés.

Faire du SVN sous Windows avec VisualSVN

- VisualSVN est un outil permettant de faire du SVN sous Windows.
- VisualSVN met à disposition un serveur et un client qui fonctionne indépendemment.
- Les deux sont téléchargeables gratuitement
- L'installation et la configuration du serveur est aisée.
- Le client VisualSVN nécessite Visual Studio pour pouvoir fonctionner.
- Ce client est totallement basé sur TortoiseSVN.

Faire du SVN avec Eclipse

- Il existe deux plugins Eclipse permettant de se connecter à un serveur SVN :
  - Subclipse : est un projet de la société Tigris qui édite Subversion.
  - Subversive : est un projet officiel de la fondation Eclipse.
- Ces deux plugins s'installent de façon classique grâce au gestionnaire de mise à jour d'Eclipse.

Installation de Subversive (1/2)

- Depuis la version 3.4, Subversive fait désormais partie des projets Eclipse.
- Il est composé de deux parties,
  - la première qui permet d'utiliser les différentes fonctionnalités SVN,
  - et la seconde partie qui permet de connecter Eclipse à un serveur SVN.
- 1ère étape : Ajouter les update-sites des deux parties de Subversive, via le Gestionnaire de Mises à Jour de Eclipse :
  - L'environnement : il est déjà référencé par défaut, dans le menu **"Help > Install New Software..."**, en sélectionnant **"Galileo - `http://download.eclipse.org/release/galileo`"**, puis en ouvrant la rubrique **"Collaboration Tools"**.
  - Les connecteurs : **`http://community.polarion.com/projects/subversive/download/eclipse/2.0/galileo-site/`**

Installation de Subversive (2/2)

- 2ème étape : Sélectionner les modules réellement nécessaires parmi tous ceux proposés :
  - Le premier module à sélectionner est celui qui s'intitule "Subversive SVN Team Provider (incubation)".
  - Le deuxième module à sélectionner est un des connecteurs disponibles, lequel doit être adapté au serveur SVN que vous utilisez, ainsi qu'à votre environnement de développement. Par défaut, vous povez prendre "Subversive SVN Connectors".
- Une fois les plug-ins sélectionnés, il suffit de valider l'installation et laisser Eclipse redémarrer.

Installation de Subclipse (1/2)

- 1ère étape : Ajouter un des update-sites de Subclipse, via le Gestionnaire de Mises à Jour de Eclipse.
- Il existe plusieurs update-sites selon la version de Subclipse :
  - Subclipse 1.6.x (pour SVN 1.6.x) : `http://subclipse.tigris.org/update_1.6.x`.
  - Subclipse 1.4.x (pour SVN 1.5.x) : `http://subclipse.tigris.org/update_1.4.x`.
- 2ème étape : Sélectionner les modules nécessaires parmi ceux proposés.
  - Après avoir ajouté un des deux update-sites, une liste de 8 ou 9 modules apparaît.
  - A moins de savoir précisément ce dont vous avez besoin, sélectionnez-les tous, de cette manière vous n'oublirez rien.
- Une fois les plug-ins sélectionnés, il suffit de valider l'installation et laisser Eclipse redémarrer.

Installation de Subclipse (2/2)

- Pour fonctionner, Subclipse nécessite :
  - soit le module JavaHL (Java High Level) qui s'appuye sur les bibliothèques natives du client SVN installé sur le poste,
  - soit le module SVNKit qui lui est codé en Java et qui n'a pas besoin de client SVN préinstallé.
- Donc vous l'aurez compris, si vous n'avez pas installé préalablement de client natif pour utiliser SVN, que ce soit le client en ligne de commande ou le client graphique tel que Tortoise, vous serez contraint d'utiliser le module SVNKit car le module JavaHL ne fonctionnera pas.
- Le choix entre les deux modules se fait au niveau des préférences de Eclipse, dans le menu **"Window > Preferences > Team > SVN"**, puis dans bloc **"SVN Interface"**, vous pouvez sélectionner le client **JavaHL (JNI)** ou le client **SVNKit (pure Java)**.

Utilisation de SVN avec Eclipse

- Ce n'est pas très compliquer !
- Il existe deux types de manœuvres possibles :
  - l'ajout d'un projet existant dans un dépôt (repository).
  - la récupération d'un projet existant de puis un dépôt.
- Vous pouvez ajouter au préalable des dépôts en ajoutant un nouveau projet de type SVN puis en sélection **"Repository Location"**.
- Vous pouvez récupérer un projet existant sur un dépôt en sélectionnant **"Project from SVN"**. Vous pourrez ensuite soit sélectionner un repository existant soit en créer un nouveau.
- Vous pouvez aussi ajouter un projet exisant dans votre Workspace : Il faut utiliser le menu contextuel du projet et dans le menu **"team"** il faut faire **"Share Project"**.