

Outils pour le développement d'application : Gestion de version avec CVS

Nicolas MARTIN

Laboratoire d'Océanographie Dynamique et Climatologie

Séminaire Vie Scientifique 10/12/2004

Plan de la présentation

Introduction

Utilisation d'un projet

Participation au développement d'un projet

Gestionnaire d'un projet

Conclusion

Fonctionnalités de CVS

- ▶ Sauvegarde des révisions successives des fichiers d'un projet
- ▶ Gestion automatique des numéros de révision
- ▶ Utilisation des noms symboliques
- ▶ Travail collaboratif : plusieurs développeurs
- ▶ Gestion de plusieurs branches de développement parallèles
- ▶ Utilisation sur tout ensemble de fichiers textes
- ▶ Interface disponible sur différents OS

Caractéristiques d'une base CVS

- ▶ Répertoire de stockage des fichiers sous contrôle de CVS.
- ▶ Un sous-répertoire "administration" : CVSROOT
- ▶ Projets sauvegardés par sous-répertoire
- ▶ Pas d'accès direct à ces fichiers
 - ▶ création d'une copie de travail
 - ▶ modification
 - ▶ mise à jour de la base
- ▶ Fichiers sauvés sous forme de différence entre révision
- ▶ Base CVS locale ou distante (plusieurs moyens d'accès)

base CVS locale ou distante

Contrôlé par la variable `$CVSROOT` ou l'option `-d $CVSROOT` de la commande `cvs`

Dépôt local : `$CVSROOT` contient un chemin du système de fichier local

```
$ setenv CVSROOT $HOME/cvsroot  
$ setenv CVSROOT :local:$HOME/cvsroot
```

Dépôt distant : `$CVSROOT` contient un chemin d'une machine distante sous la forme :

```
[ :METHOD: ] [ [USER] [ :PASSWORD ] @ ] HOSTNAME [ : [PORT] ] / path / module
```

```
$ setenv CVSROOT :ext:moi@cvs.server.net:/cvsroot
```

Numérotation des révisions

- ▶ Numérotation automatique de la forme 1.1, 1.2, 1.3,
- ▶ Changement de révision : incrémentation du dernier nombre
- ▶ Révisions : suite paire de nombres
- ▶ Branches : suite impaires de nombres
- ▶ Création d'une branche
 - ▶ Utilisation d'une révision comme départ (ex. : 1.4)
 - ▶ Création d'une nouvelle branche (ex. : 1.4.2)
 - ▶ Création de la première révision dans cette branche (ex. : 1.4.2.1)
- ▶ Numérotation indépendante pour chaque fichier
- ▶ Utilisation des noms symboliques

Structure d'une commande

```
$ cvs [CVS_OPTIONS] FONCTION [FONCTION_OPTIONS] [FICHIERS]
```

- ▶ Une seule commande **cv**s permet d'accéder à l'ensemble des fonctionnalités
- ▶ Les fonctionnalités de CVS peuvent aussi être obtenues à l'intérieur d'application (éditeurs, IDE, ...)

Structure d'une commande

```
$ cvs [CVS_OPTIONS] FONCTION [FONCTION_OPTIONS] [FICHIERS]
```

CVS_OPTIONS permet de spécifier d'éventuelles options de CVS

- ▶ -d \$CVSROOT utilise une base CVS spécifique
- ▶ -z N utilise le niveau N de compression pour tout échange
- ▶ -n n'effectue par réellement le traitement
- ▶ -t trace le fonctionnement de CVS
- ▶ ...

Structure d'une commande

```
$ cvs [CVS_OPTIONS] FONCTION [FONCTION_OPTIONS] [FICHIERS]
```

FONCTION spécifie la fonctionnalité qui sera utilisée :

commit : mettre à jour la base CVS

checkout : obtenir une copie de travail

update : mettre à jour une copie de travail

diff : différences entre copie de travail et base CVS

status : état d'un fichier vis à vis de la base CVS

...

Structure d'une commande

```
$ cvs [CVS_OPTIONS] FONCTION [FONCTION_OPTIONS] [FICHIERS]
```

FONCTION_OPTIONS permet de spécifier d'éventuelles options de la fonction

- ▶ **-r** REVISION travaille sur la révision mentionnée
- ▶ **-D** DATE travaille sur la révision de la date spécifiée
- ▶ **-b** utilise une branche
- ▶ **-l** mode local (non-récuratif)
- ▶ ...

Structure d'une commande

```
$ cvs [CVS_OPTIONS] FONCTION [FONCTION_OPTIONS] [FICHIERS]
```

FICHIERS permet de spécifier les fichiers à traiter

En l'absence de spécification de fichier, l'ensemble des fichiers de la copie de travail sont considérés

Mode utilisateur

- ▶ Ne participe pas au développement
- ▶ Récupère une révision particulière
- ▶ Peut synchroniser cette révision pour incorporer les modifications
- ▶ Utilise les informations annexes (historique, ...)

cvs checkout [OPTIONS] REP_BASE_CVS

Récupération d'une copie de travail avec les fichiers d'administration de CVS

- ▶ Par défaut `cvs checkout module` crée dans le répertoire courant l'arborescence représentée par `module`
- ▶ l'option `-d SUBDIR` permet de créer cette copie de travail dans le répertoire `SUBDIR`
- ▶ Par défaut `cvs checkout module` récupère la dernière révision en date de `module`
- ▶ l'option `-r REVISION` permet de spécifier la révision souhaitée

```
$ cvs checkout -d Cvs presentations/cvs
cvs checkout: Updating CVS
U Cvs/cvs.tex
```

Accès à une base CVS distante

- ▶ Exemple avec un serveur CVS utilisant la méthode pserver :

```
$ cvs -d:pserver:USER@SERVER:REP_BASE_CVS login
$ cvs -z3 -d:pserver:USER@SERVER:REP_BASE_CVS checkout MODULE
$ ...
$ cvs -d:pserver:USER@SERVER:REP_BASE_CVS logout
```

- ▶ Exemple avec un serveur CVS utilisant SSH :

```
$ export CVS_RSH=ssh
$ cvs -z3 -d:ext:USER@SERVER:REP_BASE_CVS checkout MODULE
```

cvs status FICHIERS

Obtention de l'état de chaque FICHIERS par rapport à la base CVS.

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex          Status: Up-to-date

Working revision:      1.2      Tue Nov 30 11:02:09 2004
Repository revision:  1.2      /usr/home/martin/CVSR00T/Presentations/Cvs/cvs.
Sticky Tag:           (none)
Sticky Date:          (none)
Sticky Options:       (none)

$
```

cv diff [OPTIONS] [FICHIERS]

Affiche les différences entre deux révisions des FICHIERS.

- ▶ sans options `-r REVISION`, la révision actuelle est comparée à la dernière de la branche en cours
- ▶ avec `-r REVISION`, la révision actuelle est comparée à la révision spécifiée
- ▶ avec `-r REVISION1 -r REVISION2`, les deux révisions mentionnées sont comparées entre elles
- ▶ Une option `-r REVISION` peut être remplacée par une option `-D DATE`
- ▶ La plupart des options de la commande `diff` peuvent être utilisées

```
$ cvs diff -u cvs.tex
```

```
Index: cvs.tex
```

```
=====
```

```
RCS file: /usr/home/martin/CVSRROOT/Presentations/Cvs/cvs.tex,v
```

```
retrieving revision 1.6
```

```
diff -u -r1.6 cvs.tex
```

```
-- cvs.tex      30 Nov 2004 15:56:22 -0000      1.6
```

```
+++ cvs.tex     30 Nov 2004 15:56:42 -0000
```

```
@@ -163,1 +163,1 @@
```

```
-\subsection*{Différences entre révisions}
```

```
+\subsection*{Différences entre révision}
```

```
$
```

cvs log [OPTIONS] [FICHIERS]

Affiche le journal associé aux FICHIERS:

- ▶ Localisation du fichier dans la base CVS et de la copie locale
- ▶ dernier numéro de révision
- ▶ nom de branche, noms symboliques
- ▶ nombre de révisions
- ▶ pour chaque révision
 - ▶ Numéro de révision
 - ▶ Détail de la révision (date, auteur, ligne ajoutées/supprimées)
 - ▶ Message associé à la révision

Possibilité de choisir d'afficher le journal de certaines révisions
(`-r[REV1] [:[:]] [REV2]`) ou des révisions de certaines dates
(`-D[DATE1] [<] [>] [DATE2]`)

```
$ cvs log -r1.7:1.6 cvs.tex
RCS file: /usr/home/martin/CVSR00T/Presentations/Cvs/cvs.tex,v
Working file: cvs.tex
head: 1.7
branch:
locks: strict
access list:
symbolic names:
    start: 1.1.1.1
    cvs: 1.1.1
keyword substitution: kv
total revisions: 8;      selected revisions: 2
description:
-----
revision 1.7
date: 2004/11/30 16:40:07; author: martin; state: Exp; lines: +28 -17
Ajout des transparents sur cvs history
-----
revision 1.6
date: 2004/11/30 15:56:22; author: martin; state: Exp; lines: +28 -10
Ajout d'un exemple de diff séparé
```

\$

Mode développeur

- ▶ Participe au développement du projet
- ▶ Met à jour la base CVS
- ▶ Synchroniser sa copie de travail pour incorporer les modifications issues d'autres développeurs
- ▶ Effectue des modifications, ajouts ou suppressions de fichiers

cvs commit [OPTIONS] [FICHIERS]

- ▶ Synchronise la base CVS avec les fichiers spécifiés ou l'ensemble du répertoire courant
- ▶ Crée les nouveaux numéros de révision uniquement pour les fichiers modifiés
- ▶ Demande un message de journal avec l'éditeur par défaut
- ▶ Vérifie que les fichiers modifiés localement sont en phase avec la base CVS

- ▶ Option `-m "LOG_MESSAGE"` spécifie le message de journal
- ▶ Option `-f` force la création d'une nouvelle révision
- ▶ Option `-r REVISION` force le numéro de révision

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex          Status: Locally Modified
  Working revision:    1.8      Wed Dec  1 15:01:07 2004
  Repository revision: 1.8      /usr/home/martin/CVSRROOT/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex           Status: Locally Modified
  Working revision:    1.8      Wed Dec  1 15:01:07 2004
  Repository revision: 1.8      /usr/home/martin/CVSR00T/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

```
$ cvs commit -m "Corrections diverses" cvs.tex
```

```
Checking in cvs.tex;
/usr/home/martin/CVSR00T/Presentations/Cvs/cvs.tex,v <- cvs.tex
new revision: 1.9; previous revision: 1.8
done
```

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex           Status: Locally Modified
  Working revision:    1.8      Wed Dec  1 15:01:07 2004
  Repository revision: 1.8      /usr/home/martin/CVSROOT/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

```
$ cvs commit -m "Corrections diverses" cvs.tex
```

```
Checking in cvs.tex;
/usr/home/martin/CVSROOT/Presentations/Cvs/cvs.tex,v <- cvs.tex
new revision: 1.9; previous revision: 1.8
done
```

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex           Status: Up-to-date
  Working revision:    1.9      Wed Dec  1 15:12:05 2004
  Repository revision: 1.9      /usr/home/martin/CVSROOT/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

```
$
```

cvs update [OPTIONS] [FICHIERS]

Permet de synchroniser l'espace de travail avec l'espace de sauvegarde afin prendre en compte les modifications d'autres développeurs

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex          Status: Needs Merge
  Working revision:    1.10    Wed Dec  1 15:23:40 2004
  Repository revision: 1.11    /usr/home/martin/CVSR00T/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex          Status: Needs Merge
  Working revision:    1.10    Wed Dec  1 15:23:40 2004
  Repository revision: 1.11    /usr/home/martin/CVSR00T/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

```
$ cvs update cvs.tex
```

```
cvs update cvs.tex
RCS file: /usr/home/martin/CVSR00T/Presentations/Cvs/cvs.tex,v
retrieving revision 1.10
retrieving revision 1.11
Merging differences between 1.10 and 1.11 into cvs.tex
M cvs.tex
```

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex          Status: Needs Merge
  Working revision:    1.10    Wed Dec  1 15:23:40 2004
  Repository revision: 1.11    /usr/home/martin/CVSRROOT/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

```
$ cvs update cvs.tex
```

```
cvs update cvs.tex
RCS file: /usr/home/martin/CVSRROOT/Presentations/Cvs/cvs.tex,v
retrieving revision 1.10
retrieving revision 1.11
Merging differences between 1.10 and 1.11 into cvs.tex
M cvs.tex
```

```
$ cvs status cvs.tex
```

```
=====
File: cvs.tex          Status: Locally Modified
  Working revision:    1.11    Result of merge
  Repository revision: 1.11    /usr/home/martin/CVSRROOT/Presentations/Cvs/cvs.
  Sticky Tag:         (none)
  Sticky Date:        (none)
  Sticky Options:     (none)
```

cvs add [OPTIONS] [FICHIERS]

L'ajout de fichier à la base se fait en plusieurs étapes:

- ▶ Créer les fichiers dans la copie de travail
- ▶ Signaler à CVS de gérer les nouveaux fichiers

```
$ cvs add FICHIERS
```
- ▶ Exporter dans la base CVS les nouveaux fichiers et les rendre disponibles

```
$ cvs commit FICHIERS
```

Options :

- ▶ `-m MESSAGE` permet d'ajouter un message de description relatif aux FICHIERS
- ▶ `-k MOTCLE` spécifie un mode de substitution de variable

cvs remove FICHIERS

- ▶ Effacer physiquement les fichiers de la copie de travail

```
$ rm FICHIERS
```

- ▶ Dire à CVS que ces fichiers ne doivent plus être considérés

```
$ cvs remove FICHIERS
```

- ▶ Synchroniser la base CVS pour les supprimer effectivement

```
$ cvs commit -m "Remove fichiers" FICHIERS
```

► Modifier la copie de travail

```
$ mv old new
```

► Dire à CVS que le fichier old ne doit plus être considéré

```
$ cvs remove old
```

► Dire à CVS qu'un nouveau fichier new doit être considéré

```
$ cvs add new
```

► Synchroniser la base CVS

```
$ cvs commit -m "old -> new" old new
```

Remarque : attention le numéro de révision de new repart du début (1.1) sans tenir compte du numéro de révision de l'ancien fichier old. On peut préciser le numéro de révision souhaité avec l'option `-r revision`

Gestionnaire de projet

- ▶ Création de nouveaux projets dans la base CVS
- ▶ Historique des commandes
- ▶ Utilisation de noms symboliques
- ▶ Création des différentes branches
- ▶ Diffusion des versions hors CVS

Création de la base CVS

Une base CVS doit d'abord être créée avant de pouvoir être utilisée.

```
$ setenv CVSROOT $HOME/cvsroot
```

Création de la base CVS

Une base CVS doit d'abord être créée avant de pouvoir être utilisée.

```
$ setenv CVSROOT $HOME/cvsroot  
$ cvs -d $HOME/cvsroot init
```

Création de la base CVS

Une base CVS doit d'abord être créée avant de pouvoir être utilisée.

```
$ setenv CVSROOT $HOME/cvsroot  
$ cvs -d $HOME/cvsroot init  
$ ls $CVSROOT  
CVSROOT
```

Création de la base CVS

Une base CVS doit d'abord être créée avant de pouvoir être utilisée.

```
$ setenv CVSROOT $HOME/cvsroot
```

```
$ cvs -d $HOME/cvsroot init
```

```
$ ls $CVSROOT
```

```
CVSROOT
```

```
$ ls $CVSROOT/CVSROOT
```

Emptydir	config	editinfo,v	modules,v	taginfo
checkoutlist	config,v	history	notify	taginfo,v
checkoutlist,v	cvswrappers	loginfo	notify,v	val-tags
commitinfo	cvswrappers,v	loginfo,v	rcsinfo	verifymsg
commitinfo,v	editinfo	modules	rcsinfo,v	verifymsg,v

cvs import -m "message" repertoire module balise

- ▶ Création des différents répertoires nécessaires au projet

cvs import -m "message" repertoire module balise

- ▶ Création des différents répertoires nécessaires au projet

```
$ mkdir projet
```

cvs import -m "message" repertoire module balise

- ▶ Création des différents répertoires nécessaires au projet

```
$ mkdir projet  
$ mkdir projet/src
```

cvs import -m "message" repertoire module balise

- ▶ Création des différents répertoires nécessaires au projet

```
$ mkdir projet  
$ mkdir projet/src  
$ mkdir projet/doc
```

cvs import -m "message" repertoire module balise

- ▶ Création des différents répertoires nécessaires au projet

```
$ mkdir projet  
$ mkdir projet/src  
$ mkdir projet/doc
```

- ▶ Importation dans la base CVS

cvs import -m "message" repertoire module balise

- ▶ Création des différents répertoires nécessaires au projet

```
$ mkdir projet  
$ mkdir projet/src  
$ mkdir projet/doc
```

- ▶ Importation dans la base CVS

```
$ cd projet
```

cvs import -m "message" repertoire module balise

- ▶ Création des différents répertoires nécessaires au projet

```
$ mkdir projet  
$ mkdir projet/src  
$ mkdir projet/doc
```

- ▶ Importation dans la base CVS

```
$ cd projet  
$ cvs import -m "Creation du projet" domaine/projet projet start
```

cvs history fichiers

Affiche l'historique des commandes effectuées sur un fichier

- ▶ -c -> commit
- ▶ -o -> checkout
- ▶ -e -> toutes les actions
- ▶ -a de tous les utilisateurs (pas seulement soi-même)

```
$ cvs history -e cvs.tex
O 2004-11-30 10:05 +0000 martin Presentations/* =Cvs= ~/Work/*
M 2004-11-30 11:02 +0000 martin 1.2 cvs.tex Presentations/Cvs == ~/Work/Cvs
M 2004-11-30 15:05 +0000 martin 1.3 cvs.tex Presentations/Cvs == ~/Work/Cvs
M 2004-11-30 15:06 +0000 martin 1.4 cvs.tex Presentations/Cvs == ~/Work/Cvs
M 2004-11-30 15:42 +0000 martin 1.5 cvs.tex Presentations/Cvs == ~/Work/Cvs
M 2004-11-30 15:56 +0000 martin 1.6 cvs.tex Presentations/Cvs == ~/Work/Cvs
```

Nom symbolique et branche

- ▶ Donne un nom à un état particulier du développement (version diffusée, ...)
- ▶ Affecté à une révision donnée pour chacun des fichiers
- ▶ Peut être utilisé en lieu et place d'un numéro de révision
- ▶ 2 noms sont réservés par CVS :
 - BASE** : fait référence à la révision qui a servi de copie de travail
 - HEAD** : fait référence à la révision la plus récente disponible dans la base CVS
- ▶ Permet de créer et nommer des branches de développement

cvs tag [OPTIONS] NOM [FICHIERS]

Donner le nom symbolique NOM à la révision qui a servi de base aux FICHIERS de la copie de travail

- ▶ -c : vérifie au préalable que la base CVS est bien en phase avec le fichier local
- ▶ -r REVISION : spécifie le numéro de REVISION auquel le NOM sera affecté
- ▶ -d DATE : affecte le NOM à la dernière révision de la DATE

cvs rtag [OPTIONS] NOM REP_BASE_CVS

Donner le nom symbolique NOM à une révision de la base CVS

- ▶ par défaut, attribue le NOM à la dernière révision des fichiers de REP_BASE_CVS
- ▶ `-r REVISION` : spécifie ne numéro de REVISION auquel le NOM sera affecté
- ▶ `-D DATE` : affecte le NOM à la dernière révision de la DATE
- ▶ `-f` : utilise la dernière version en l'absence de concordance avec les options `-r` ou `-f`
- ▶ `-b [-r REVISION]` : crée une nouvelle branche nommée NOM à partir de la dernière révision ou de la révision spécifiée par REVISION

Utilisation de branches de développement

- ▶ `cvs rtag -b [-r REVISION] NOM_BRANCHE REP_BASE_CVS`
crée une nouvelle branche `NOM_BRANCHE` à partir de la dernière révision ou de la révision `REVISION` des fichiers du répertoire `REP_BASE_CVS` de la base CVS
- ▶ `cvs checkout -r NOM_BRANCHE PROJET REP_BASE_CVS`
récupère une copie de travail de la branche nommée `NOM_BRANCHE` du projet `REP_BASE_CVS`
- ▶ `cvs checkout -j MODIF`
fusionne les modifications effectuées jusqu'à `MODIF` dans la copie de travail
- ▶ `cvs checkout -j MODIF1 -j MODIF1`
fusionne les modifications effectuées dans la branche `NOM_BRANCHE` à partir de `MODIF` jusqu'à maintenant dans la copie de travail

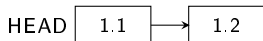
Exemple d'évolution d'un projet

HEAD 1.1

Création du projet et importation dans la base CVS

```
cvs import -m "Importation de PROJET dans CVS" BASE_REP_CVS PROJET Start
```

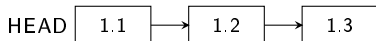
Exemple d'évolution d'un projet



Poursuite du développement et synchronisation de la base CVS

```
cvs commit -m "diverses modifications"
```

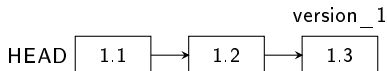
Exemple d'évolution d'un projet



Poursuite du développement et synchronisation de la base CVS

```
cvs commit -m "derniere modif avant 1ere version stable"
```

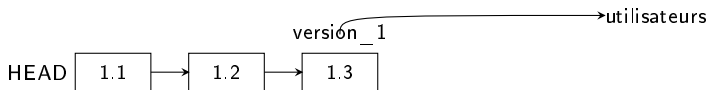
Exemple d'évolution d'un projet



Création d'un nom symbolique pour repérer la version `version_1`

```
 cvs rtag version_1
```

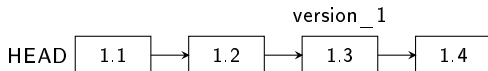
Exemple d'évolution d'un projet



Diffusion auprès des utilisateurs

```
cvs -d $CVSROOT export -r version_1 Projet
```

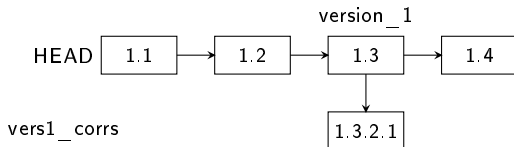
Exemple d'évolution d'un projet



Poursuite du développement et synchronisation de la base CVS

```
cvs commit -m "en avant vers la version_2"
```

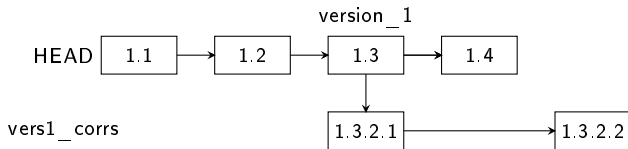
Exemple d'évolution d'un projet



Création d'une branche afin d'incorporer les corrections de la version_1

```
cv$ rtag -b -r version1 vers1_corrections BASE_REP_CVS  
cv$ checkout -r vers1_corrections BASE_REP_CVS
```

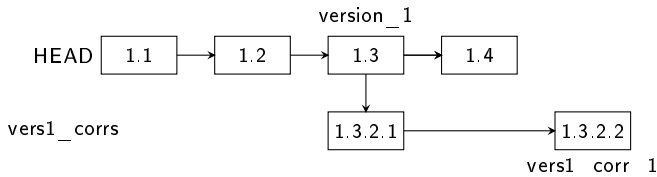
Exemple d'évolution d'un projet



Incorporation de correction dans la branche vers1_corrections

```
cvsv commit -m "diverses correction de la version 1"
```

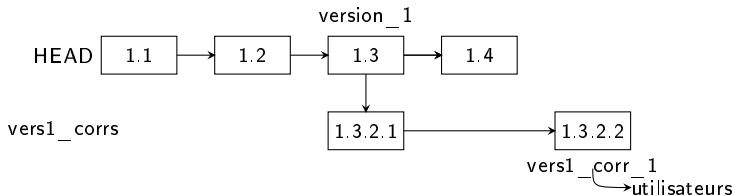
Exemple d'évolution d'un projet



Création d'un nom symbolique pour repérer ces corrections

```
 cvs rtag vers1_corr_1
```

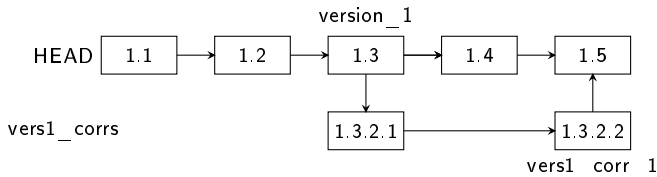
Exemple d'évolution d'un projet



Diffusion auprès des utilisateurs

```
# Mise à jour à partir de la version_1
cvs -d $CVSROOT update -j vers1_corr_1
# ou directement en récupérant la version
cvs -d $CVSROOT checkout -r vers1_corr_1 Projet
```

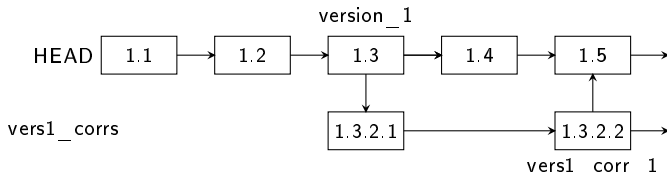
Exemple d'évolution d'un projet



Fusion des correction de la branche vers1_corrections dans la branche principale

```
cv$ checkout -j vers1_corrections BASE_REP_CVS
cv$ commit -m "Fusion des corrections de la version_1"
```

Exemple d'évolution d'un projet



Poursuite du développement ...

Livraison de projet

- ▶ Ensemble de fichiers : `cvs export [OPTIONS] REP_BASE_CVS`
 - ▶ `-r REVISION` : correspondant à la révision REVISION
 - ▶ `-D DATE` : dernière révision de la date DATE
- ▶ Ensemble de différences : `cvs rdiff [OPTIONS] REP_BASE_CVS`
 - ▶ `-r REVISION` : différence entre la REVISION et la dernière révision (HEAD)
 - ▶ `-r REV1 -r REV2` : différence entre deux révisions REV1 et REV2
 - ▶ `-D DATE` : différence depuis la date DATE
 - ▶ `-D DATE1 -D DATE2` : différence entre DATE1 et DATE2

Utilisation de la commande `patch` pour appliquer les différences.

- ▶ Documentation intégrée : info cvs
- ▶ <http://www.cvshome.org/>
- ▶

CVS Pocket Reference, 2nd Edition
By Gregor N. Purdy
2nd Edition August 2003
Series: Pocket References
ISBN: 0-596-00567-9
80 pages

