Versioning

Définition

Sources : Site du zéro et manuel utilisateur de TortoiseSvn chez TortoiseSVN

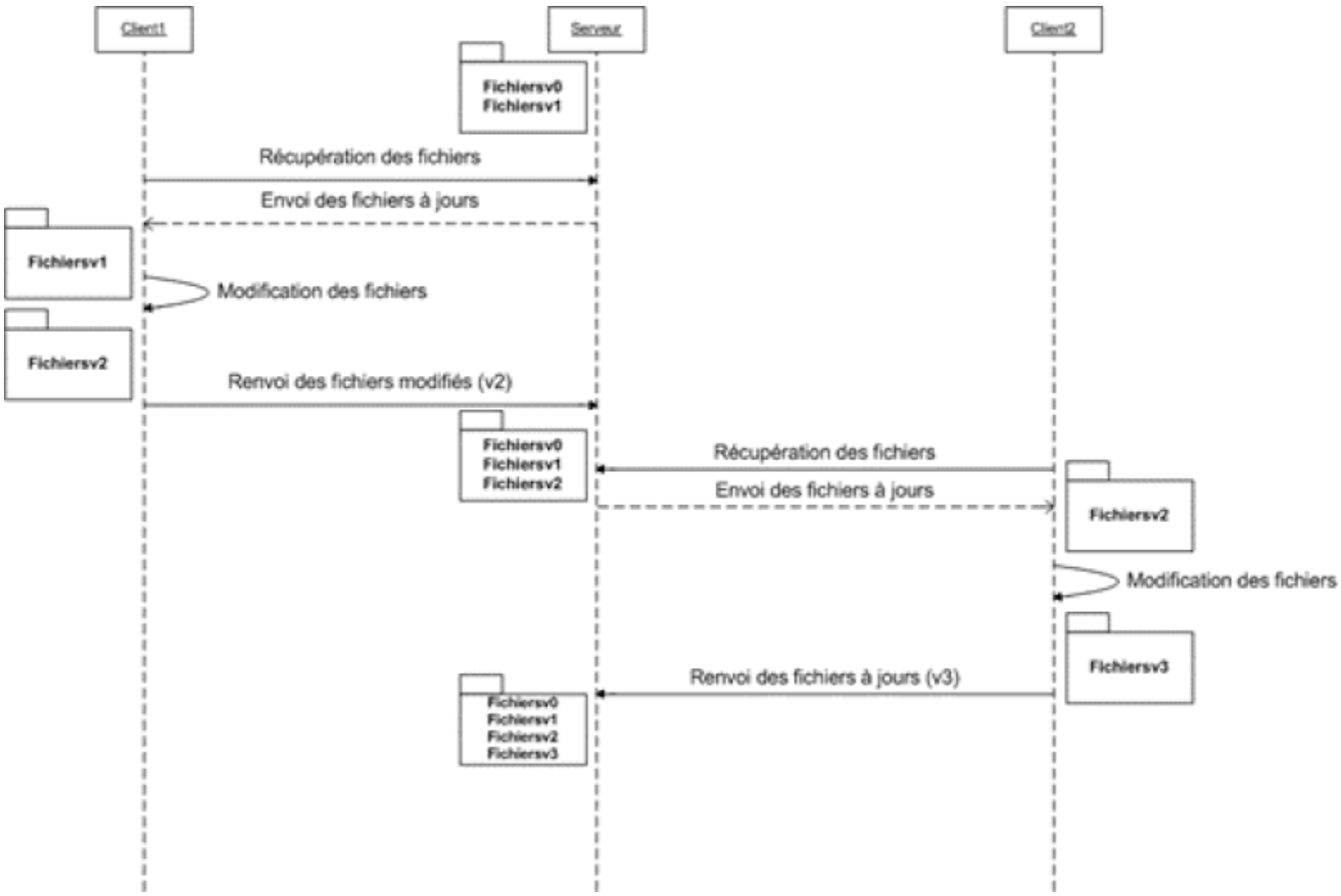
Le contrôle des versions d'un logiciel répond à deux problématiques :

- D'abord, le développement d'une application informatique implique la création et la modification de nombreux fichiers. Au fil du temps, ces fichiers sont modifiés (enrichis, supprimés, ...). Or, il arrive souvent que l'on veuille revenir à l'état précédent d'un fichier, afin par exemple de voir quelle modification a entraîné l'apparition d'un bug, ou de remettre en service du code que l'on avait supprimé. Un système de gestion de versions garde l'intégralité des états successifs de chaque fichier, permettant à tout instant de revenir en arrière. En bonne pratique, tout changement d'état est accompagné d'un commentaire.
- Ensuite, le développement d'une application informatique implique généralement plusieurs personnes simultanément. Dès lors que ces personnes sont amenées à intervenir sur les mêmes parties du projet, il convient de gérer correctement les modifications concurrentes des fichiers. Un système de gestion de versions joue ce rôle de médiation.

Répartition des rôles entre client et serveur

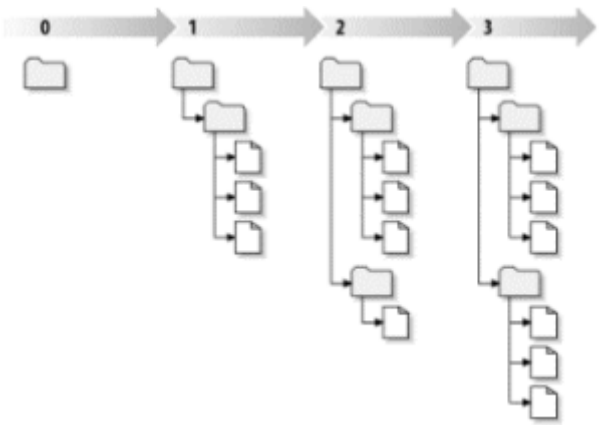
Le principe est le suivant :

- Le serveur conserve en mémoire toutes les versions de tous les fichiers, ainsi que les changements de l'arborescence des répertoires, comme l'ajout, la suppression et le réarrangement des fichiers et des dossiers. La zone où sont archivées, sur le serveur, les modifications successives s'appellent le référentiel (repository, en anglais) ou dépôt. Il n'est pas possible de travailler directement sur les fichiers du dépôt. C'est la raison pour laquelle il faut obligatoirement un client dans le système.
- Le client, lui, va télécharger du serveur les fichiers à jour afin de pouvoir travailler dessus localement. Une fois qu'il aura fini d'apporter ses modifications, il va télé-verser la nouvelle version des fichiers sur le dépôt de sorte qu'ils puissent être, à leur tour, modifiés par d'autres intervenants.

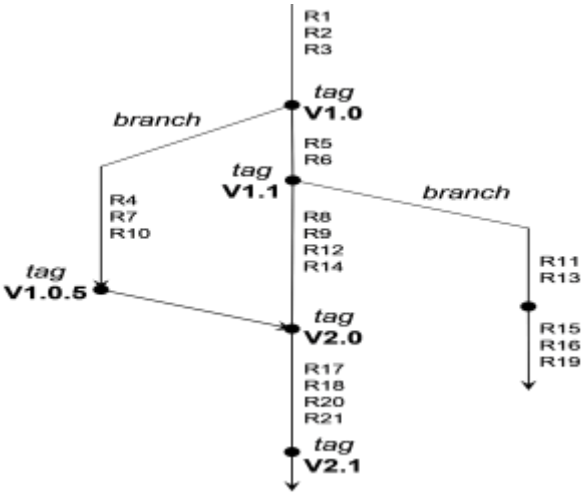


Les révisions

Dans une copie locale de travail, il est possible de changer le contenu des fichiers existants, ou encore créer, supprimer, renommer, copier des fichiers et des répertoires et ensuite livrer le jeu complet de changements comme une unité. Sur le serveur, chaque livraison est traitée comme une transaction atomique : tous les changements de la livraison ont lieu, ou aucun n'a lieu. L'acceptation d'une livraison par le référentiel crée un nouvel état de l'arborescence du système de fichiers, appelé une révision. À chaque révision est assigné un numéro (soit un numéro d'ordre, soit une signature sous la forme d'un hash).
Le référentiel peut être considéré comme un arbre multidimensionnel : les branches forment une arborescence, chaque révision représente une arborescence de fichiers.



Brancher / Étiqueter



La capacité d'isoler des changements sur une ligne de développement particulière est une des fonctionnalités des systèmes de contrôle de versions. Cette ligne de développement particulière est connue sous le nom de branche (branch). Les branches sont souvent utilisées pour expérimenter de nouvelles fonctionnalités sans déranger la ligne de développement principale. Dès que la nouvelle fonctionnalité est suffisamment stable, alors la branche de développement peut être fusionnée avec la branche principale (le tronc). Une autre fonctionnalité des systèmes de contrôle de versions est la capacité de marquer des révisions particulières, par exemple une version à déployer (livrables). Il est alors possible de recréer à tout moment cette version d'application sans effort. Ce processus est connu comme l'étiquetage (tag). Cette fonctionnalité est largement utilisée afin de marquer à intervalles plus ou moins réguliers des versions d’une application dans la ligne de développement.

Organisation

Sur un serveur SVN, le projet, les Branches et les Tags sont stockés dans des dossiers spécifiques nommés res-pectivement « trunk », « branches », et « tags ». Ce n’est qu’une convention de nommage que vous êtes libres de respecter ou d’adapter, mais les bonnes pratiques vous conduiront naturellement à adopter ce nommage. L’intérêt des étiquettes sous Subversion est d’utiliser des noms symboliques plutôt que des numéros de révi-sions pour se référer à un état précis, comme par exemple ’release-1.1’, plutôt que ’488’. Un nom symbolique permet de revenir facilement à une version identifiée. Pour créer un tag, il suffit de copier l'état actuel d'une version de développement dans un sous-répertoire du dépôt. Les règles suivantes sont communément admises :

- Liste à puceLes différents tags correspondent chacun à un sous-répertoire, lui-même contenu dans un sous-répertoire nommé tags du projet ;
- On ne « commit » pas dans un tag ;

En fonction de l’ampleur et du nombre de projets contenus dans votre dépôt, l’emplacement de ces trois dos-siers peut varier. Il existe en fait deux formes recommandées en fonction de vos besoins :



1. Commun à tous les projets

```
trunk/
  projetA/
  projetB/
tags/
  projetA/
    version1/
  projetB/
branches/
```

2. Propre à chaque projet

```
projetA/
  trunk/
  tags/
  branches/
projetB/
  trunk/
  tags/
  branches/
```

Quelle que soit l'architecture choisie pour les dossiers de base, l'utilisateur crée tous les dossiers intermédiaires librement. Par exemple, la branche « germanVersion » pourra être stockée dans le dossier branches/germanVersion.
À consulter : http://www.lacl.fr/gava/cours/M2/IngLog/annexe3.pdf

From:
https://wiki.siochaptalqper.fr/ - **Wiki SIO Chaptal**

Permanent link:
https://wiki.siochaptalqper.fr/doku.php?id=bloc2:prog:gen:versioning&rev=1730814308

Last update: **2024/11/05 14:45**