Licence d'informatique
Module de C/Unix

La commande `cvslocalstatus`

Examen – Partie Unix

Philippe MARQUET

Janvier 2002

Les documents de cours et TD sont autorisés.
On rendra deux copies séparées pour la partie langage C et pour la partie Unix ; la partie Unix sur une *copie de couleur*.
On remarquera que les questions peuvent être traitées indépendamment les unes des autres.
On introduira tout code Bourne-shell par une brève description de son comportement ; on attachera un commentaire à toute portion non triviale de code.

1 Questions de cours

Exercice 1 (`quota` *vs.* `du`)
Expliquez pourquoi les résultats des deux commandes `quota -v` et `du -s $HOME` peuvent différer.

Exercice 2 (`cp` erroné)
Expliquez pourquoi la commande suivante peut produire un tel résultat :

```
% cp *.c~ ~/c/xam &
cp: 'main.c~' and '/users/phm/c/xam/main.c~' are the same file
cp: 'tools.c~' and '/users/phm/c/xam/tools.c~' are the same file
```

2 Statut cvs local

Le système CVS est un gestionnaire des versions successives d'un ensemble de fichiers. CVS maintient l'ensemble des versions successives des fichiers dans une archive. Le plus souvent, cette archive est accédée au travers un serveur CVS accessible depuis le réseau. On travaille sur des copies locales des fichiers. Le plus souvent ces copies locales contiennent la version la plus récente des fichiers. Ces copies peuvent être modifiées (par exemple avec un éditeur de textes). On met ensuite l'archive à jour par une opération de *commit*. On peut aussi récupérer de nouvelles versions, par exemple des fichiers qui ont été modifiés par d'autres utilisateurs. Ces actions de mise à jour nécessitent d'avoir une connexion avec le serveur qui gère l'archive CVS.

Le but de la commande `cvslocalstatus` objet de cet examen est de fournir une information sur les fichiers contrôlés par CVS sans accès au serveur. Certaines informations sont conservées localement, par exemple la liste des fichiers gérés par CVS et la date de dernière modification des fichiers lors d'une synchronisation avec l'archive.

2.1 Le répertoire CVS

Pour un répertoire local contrôlé par CVS, les informations disponibles localement sont :
– les fichiers, avec leur contenu et leur dernière date de modification ;
– un répertoire nommé `CVS`.

Ce répertoire `CVS` contient les fichiers suivants :

Id: xam02.tex,v 1.6 2002/02/04 10:27:52 marquet Exp

- `CVS/Root` identifie l'archive, par exemple le nom de la machine serveur, le nom d'utilisateur à utiliser sur cette machine, etc.
- `CVS/Repository` identifie le chemin relatif du répertoire dans l'archive ;
- `CVS/Entries` identifie la liste des fichiers gérés par CVS pour le répertoire. Chacune des lignes de ce fichier est de la forme :
  `[code]/filename/revision/date/[keyword]/[option]`
  - Si `code` est présent, sa valeur est `D` pour indiquer que la ligne concerne un sous-répertoire lui aussi sous contrôle de CVS.
  - `filename` est le nom du fichier dans le répertoire. C'est forcément un nom simple, les caractères `/` de la ligne peuvent ainsi sans ambiguïté servir de délimiteurs.
  - `revision` est le numéro de version du fichier dans l'archive. C'est par exemple un numéro de la forme `1.4`.
  - `date` est la date de modification du fichier lors de la dernière mise à jour avec l'archive. Ces dates sont au format standard tel qu'affiché par la commande `date`.
  - Les champs `keyword` et `option` seront ignorés.

2.2 Fichiers et variables de configuration

Un certain nombre de fichiers et variables de configuration sont considérés par CVS.

Le fichier `$HOME/.cvsignore`, le contenu de la variable `$CVSIGNORE`, et le fichier `.cvsignore` du répertoire courant identifient, dans cet ordre, des noms de fichiers ignorés de CVS. Ces fichiers `cvsignore` contiennent une expression régulière de nom de fichier par ligne. La variable `$CVSIGNORE` contient une liste d'expressions régulières séparées par des espaces. Les fichiers dont le nom correspond à une de ces expressions régulières ne sont pas gérés pas CVS. De plus, un certain nombre de fichiers sont toujours ignorés par CVS, citons les fichiers nommés `CVS`, `RCS`, `SCCS`, `*.old`, `*.bak`, `core`.

Le fichier `CVS/Notify` contient les noms des fichiers locaux dont l'ajout a été notifié à l'archive mais dont la première version n'a pas encore été transmise à l'archive (on dira *non commis*).

2.3 Exemple

Soit la session shell suivante :

```
% ls -F
CVS/                xam01.make
Makefile            xam01.tex~
Makefile~           xam02.aux
Makefile.bak        xam02.dvi
auto/               xam02.log
fig/                xam02.make
xam00.make          xam02.make~
xam00.tex           xam02.tex
xam01-test/         xam02.tex~
% cat CVS/Entries
D/fig////
/xam00.make/1.2/Sun Jan 2 14:21:16 2000//
/xam00.tex/1.11/Mon Jan 10 21:18:04 2000//
/xam01.tex/1.6/Tue Aug 28 09:51:10 2001//
/xam01.make/1.2/Mon Oct 29 09:46:12 2001//
D/xam01-test////
/Makefile/1.15/Mon Nov 12 13:16:00 2001//
/.cvsignore/1.7/Mon Nov 12 13:17:24 2001//
% cat CVS/Notify
/xam02.tex//
```

```
/xam02.make//
% cat ~/.cvsignore
*~
auto
% cat .cvsignore
*.aux
*.dvi
*.haux
*.idx
*.ilg
*.log
*.pdf
*.ps
*.ps.bz2
*.ps.gz
% date -r Makefile
Sun Jan 13 21:13:36 2002
% man date | grep FILE
       -r, --reference=FILE
              display the last modification time of FILE
```

Notre commande `cvslocalstatus` peut reporter, sans accéder au serveur CVS, les informations suivantes :

```
% cvslocalstatus
Fichiers modifies:
----------
./Makefile

Fichiers notifies, non commis:
----------
xam02.make
xam02.tex

Fichiers detruits:
----------
xam01.tex

Nouveaux fichiers:
----------
(aucun)
```

2.4 Modifications concurrentes

Soit le cas de figure suivant. Un fichier géré par CVS est modifié localement. Une opération de mise à jour de la copie locale en fonction de l'archive est réalisée. Ce fichier a aussi été modifié par ailleurs. Cette opération est rapportée par CVS comme un *merge*. Une telle opération peut conduire à des conflits, ils sont alors reportés par CVS. Cependant si les modifications apportées concernent des parties disjointes du fichier, CVS assure la cohérence de l'opération en réunissant les modifications locales et les modifications apportées à la version enregistrée dans l'archive. C'est de ces possibles modifications concurrentes que CVS tire son nom (Concurrent Version System) et sa puissance (essayez, vous verrez...).

Dans ce cas, la sémantique de la date de dernière modification enregistrée pour le fichier dans `CVS/Entries` ne peut être celle de la date de modification car on ne saurait alors distinguer que le fichier n'est pas en phase avec l'archive. CVS range alors la chaîne « `Result of merge` » en lieu et place de la date ; par exemple :

```
% grep merge CVS/Entries
/Makefile/1.16/Result of merge//
```

2.5 Exercices

Exercice 3 (Nouveaux fichiers)
Pour cet exercice, on suppose

1. Travailler sur le répertoire courant géré par CVS.
2. Ignorer les sous-répertoires.

Question 3.1 Expliquez pourquoi les fichiers `Makefile.bak` et `xam02.make` ne sont pas reportés comme nouveaux par `cvslocalstatus`.

Question 3.2 Soient une expression régulière de nom de fichier contenue dans la variable `$regexpr` et un nom de fichier contenu dans la variable `$filename`. Comment identifiez-vous le fait que le nom de fichier est reconnu par l'expression régulière ?

Question 3.3 Donnez le code commenté d'une fonction Bourne-shell `ignored_file` qui termine sur un succès si et seulement si un nom de fichier passé en paramètre correspond à un nom de fichier ignoré de CVS (c'est-à-dire identifiée par une expression régulière de `.cvsignore` et autres).

Question 3.4 Donnez le code commenté d'une fonction Bourne-shell `referenced_file` qui termine sur un succès si et seulement si le fichier dont le nom est passé en paramètre est référencé par un des fichiers `Entries` ou `Notify` du répertoire `CVS`.

Question 3.5 Donnez le code commenté d'une fonction Bourne-shell `print_new_files` qui affiche sur sa sortie standard la liste des fichiers du répertoire courant considérés comme nouveaux vis-à-vis de l'archive CVS.

Exercice 4 (Fichiers modifiés, non commis ou détruits)
Pour cet exercice, on suppose toujours

1. Travailler sur le répertoire courant géré par CVS.
2. Ignorer les sous-répertoires.

Question 4.1 Afin de déterminer si un fichier donné a été modifié, il va être nécessaire de comparer des dates avec la date de modification d'un fichier.

Proposez la spécification d'une ou plusieurs fonctions qui aidera/aideront à la manipulation de ces dates. Il n'est pas demandé de donner le code de cette/ces fonctions.

Question 4.2 Donnez le code Bourne-shell commenté d'une fonction `list_mud_files` (`m.u.d.` pour modified, uncommit, and deleted) qui produit les noms des fichiers modifiés (resp. non commis, resp. détruits) du répertoire courant dans le premier (resp. second, resp. troisième) fichier dont le nom est passé. Ce résultat est écrit à raison d'un nom de fichier par ligne.

Exercice 5 (Prise en compte des répertoires)
Question 5.1 Discutez de la prise en compte des sous-répertoires dans la commande `cvslocalstatus`. En particulier, imaginez de passer des options à la commande et/ou aux fonctions définies, proposez une ou plusieurs présentations possibles des résultats, explicitez les mécanismes pouvant être mis en œuvre pour prendre en compte les sous-répertoires et les modifications à apporter aux algorithmes proposés précédemment.

Fin du sujet