HOWTO du CVS d'Arabeyes

Mohammed Elzubeir, Projet Arabeyes <contact (at) arabeyes dot org>

$Revision: 1.6 $

Ce document est destiné à vous montrer comment accéder et utiliser le repository CVS d'Arabeyes.

La traduction française de ce document a été réalisée par Youcef Rabah Rahal <rahal (at) arabeyes dot org>.

Table of Contents

Introduction 1
Accès au CVS 2
Se loguer 2
Faire un 'check out' (télécharger) 2
Faire un 'commit' des changements (upload) 3
Mise à jour de votre copie 4
Lignes de conduite du CVS 4
Les nouveaux répertoires 4
Le 'commit' des compte-rendus 4
Le 'commit' multiple de fichiers 5
L'import d'un nouveau projet 5
Les fichiers acceptables sur CVS 6

Introduction

CVS (Concurrent Versions System) veut dire système de gestion de versions concurrentes. C'est un système de control de versions qui a été developpé dans le domaine public par plusieurs personnes depuis 1986. En utilisant CVS, vous pouvez enregistrer l'historique de vos fichiers source et/ou de vos documents. Il ne vérifie pas votre travail pour vous, mais c'est une bonne protection contre les désastres, si jamais il se produisent.

Ce HOWTO ne vas pas tout vous apprendre sur CVS, mais il est conçu pour être un guide rapide pour l'utilisation du repository CVS. Pour plus d'informations sur l'utilisation de CVS vous pouvez consulter:

- CVSHome - http://www.cvshome.org/
- Introduction à CVS - http://linux.oreillynet.com/pub/a/linux/2002/01/03/cvs_intro.html [http://linux.oreillynet.com/pub/a/linux/2002/01/03/cvs_intro.html]

Il existe plusieurs clients CVS gratuits que vous pouvez télécharger. Si vous êtes un utilisateur Linux/Unix, alors il fait certainement dèjà partie de votre installation standard. Si vous êtes un utilisateur Windows, vous avez deux options:

- Le shell bash Cygwin - http://sources.redhat.com/cygwin/ [http://sources.redhat.com/cygwin/]

Cygwin est un environement UNIX pour Windows, livré avec un client CVS en ligne de commande.

- WinCVS - http://www.wincvs.org/

  WinCVS est un client CVS graphique pour MS-Windows.

Il est fortement recommandé d'utiliser le client CVS Linux/Unix. Cependant, si vous devez utiliser Windows, le shell bash Cygwin est ce que nous recommandons. Ce HOWTO fait l'hypothèse que soit vous connaissez l'equivalent des commandes dans le client graphique, soit vous utilisez le client en ligne de commande.

Accès au CVS

Il existe deux type d'accès au CVS: un accès anonyme (accès en lecture seule; ouvert à tous) et un accès aux developpeurs (lecture/écriture; restreint).

Afin d'accéder au serveur CVS (connu sous le nom de 'pserver'), vous devrez avoir un compte (nom d'utilisateur et mot de passe). Pour le besoin de ce document, nous ferons l'hypothèse que votre nom d'utilisateur est *myaccount* et que votre mot de passe est *rocks*.

Pour un accès anonyme remplacez le nom d'utilisateur par *anoncvs* et le mot de passe par *anoncvs*.

Se loguer

La première chose à faire est de se loguer ! Voici quelques informations vitales: Vous allez devoir régler votre CVSROOT, par:

```
bash-2.04$ export CVSROOT=:pserver:myaccount@cvs.arabeyes.org:/home/arabeyes/cvs
```

Si vous utilisez WinCVS alors vous devrez casser celà en:

- Authentication type - pserver
- Username - myaccount
- Host name - cvs.arabeyes.org
- CVSROOT - /home/arabeyes/cvs

Un fois que votre environement est réglé, vous pouvez vous loguer comme suit:

```
bash-2.04$ cvs login
```

Entrez votre mot de passe quand il vous sera demandé. Vous y êtes !

Faire un 'check out' (télécharger)

Le mot 'checkout'1 dans CVS veut dire télécharger les fichiers sur votre machine locale. C'est analogue à

l'emprunt d'un livre depuis la bibiothèque, avec cependant la possibilté de le mettre à jour au retour !

```
bash-2.04$ cvs checkout projects
```

La commande ci-dessus va créer une strucuture de répertoires sur votre ordinateur. Qui contiendra tout ce qui est en-dessous de projects dans le repository CVS. Donc, disons que nous ne voulons pas tout ce qui est sous projects. Faisons l'hypothèse que vous voulez les sources du projet Akka et tout ce qui le concerne.

```
bash-2.04$ cvs checkout projects/akka
```

Ceci devra télécharger tout ce qui est sous `akka/`.

Faire un 'commit' des changements (upload)

Quand vous faites le 'commit' d'un fichier, vous faites essentiellement un upload vers le repository. Donc, maintenant que avez téléchargé les fichiers et que vous avez fait quelques changements, vous voudrez les remettre dans le repository.

Pour le besoin de cet exemple, nous dirons que vous modifiez le fichier 'myfile.cc'. Voici comment faire un 'commit':

```
bash-2.04$ cvs commit myfile.cc
```

Si le fichier n'existe pas déjà, et que vous créez un nouveau fichier, alors vous devez l'ajouter par un 'add' au préalable. Voici comment:

```
bash-2.04$ cvs add myfile.cc
bash-2.04$ cvs commit myfile.cc
```

La première commande crée une entrée pour le fichier dans le repository, et la seconde fait le 'commit' pour de vrai (le rend disponible). Si le fichier existe déjà et que vous l'avez changé, vous pouvez ignorer la commande 'cvs add'. Pour une aide supplémentaire, faites 'man cvs' dans votre prompt Linux/Unix.

Uhe fois que vous exécutez la commande 'commit', un éditeur de texte sera lancé pour que vous puissiez entrer un bref compte-rendu de ce que vous avez fait. Ceci est particulièrement utile pour les autres personnes qui travaillent sur le même projet - puisque ça leur donne une idée sur ce que vous avez changé sans pour autant avoir à consulter le(s) fichier(s). L'éditeur est habituellement vi, sur la plupart des systèmes Linux/Unix. Cependant, ce n'est pas nécessairement le cas. Ce sera n'importe quel éditeur auquel vous auriez réglé la variable d'environement $EDITOR. Pour trouver à quoi c'est réglé dans votre environement faites:

```
bash-2.04$ env | grep EDITOR
```

1'checkout' et 'co' désignent la même opération en ligne de commande - Vous pouvez les utiliser tous deux indifféremment

Pour avoir le mode *insertion* dans vi, tapez **i**. Vous pourrez alors taper votre bref compte-rendu. Une fois que vous avez fini, tapez **ESC** (qui vous fera sortir du mode insertion), et puis tapez **:wq** puis Entrée. Ceci vous fera sauvegarder le fichier (le bref compt-rendu) et vous fera quitter vi. Votre 'commit' sera alors exécuté.

Mise à jour de votre copie

Comme il est probable que d'autres ont pu faire des changements durant le developpement du projet, votre copie locale du contenu de CVS peut ne pas être tout le temps à jour. Afin de garder votre copie à jour vous pouvez faire:

```
bash-2.04$ cvs update
```

Notez que ceci ne peut être fait que depuis le répertoire que vous voulez mettre à jour.

Lignes de conduite du CVS

Ce qui suit contient quelques règles simples et les lignes de conduite à propos de toutes les activités CVS d'Arabeyes. Merci de bien les suivre afin de rendre agréable le travail de tous. La violation des lignes de conduite qui suivent peut résulter en la suspension temporaire de votre compte CVS jusqu'à ce que le problème soit résolu.

Les nouveaux répertoires

Avant de créer un quelconque nouveau répertoire, merci de toujours consulter l'administrateur CVS. Ceci est particulièrement important pour éviter la redondance et maintenir une organisation ordonnée.

Ceci ne s'applique pas aux propriétaires des modules. Par exemple, si l'auteur de l'application X décide d'ajouter `X/yz/` à son répertoire `X/`, il n'y a pas de nécessité à une quelconque approbation de l'administrateur.

Le 'commit' des compte-rendus

Il va sans dire qu'il est très important d'entrer un commmentaire utile et compréhensible lors des 'commits'.

- La traduction (fichiers PO):

  Quand vous faites un 'commit' de fichier(s) `.po`, assurez vous toujours que le commentaire inclut bien la sortie de:

  ```
  $ msgfmt --statistics file.po 2
  ```

  Ceci pour éviter les commentaires de type "translated a little", qui ne veut pas dire grand chose, si ce n'est rien du tout.

- Le developpement du code source:

[2] Les commentaires sur les 'commits' du code source peuvent être un peu plus faciles (puisque le type 'msgfmt' est un utilitaire qui est livré avec le package gettext.

de changement varie). Assurez vous toujours de garder une trace des changements que vous faites *au moment même* où vous les faites, ainsi vous vous en souviendrez au moment du 'commit'.

Un exemple d'un mauvais commentaire est: "I made a few fixes". Quels corrections ? Si vous n'arrivez plus à vous souvenir de ce que vous avez fait, alors faites un 'diff' des changements, consultez le et notez les changements faits.

- Les documents:

  En général, les commentaires sur la documentation seront inévitablement vagues. Si le document est petit, ce n'est pas un problème. Si ce n'est pas le cas, indiquez l'endroit que vous avez modifié. Par exemple: "Chapter 3, Section 4 - Re-wrote the second paragraph for clarity" est un bon commentaire.

Le 'commit' multiple de fichiers

Quand vous avez faits des changements sur plusiers fichiers, c'est mieux de faire un 'commit' en un seul paquet, au lieu de le faire fichier par fichier. Ceci pour 2 raisons essentielles:

- Minimiser le nombre de notifications par mail de CVS qui sont reçues par tous ceux qui sont inscrits sur la liste 'cvs', à chaque fois qu'un changement est fait.
- La plupart des fichiers changés concernent un seul sujet. Par exemple, si deux fichiers `test.cc` et `test.h` sont changés, il n'est pas nécessaire de faire des 'commits' individuels.

Il existe deux façons pour faire un 'commit' multiple de fichiers.

```
$ cvs commit file1 file2 file3
```

Ou vous pouvez laisser CVS le faire recursivement:

```
$ cvs commit
```

La première manière vous laisse plus de flexibilité sur les fichiers a envoyer, tandis que la dernière fait simplement un 'commit' de n'importe quel fichier qui aurait changé dans le répertoire courant et dans ses sous-répertoires.

L'import d'un nouveau projet

Bien que ceci n'arrive pas très souvent, c'est utile de le mentionner, spécialement pour ceux qui possèdent un projet et voudraient ajouter un nouveau répertoire qui contient déjà un contenu non nul.

Quand c'est le cas, utilisez **cvs import**. Par exemple, si projX dans `projects/projX` va ajouter un nouveau répertoire nommé `games` contenant 10 fichiers, faites ceci:

```
$ cvs import projects/projX/games arabeyes start
```

Pour plus d'informations, tapez **'cvs import --help'** ou consultez le manuel CVS.

Les fichiers acceptables sur CVS

Il existe une règle très simple à suivre. N'importe quel fichier qui est généré par un programme depuis un autre fichier qui réside sur CVS ne doit pas être mis dans CVS. C'est de la redondance inutile.

Les fichiers compressés (gzip/bzip2/Z/etc) ne doivent pas être mis dans CVS sous cette forme (ou sous la forme de fichiers tar).

Généralement, les seuls types de fichiers binaires acceptables sur CVS sont les images graphiques et les fichiers audio. Tout autre format de fichier doit être du texte.