

CVS mini-tutoriel

Emmanuel Cornet
Emmanuel.Cornet@Ens.fr

31 mars 2004

Table des matières

I Principe de CVS	1
II Créer un dépôt CVS	2
II.1 Ce que doit faire l'administrateur du projet	2
II.2 Ce que doivent faire les utilisateurs	3
III Synchronisation	3
III.1 Récupérer le projet	3
III.2 Synchroniser ses modifications	4
IV Ajouter de nouveaux fichiers et répertoires	4
V Aller plus loin	4

Avant-propos

Ce document explique en quelques pages les principes de l'utilisation courante de CVS.

- Je n'expliquerai pas comment installer le programme CVS sur votre poste de travail. Je supposerai donc que vous avez déjà accès à cet outil; sous Linux Debian, vous pouvez installer CVS grâce à

```
apt-get install cvs
```

en ayant les privilèges d'administration.

- Je ne rentrerai pas dans les détails d'une utilisation plus avancée de CVS; je me limiterai au strict minimum, aux commandes les plus courantes. Pour plus d'informations, reportez-vous à la section « Aller plus loin ».

I Principe de CVS

CVS (*Concurrent Version System*) est un outil extrêmement pratique, utilisé généralement dans des projets informatiques d'une certaine envergure, et qui permet à plusieurs personnes de synchroniser leur travail. En particulier, CVS sait bien garder les versions les plus récentes de tous les fichiers (ou parties de fichiers) concernés, même s'il n'est pas non plus devin et aura, par exemple, des problèmes pour faire la part des choses lorsque deux personnes retouchent la même partie d'un même fichier à des instants proches.

Un point important : **CVS garde tout** ! Chaque modification est sauvegardée, chaque stade de l'évolution du projet laisse une trace, aucune nouvelle version n'efface totalement les anciennes. En particulier, le responsable du projet peut, lorsqu'il voit qu'un stade

important a été atteint, créer un « point de sauvegarde » auquel il sera aisé de revenir par la suite en cas de besoin.

Second point important : une fois le dépôt initialisé (voir plus bas),

Ne touchez pas au dépôt CVS manuellement.

Il est, de très loin, préférable, de récupérer (via CVS) la version la plus récente du projet, puis de modifier cette copie locale, et enfin de synchroniser les modifications (toujours via CVS, voir les procédures plus bas).

II Créer un dépôt CVS

II.1 Ce que doit faire l'administrateur du projet

Pour créer un dépôt CVS qui servira de « quartier général » et avec lequel les utilisateurs synchroniseront leurs données, voici la marche à suivre.

1. Commencez par créer une esquisse de l'arborescence de votre projet dans un répertoire temporaire, en essayant d'anticiper sur la manière dont se répartiront les différentes branches du projet. Par exemple :

```
mkdir $HOME/temp/cvs_esquisse/monprojet
mkdir $HOME/temp/cvs_esquisse/monprojet/scripts
mkdir $HOME/temp/cvs_esquisse/monprojet/documentation
mkdir $HOME/temp/cvs_esquisse/monprojet/fichiers_config
```

2. Choisissez l'endroit où se situera le répertoire dédié à CVS. Il pourrait se situer, par exemple, directement dans votre répertoire \$HOME. Créez donc ce répertoire grâce à la commande

```
mkdir $HOME/cvs
```

3. Définissez la variable d'environnement CVSROOT (reportez-vous quelques lignes plus bas pour comprendre l'utilité de cette variable.) dans le fichier de configuration de votre shell préféré (par exemple, \$HOME/.zshrc pour le shell zsh) en y plaçant la ligne

```
export CVSROOT=$HOME/cvs
```

(syntaxe pour un shell d'origine « Bourne » comme bash ou zsh).

4. Maintenant que CVS sait où se trouvera son dépôt, demandez-lui de l'initialiser :

```
cvs init
```

5. Comme nous l'avons vu plus haut, **il ne faut jamais toucher au dépôt manuellement**. Ne copiez donc pas vous-même votre début d'arborescence dans le dépôt, mais demandez plutôt à CVS de le faire pour vous :

```
cd $HOME/temp/cvs_esquisse
cvs import -m "Première importation" travail manu version1
```

Dans cette dernière ligne, le premier argument de la commande `cvs import` est l'endroit où vous voulez que le projet soit stocké. Dans notre cas, CVS créera l'arborescence à partir de `$HOME/cvs/travail/monprojet`. L'option `-m` et la chaîne de caractères qui la suit permet de placer un commentaire sur ce que vous êtes en train de faire, dont CVS gardera une trace. Le deuxième argument est appelé la « marque du constructeur » : mettez simplement votre nom ou un pseudonyme qui vous identifie. Le dernier argument permet de préciser le numéro de version.

6. Une dernière chose à faire, pour simplifier la tâche des utilisateurs, est de créer une sorte d'« alias » pour un projet donné, qui contiendra tous les fichiers concernés, même s'il sont situés dans une arborescence complexe. Il faut pour cela modifier le fichier `$CVSROOT/CVSROOT/modules`. Si vous avez compris la règle ci-dessus, vous savez déjà qu'il faut modifier ce fichier via CVS et non directement ! Supposons que nous voulions appeler ce projet `Mon_projet`. Écrivons alors la suite de commandes suivante (commencez par vous placer dans un répertoire temporaire) :

```
cvsv checkout CVSROOT/modules
cd CVSROOT
```

Éditez le fichier `modules` en y ajoutant (après les commentaires) la ligne `Mon_projet travail/monprojet`. Enregistrez ce fichier et terminez par :

```
cvsv commit -m "Ajout du module Mon_projet" modules
cd ..
cvsv release -d CVSROOT
```

7. Détruisez maintenant l'esquisse d'arborescence que vous aviez créée pour ne pas être tenté de la modifier directement : vous modifierez les fichiers et les répertoires uniquement par CVS, et ils seront gardés bien au chaud dans le dépôt.

II.2 Ce que doivent faire les utilisateurs

1. Le programme `cvsv` doit savoir où vous avez choisi de placer le dépôt ; il utilise pour cela la variable d'environnement `CVSROOT`. Le mieux est donc d'initialiser cette variable dans un fichier automatiquement ouvert à votre connexion
2. Pour éditer les commentaires faits à chaque modification du projet, vous pouvez de même affecter les variables `CVSEEDITOR` et `EDITOR` (par exemple, `nano` ou `vi`).
3. Créer un répertoire qui accueillera votre travail à synchroniser :

```
mkdir CVSWork
```

Et pour être prêt à travailler, récupérez le projet (on poursuit l'exemple ci-dessus) `Mon_projet` :

```
cd CVSWork ; cvsv checkout Mon_projet
```

III Synchronisation

III.1 Récupérer le projet

Facile :

```
cvsv checkout nom_du_projet
```

pour la première fois. Si vous disposez déjà d'une copie locale d'un ou plusieurs projets et que vous souhaitez simplement les mettre à jour avec le dépôt principal, placez-vous à la racine de votre répertoire CVS et tapez

```
cvsv update
```

Si vous ne voulez récupérer les modifications que pour un fichier donné :

```
cvsv update nom_du_fichier
```

III.2 Synchroniser ses modifications

Pour transporter vos modifications locales vers le dépôt, ce n'est pas beaucoup plus compliqué :

```
cvsv commit
```

CVS vous informe des fichiers qu'il met à jour. Si vous voulez mettre à jour un fichier isolé :

```
cvsv commit nom_du_fichier
```

IV Ajouter de nouveaux fichiers et répertoires

Commencez par créer le fichier ou le répertoire en question, puis :

```
cvsv add truc_a_ajouter
```

et, comme CVS vous l'indique, il faut terminer par un

```
cvsv commit truc_a_ajouter
```

pour que la procédure soit effective.

V Aller plus loin

Voici quelques sites web où vous pourrez en apprendre un peu plus sur le principe et l'utilisation de CVS.

- <http://www.cvshome.org>
- <http://europa.ac-grenoble.fr/slis/devel/cvs-howto.html>
- <http://www.loria.fr/~molli/cvs/HowToCVS.ps>