Gestionnaire de version
Git et Netbeans

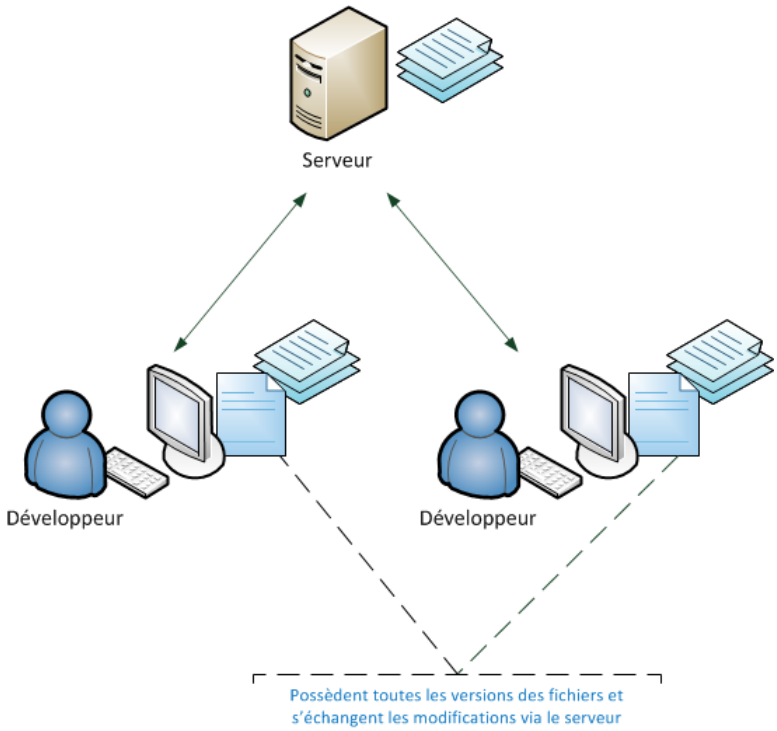
1 Un gestionnaire de version...

Un CVS (sigle de Concurrent Versions System), il y en a plusieurs, on retient GIT est un système de gestion de versions des fichiers, c'est le successeur de subversion (abréviation SVN). [1]

Citation : Wikipedia

"Un logiciel de gestion de versions est un logiciel de *gestion de configuration*, il agit sur une arborescence de fichiers afin de conserver toutes les versions des fichiers, ainsi que les différences entre les fichiers. Ce système permet par exemple de mutualiser un développement. Un groupe de développeurs autour d'un même développement se servira de l'outil pour stocker toute évolution du code source."

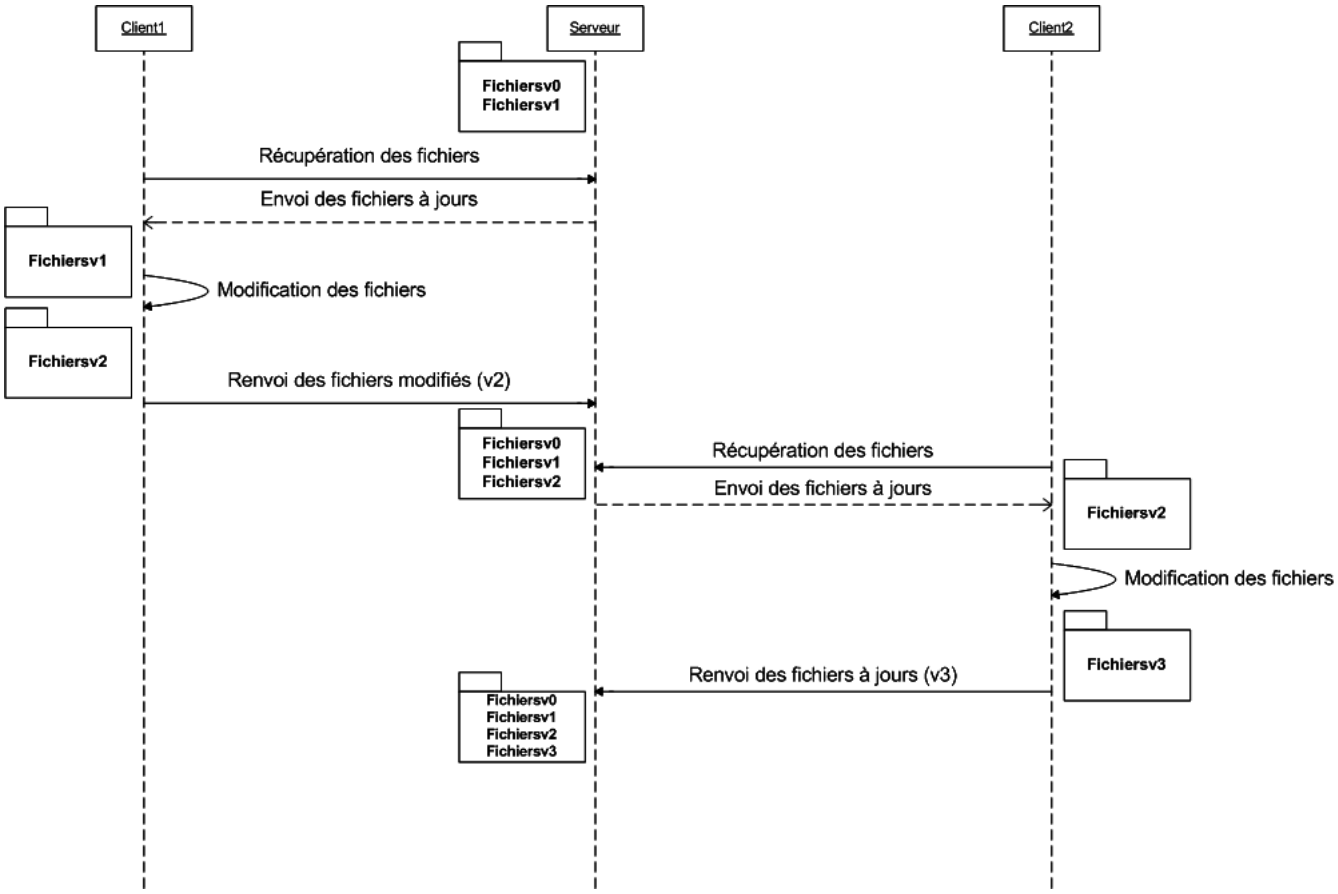
Principe :



Le client va télécharger les fichiers à jour sur le serveur afin de pouvoir travailler dessus localement. Une fois qu'il aura fini d'apporter ses modifications, il va envoyer la nouvelle version des fichiers au serveur qui va les stocker pour la personne suivante qui voudrait les utiliser.

1. Quand on lui a demandé pourquoi il avait appelé son logiciel "git", qui est à peu près l'équivalent de "connard" en argot britannique, Linus Torvalds a répondu "je ne suis qu'un sale égocentrique, donc j'appelle tous mes projets d'après ma propre personne. D'abord Linux, puis Git.".

Exemple :



Un logiciel de gestion de version a donc deux utilités principales :
- suivre l'évolution d'un code source, pour retenir les modifications effectuées sur chaque fichier et être ainsi capable de revenir en arrière en cas de problème ;
- travailler à plusieurs, sans risquer de se marcher sur les pieds. Si deux personnes modifient un même fichier en même temps, leurs modifications doivent pouvoir être fusionnées sans perte d'information.

2 Les commandes principales (à noter sur les schémas)

2.1 Clone

Clone est la demande du client de récupération de tout le dépôt. Le serveur va lui renvoyer tous les fichiers. Le répertoire ou sont stockées les données sous contrôle de version est appelé "dépôt" ou "repository" ou pire "référentiel". Avec Git il y a donc un dépôt local, et un dépôt sur le serveur. Le dossier local (sur le client) est appelé copie de travail.

2.2 Commit

On travaille sur les fichiers de la copie de travail, et quand une fonctionnalité marche, on l'enregistre sur le dépôt local par la commande Commit. On peut commiter un seul fichier, ou toute une arborescence.

Il est indispensable de bien noter un commentaire pour chaque commit de façon à comprendre les mises à jour effectuées.

2.3 Push

En fin de journée (ou de TP), on envoie sur le serveur les travaux qu'on a commité. C'est la commande push (pousser).

Si on travaille en équipe, et qu'un de nos camarade a poussé avant nous, le serveur va refuser notre push : il a peur que ça casse le travail de l'autre. Il faut d'abord intégrer les modifs de l'autre à notre travail local avec la commande pull.

2.4 Pull

Récupère les données du serveur, et merge les modifs faites par les autres dans notre dépôt local. S'il y a eu des modifications sur un fichier que l'on a nous même modifié, Git ne sait pas quoi faire il y a un *conflit*.

3 Les conflits

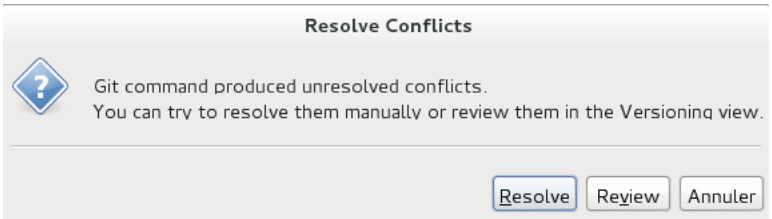
Petit scénario classique :

Fin du taf, j'ai bien bossé, je fais mon dernier commit (dépôt local donc).

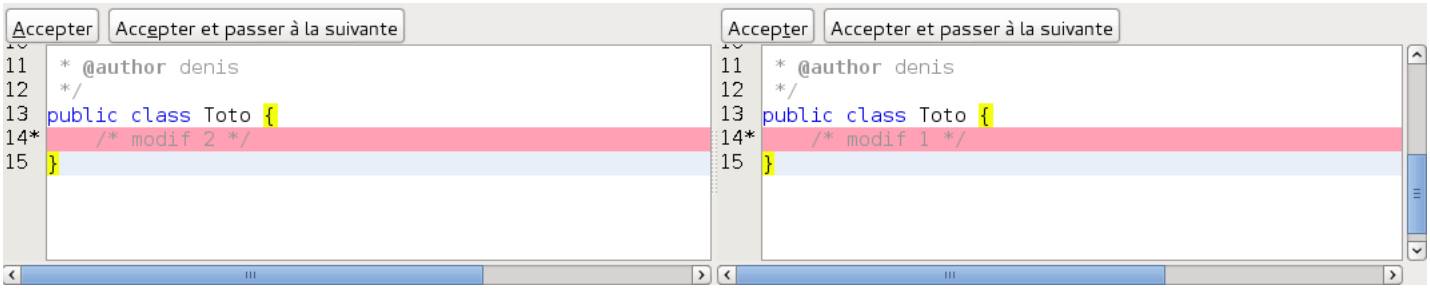
Puis je push : ça sauvegarde sur le serveur et les autres auront accès à mon travail.

Résultat (si quelqu'un a poussé avant moi sur le projet) :

Alors je dois merger les modifs dans mon dépôt local, je fais un pull. Pas de bol on a bossé sur le même fichier :



Je clique sur Resolve :



et je choisis quelles modifs je garde.

Enfin, commit, pull et push ... Ouf.

4 Mise en place

4.1 Création du projet sur SourceForge

https ://sourceforge.net/user/registration
- remplir le formulaire d'inscription
- récupérer le mail qui vous est envoyé pour confirmer votre inscription
- se connecter
- cliquer en haut à droite sur "me" puis "profile"
- cliquer sur "add project"
  - name : exemple "gsbphp-git-denis"
  - garder l'url par défaut
  - sélectionner les services suivants : Wiki, Files&Stats, Tickets, Git
  - ajouter votre camarade dans les admins du projet
  - récupérer dans l'onglet "code" l'URL associée à votre nom (chacun la sienne). Par exemple vous trouvez la commande suivante, l'URL est la partie soulignée.

```
git clone ssh://denisbrodard@git.code.sf.net/p/gsbphp-git-denis/code gsbphp-git-den
```

4.2 Dans Netbeans

- récuperer sur le campus le zip "gsbmvc", le décompresser

Aller dans le menu "Equipe" (ou team) -> Git -> Initialise Repository
- C'est le dossier dans lequel Git va gérer le projet local. Vous devez TOUJOURS faire un nouveau dossier et le nommer gitCeQueVousVoulez. Ça commence par git, et vous voyez tout de suite que c'est git qui s'en occupe. C'est dedans que vous mettrez votre projet NetBeans.
- créer le projet, ici un projet PHP.
- copier dedans le répertoire gsbmvc.
- team remote push
- remplir l'URL, le username, le mdp sourceForge
- pour le copain de l'équipe : git -> clone, idem TOUJOURS répertoire gitQuelqueChose, ne pas créer le projet automatiquement, l'ouvrir ensuite
- ... au boulot