Subversion

Michel Jouvin

LAL/Orsay

jouvin@lal.in2p3.fr

Agenda

- Pourquoi Subversion ?
- Rappel CVS
- Les révisions
- Les branches et les tags
- Le contrôle d’accès
- Restructurer un repository
- Web browsers

Pourquoi Subversion ?

- Successeur de CVS pour résoudre ses limitations
  - Basé sur les concepts et l'expérience CVS
  - Reprend les commandes CVS autant que possible
  - Nouveau design des repositories : pas de compatibilité
- Problèmes CVS adressés
  - Prise en compte des répertoires, symlinks…
  - Renommage de composants, restructuration de repository
  - Accès distribué et contrôle d'accès (ex : ACL dans une branche)
  - Performances : structure des repositories
  - Plus (+) d'opérations déconnectées

Rappel CVS…

- Chaque fichier à son cycle de révision
  - Révision d'un fichier incrémenté lors de sa modification
  - Basé sur la révision de la branche
- 2 « dimensions » : les révisions et les tags… + les branches
  - Les revisions : numéro associé à un fichier, avec un format a.b.c.d
  - Tag : nom désignant un ensemble de fichier/révision
  - Branche : extension d'un tag
    - Réinitialisation du numéro de révision des fichiers
    - Branche principale est la première créée (révision 1.x.y.z)

… Rappel CVS

- Accès distribué est un hack…
  - Pserver : password « en clair »
  - Contrôle d’accès très sommaire, difficile de protéger une partie de repository
  - Repose essentiellement sur les protections du file system
- Repository non restructurable
  - Perte du log si rename d’un fichier (delete + add)

Subversion : Repository et Révisions

- Repository est une « base de donnée »
  - Format « opaque »
  - Meilleures performances
  - Commit « atomique »
- Commit = nouvelle révision du **repository**
  - Pas de révision associée à un fichier ou une branche
  - Révision : snapshot de l'ensemble des fichiers
    - Delta par rapport à la révision précédente du repository
    - Incrémentée à chaque commit à partir de 1
    - Fonctionnellement équivalent à un tag CVS

Subversion : Copy, Rename, …

- Subversion permet de dupliquer, renommer fichiers et répertoires
  - A la fois dans l'espace de travail et directement dans le repository
  - Dans l'espace de travail, utiliser les commandes svn au lieu des commandes shells
    - 'svn copy', 'svn mv'
  - Conservation de l'historique

Subversion : les Branches

- 1 branche = 1 répertoire de l'espace de travail
  - On peut en créer à n'importe quel niveau de la hiérarchie
  - Créer une nouvelle branche vide : svn mkdir
  - Créer une nouvelle branche à partir d'une existante : 'svn cp'
  - Changer la branche associée à un répertoire de travail : 'svn switch' (implique un 'svn update')
  - Pas de lien symbolique entre branches...
    - Une branche créée par 'cp' ne voit pas les nouvelles révisions de la branche d'origine.

Subversion : les Tags

- Ils n’existent pas car ils sont inutiles
  - 1 tag est une révision du repository
  - Il est possible de les ‘imiter’ en créant une branche correspondant au tag (‘svn cp’)
  - Plus de ‘sticky tag’…
- Une habitude courante est de créer 3 branches :
  - trunk : la branche principale
  - branches : la branche qui contiendra… les branches
  - tags : la branche qui contiendra… les tags
- C’est une habitude, absolument pas une obligation
  - Peut être créé au niveau du repository ou d’une branche… ou les 2 !

Historique d'un fichier

- svn log [--stop-on-copy] [-v]
  - Avec –v, permet de savoir de quelle branche est dérivée la branche courante (ou le fichier de la branche courante)
  - --stop-on-copy donne l'historique jusqu'à l'opération de copie qui a créé la branche
  - Equivalent de 'cvs log'

Connaître l'état de l'espace de travail

- svn status [–n]
  - Equivalent 'cvs –n update'
- svn info
  - Indique d'où provient le répertoire de travail courant
  - Indique la révision de la dernière synchronisation avec le repository
    - dernier 'svn merge' ou 'svn update'

Merge/Update et conflits

- Merge : update dont la source n'est pas une autre révision du repository mais la différence entre 2 autres révisions
  - Fonctionalité la plus précieuse de CVS/SVN
  - N'affecte que l'espace de travail
- En cas de conflit lors d'un merge, 4 fichiers créés
  - Récupérer la version la plus appropriée ou reconstruire la version ad-hoc
  - 'svn resolve fichier' pour indiquer que le conflit a été résolu
  - Pas de commit possible tant qu'il y a un conflit

Restaurer un fichier

- Possible de restaurer la version de base d'un fichier : 'svn revert fichier'
  - La version de base est la version correspondant à la dernière synchro avec le repository (update ou merge)
  - Possible même en mode déconnecté : copie des fichiers de base dans les répertoires .svn

Les Clients SVN

- Il en existe de nombreux pour toutes les plateformes
- Le client de base Unix reprend les commandes CVS autant que possible…
- Sur Windows et Macintosh, client intégré au gestionnaire de fichier
  - Windows : TortoiseSVN (déployé automatiquement sur demande)
  - Macintosh : SCPlugin
- Plusieurs clients Web (browser only) : WebSVN, Trac, SVNBrowser …

Divers

- Support des fichiers binaires
  - Nouvel algorithme de calcul des différences
- API pour de nombreux langages
  - Permet l'écriture d'application utilisant SVN sans passer par le parsing de commande shell
    - Exemple : Trac
  - C/C++, Java, Python, Perl, (PHP), …