Concurent Versioning System - CVS

1) Introduction :

Concurent Versioning System (CVS) est un système de contrôle des versions d'un logiciel comprenant un serveur de code source et de plusieurs outils.
Les systèmes de contrôle de version sont généralement utilisés dans le cycle de développement d'un logiciel pour suivre et coordonner les changements du code source au sein de l'équipe de développement. Un serveur CVS est adapté à la spécificité du développement logiciel avec des fichiers au format texte (compilation), avec un grand nombre de fichiers (modularité), avec des modifications simultanés (parallélisation) et avec des livraisons multiples (historisation).

Des bugs se glissent parfois quand le programme est modifié ;-) On peut ne détecter le bug que longtemps après avoir fait les modifications. Avec CVS, on extrait facilement les anciennes versions pour voir exactement quel changement a causé le bug. CVS stocke toutes les versions d'un fichier dans un fichier unique d'une manière intelligente qui stocke seulement les différences entre les versions. CVS utilise un et un seul dépôt (repository) qui centralise tous les fichiers du développement logiciel.

Chaque développeur travaille dans son propre espace de travail ou "bac à sable" (sandbox), et CVS fusionne les changements quand chaque développeur a terminé. CVS autorise un nombre non limité de copies locales de chaque fichier permettant à chaque développeur de disposer de ses propres fichiers sources sur son poste de travail.

CVS gère la concurrence sans verrou. C'est à dire qu'il est possible à plusieurs développeurs d'intervenir sur le même fichier source et d'apporter chacun ses propres modifications. Dans la mesure du possible CVS gère automatiquement la fusion des modifications.

2) Cycle de vie du développement logiciel :

Les spécificités du développement logiciel sont les suivantes :

- Grande majorité de documents au format texte
- Production très modulaire : un projet logiciel est constitué au minimum de plusieurs dizaines de fichiers d'où un problème de cohérence
- Parallélisation du travail : les fichiers sont modifiés simultanément par plusieurs développeurs d'où un problème de coordination ; des évolutions simultanées de plusieurs versions d'un même produit logiciel sont réalisables d'où un problème de consistance
- Retour en arrière et comparaison des versions sont nécessaires d'où un problème d'historisation et d'archivage.

La maîtrise de ces spécificités passe par l'établissement de règles d'organisation, la réalisation d'un plan de configuration logiciel, et par l'utilisation d'outils adaptés.

CVS va accompagner le développeur durant toutes les phases classiques du développement logiciel en partant des spécifications, en passant par la conception, le codage, le test unitaire, la validation et en arrivant finalement à la maintenance. Ces phases sont itératives dans le temps, CVS est solicité à chaque tour de boucle.

Chaque modification d'une spécification, d'un algorithme de conception, chaque correction d'erreur provoque une nouvelle version du code source. On peut bien sûr enregistrer chaque version de chaque fichier que l'on crée. Cependant ceci gaspille une énorme quantité d'espace disque. CVS, lui, stocke toutes les versions d'un fichier dans un fichier unique d'une manière intelligente qui n'enregistre seulement que les différences entre les versions.

CVS est d'une grande aide aussi si un groupe de personnes travaille sur le même projet. C'est très facile d'écraser les changements des autres même en faisant très attention. CVS résout ce problème en isolant les programmeurs les uns des autres. Chaque développeur travaille dans son propre bac à sable (sandbox), et CVS fusionne le travail quand chaque développeur a terminé.
CVS utilise un et un seul dépôt (repository) qui centralise tous les fichiers du développement logiciel. Il autorise un nombre non limité de copies locales de chaque fichier permettant à chaque développeur de disposer de ses propres fichiers sources sur son poste de travail.

3) Fonctionnement :

CVS utilise un mode de synchronisation entre les copies locales et le dépôt par l'intermédiaire des trois actions suivantes :

- Checkout : obtention d'une copie locale du dépôt,
- Update : mise à jour de la copie locale avec conservation des modifications personnelles,
- Commit : intégration dans le dépôt des modifications apportées à la copie locale.

Pour cela, CVS possède les deux fonctionnalités suivantes :

- Historisation : permet de créer et de connaître à tout moment, pour chaque élément du logiciel, les modifications qui ont été apportées par qui et pourquoi.
- Archivage : permet d'obtenir à tout moment une copie de l'ensemble des éléments du logiciel.

CVS a une méthode de travail commune à la plupart des autres systèmes de contrôle de version par le fait que les développeurs peuvent éditer le même fichier en même temps.
D'abord on doit "Emprunter" (Checkout) une version d'un code source depuis le référentiel dans une copie locale dans votre ordinateur. Cette copie locale est appelée un bac à sable (sandbox).
On édite alors simplement les fichiers que l'on souhaite changer. Quand les modifications sont terminées il faut "Valider" (Commit) les changements dans le référentiel.
Parfois il est nécessaire d' "Ajouter" (Add) des nouveaux fichiers et supprimer des fichiers qui ne sont plus nécessaires.
Si quelqu'un d'autre a changé le même fichier pendant que l'on a travaillé dessus, alors la validation va échouer. Il est nécessaire alors d' "Actualiser" (Update) tous ses fichiers code source depuis le référentiel. Ceci fusionnera automatiquement les changements des autres développeurs dans sa copie du fichier.
Parfois CVS ne peut pas le faire automatiquement, par exemple si la même ligne de code a été changée par deux développeurs. On appelle cela un Conflit. CVS met les deux versions du code conflictuel dans le fichier, en marquant chacune d'elles. Il faut alors éditer manuellement le fichier pour résoudre le conflit avant de pouvoir valider les changements.

Le mois prochain le serveur et le client CVS.

Pascal Pignard, mai 2006.