Gestion de version avec Subversion - TP

Travailler en local

Création et initialisation du dépôt

Pour commencer, nous allons simplement travailler en local, sans coopération avec d'autres développeurs. Créez un nouveau répertoire (par exemple `repo`) qui contiendra le dépôt Subversion. Initialisez ce dépôt avec l'aide de la commande `svnadmin`. À noter que le répertoire contenant le dépôt ne doit pas être modifié manuellement, mais uniquement au travers de Subversion.

Une fois le dépôt initialisé, vous pouvez y faire référence dans toute commande *svn* en utilisant l'adresse `file:///chemin/complet/vers/le/repertoire/repo/`. Vous pouvez par exemple essayer la commande `svn ls` pour visualiser le contenu du dépôt, et constater qu'il est actuellement vide.

Créez maintenant un répertoire pour notre projet, nommé `helloworld`, ainsi que les sous-répertoires `trunk`, `branches` et `tags` qui constituent l'organisation classique d'un dépôt Subversion.

Récupérez à l'adresse `http://thomas.enix.org/pub/conf/ups2010/helloworld-0.1.tar.gz` le code source du projet `helloworld`. Décompressez l'archive, compilez en tapant `make`, puis testez l'application compilée (qui devrait afficher un simple « `Hello World` » à l'écran).

Importez maintenant ce projet dans le dépôt Subversion en utilisant la commande `svn import`, de manière à ce que le répertoire `helloworld/trunk/` du dépôt contienne la dernière version du code source. Faites attention à n'importer dans le dépôt que le code source et la documentation du projet : les fichiers résultants d'une compilation ou générés automatiquement ne doivent pas être versionnés !

Utilisation du dépôt

Notre projet est maintenant importé, mais le répertoire courant n'est pas une copie de travail nous permettant d'utiliser les fonctionnalités du gestionnaire de version. Avec `svn checkout`, créez une copie de travail de ce projet. Une fois cette copie de travail récupérée, vous disposerez de tous les fichiers du projet. Vous noterez également la présence d'un répertoire caché `.svn/` dans chaque répertoire de votre copie de travail. Ce répertoire caché contient des méta-données utilisées par Subversion.

Modifiez la chaîne affichée par l'application (« `Bonjour Monde` » à la place de « `Hello World` » par exemple). Vérifiez que la modification est correcte (compilation puis exécution de l'application) puis visualisez les modifications à l'aide de `svn status` et `svn diff`. Enfin, committez cette modification dans le dépôt et visualisez avec `svn log` la liste des modifications.

Pour finir, créez un fichier `README` contenant la documentation du projet. Visualisez les modifications avec `svn status` et `svn diff`, puis ajoutez le fichier au dépôt avant de le committer.

Collaborer via Subversion

Initialisation du dépôt

La gestion de version prend tout son intérêt pour collaborer à plusieurs sur un même projet. Pour cela, le dépôt doit être rendu accessible à tous les participants. Pour ce TP, nous allons utiliser un serveur HTTP. À l'adresse `http://humanoidz.org/repo-svn/` se trouve un dépôt déjà initialisé. En entrant cette adresse dans un

navigateur Web, vous disposez d'un accès en lecture seule au dépôt, qui vous permet de visualiser le contenu de la dernière version des fichiers. À l'adresse http://humanoidz.org/trac/ le gestionnaire de projets Trac est installé, il contient un Wiki, un système de suivi de tickets ainsi qu'un outil de visualisation d'un dépôt Subversion. N'hésitez pas à l'utiliser au fil du TP pour visionner le contenu du dépôt Subversion et naviguer dans l'historique.

L'accès en écriture est disponible au travers des comptes `user1`, `user2`, `user3`... `user20`, dont les mots de passe sont respectivement `pass1`, `pass2`... `pass20`. Arrangez-vous pour utiliser un compte qui n'est pas utilisé par une autre personne, et constituez une équipe de deux ou trois personnes pour travailler sur le projet.

Au sein de cette équipe, une personne va créer un répertoire dans le dépôt pour héberger le projet de l'équipe, peu importe son nom (mais chaque équipe doit prendre un nom différent). Les sous-répertoires traditionnels `trunk`, `branches` et `tags` doivent également être créés. Le projet `helloworld` sera importé dans le sous-répertoire `trunk`.

Ensuite, chaque membre de l'équipe récupère une copie de travail du projet `helloworld` depuis le dépôt. Un des développeurs créé un tag pour marquer la version 0.1 du projet, en utilisant `svn copy` (le tag pourra s'appeler `v0.1` par exemple).

Collaboration et résolution des conflits

Deux personnes de l'équipe font simultanément une modification dans deux fichiers différents (par exemple une modification dans le `Makefile` et une autre dans le code source), et committent. Cela ne pose pas de problème, et les membres de l'équipe peuvent récupérer la dernière version en utilisant `svn update`.

Pour tester la résolution des conflits, les deux personnes de l'équipe font simultanément une modification au même endroit dans le même fichier (par exemple une personne change « `Hello World` » en « `Bonjour Monde` » et une autre change cette chaîne en « `Hola Mundo` »). Quand le second membre de l'équipe va vouloir committer sa modification, cela sera impossible car le fichier n'est pas à jour avec le dépôt. `svn update` permettra de mettre à jour le fichier, mais laissera apparaître un conflit, qu'il faudra résoudre avant de committer la solution définitive.

Une fois le conflit résolu, créez un nouveau tag `v0.2` pour donner un nom à cette nouvelle version du projet.

Branche

Pour finir, un des développeurs de l'équipe créé une branche, pour développer une nouvelle fonctionnalité : l'internationalisation de notre application (dans notre cas, parler d'internationalisation sera un bien grand mot, mais il s'agit d'un exemple visant à découvrir les fonctionnalités de la gestion de version). Nous allons donc travailler sur le mode de la branche de développement.

Une fois la branche créée, un autre développeur en récupère une copie de travail, et implémente la fonctionnalité : quand la chaîne « `fr` » est passée en argument à l'application, elle affiche « `Bonjour Monde` », quand la chaîne « `en` » est passée à l'application, elle affiche « `Hello World` ». La fonctionnalité peut si nécessaire être implémentée en plusieurs étapes, chaque étape correspondant à un commit dans la branche de développement.

Une fois la fonctionnalité totalement implémentée, un membre de l'équipe se charge de fusionner cette branche dans la branche principale en utilisant la commande `svn merge`.

Références

- Subversion book, http://svnbook.red-bean.com/
- Supports du cours, http://thomas.enix.org/pub/conf/ups2010/gestion-version.odp et http://thomas.enix.org/pub/conf/ups2010/gestion-version.pdf