

Créé pour simplifier le travail à plusieurs sur des projets de programmation, un système de gestion de versions tel que Subversion peut se révéler précieux pour vos fichiers importants même si vous êtes le seul à les modifier.

Qui n'a pas de tels fichiers sur son disque dur ? ~~article_v4.tex, version envoyée pour relecture.pdf, proftpd.conf.ancien, these_20_sept_05.tex~~... Évidemment, au bout d'un moment, il devient impossible de s'y retrouver : on est submergé par le nombre de fichiers de sauvegarde, on ne sait plus à quelle occasion on a créé un fichier donné, on ne peut pas voir facilement à quels changements correspond le fichier ~~article_v4.tex~~ par rapport au fichier ~~article_v5.tex~~... Bref, tout le monde en conviendra, ce genre de sauvegardes révèle vite ses limites.

Alors qu'un programmeur trouvera tout naturel d'utiliser un système de gestion de versions pour la moindre ligne de code, cette idée pourra ne pas venir à l'esprit d'une personne qui passera de longs mois à rédiger un rapport de stage ou même une thèse.

Cet article a pour but de vous montrer que Subversion peut aider toutes les personnes qui consacrent beaucoup de temps et d'énergie à la création de fichiers textes de toutes sortes. Les systèmes de gestion de versions tels que Subversion ont été créés pour faciliter le développement collaboratif à travers le réseau.

Mais ils peuvent très bien être utilisés par une seule personne pour conserver l'historique de fichiers qui ne contiennent pas de code source.

Subversion sait gérer les fichiers binaires (images, PDF, fichiers OpenOffice...), mais toute sa puissance se révélera avec des fichiers textes (LaTeX, Docbook, pages web, feuilles de style, code source, fichiers de configuration...).

Principe de fonctionnement



- De la copie de travail vers le référentiel : on parle de commit (de l'anglais « confier, remettre quelque chose à quelqu'un ». On effectue cette action lorsque l'on a modifié la copie de travail et que l'on veut transmettre ces modifications au référentiel.
- Du référentiel vers la copie de travail : il s'agit d'un update. Cette action est utilisée lorsque l'on travaille à plusieurs sur un même projet pour intégrer à sa copie de travail les modifications apportées par les autres utilisateurs. Nous verrons plus loin qu'elle peut aussi servir lorsqu'on est seul à travailler sur un projet.

La gestion des versions à proprement parler se fait donc dans le référentiel. Les fichiers de la copie locale sont identiques à ceux que vous avez lorsque vous n'utilisez pas de système de version.

La seule différence est la présence d'un répertoire `.svn` à tous les niveaux de votre arborescence. Ce répertoire contient des informations comme la localisation du référentiel, le numéro de version de la copie locale, etc. ainsi qu'une version non modifiée des fichiers de la copie locale. Il n'y a aucune bonne raison de modifier le contenu de ce répertoire.

Mise en route

Maintenant que nous avons vu les grandes lignes du fonctionnement de Subversion (qui sont les mêmes que d'autres systèmes de gestion des versions, comme CVS), nous allons pouvoir passer à la pratique. Comme les concepts soulevés par ces systèmes sont assez originaux, nous vous proposons de continuer sous forme de tutoriel. Il est en effet plus rassurant de « jouer » un peu avec les commandes avant de passer à une utilisation sur un projet réel.

Installation

L'installation de Subversion ne pose pas de problème, puisque vous devriez trouver tout ce qu'il faut dans le gestionnaire de paquets de votre distribution. Si ce n'est pas le cas, vous pouvez

```
$ mkdir /tmp/copie_locale  
$ svn checkout file:///tmp/referentiel /tmp/copie_locale
```

Checked out revision 0.

Si nous regardons dans le répertoire `/tmp/copie_locale`, nous pouvons voir que Subversion y a mis un sous-répertoire `svn` (auquel il ne faudra pas toucher !). Grâce aux données qu'il contient, nous n'aurons par exemple plus besoin de spécifier l'adresse du référentiel lors de l'utilisation du client Subversion (`svn`).

Importation d'un projet existant

Cette partie n'est pas utile dans le cadre du tutoriel, mais elle vous servira pour transférer vos projets existants sur Subversion.

Si vous avez déjà travaillé sur votre projet avant d'utiliser Subversion, il va falloir importer vos fichiers dans le référentiel avant d'effectuer le `checkout`. C'est tout naturellement le rôle de la commande `import`:

```
$ svn import adresse_des_fichiers file:///tmp/referentiel -m «initialisation»  
Adding      adresse_des_fichiers/fichier  
Committed revision 1.
```

Notez que l'on importe alors tout ce qui se trouve dans le répertoire `adresse_des_fichiers`. Il n'y a donc besoin de l'exécuter qu'une fois, même si votre projet contient déjà de nombreux fichiers et sous-répertoires.

La commande `svn import` prend trois arguments : le chemin de la source de l'importation, celui du référentiel et l'étiquette à attacher à l'importation (commutateur `-m`).

Cette dernière est très importante, puisque c'est elle qui nous permettra de nous y retrouver entre les différentes versions de nos fichiers. C'est en fait un commentaire que l'on ajoute au «

Retrouvez l'historique de vos documents grâce à Subversion
\$ svnadmin create adresse_de_mon_referentiel
\$ svn import mon_modele file:///adresse_de_mon_referentiel/mon_modele -m «Import de mon modèle pour article 1»
Adding mon_modele/article.tex
Adding mon_modele/figures
Adding mon_modele/Makefile

Committed revision 1.

\$ svn checkout file:///adresse_de_mon_referentiel/nouveau_projet ma_copie_de_tavail
A ma_copie_de_tavail/article.tex
A ma_copie_de_tavail/figures
A ma_copie_de_tavail_de_mon_nouveau_projet/Makefile
Checked out revision 1.

Un référentiel peut contenir plusieurs projets. Ceci dit, la solution la plus simple et la plus souple consiste à créer un référentiel pour chaque nouveau projet.

Utilisation d'un référentiel pour plusieurs projets

Création du référentiel contenant plusieurs projets :

```
svnadmin create mes_articles  
svn import mon_modele file:///mes_articles/article1 -m «Import de mon modèle pour article 1»  
svn import mon_modele file:///mes_articles/article2 -m «Import de mon modèle pour article 2»
```

Pour connaître le contenu d'un référentiel :

```
$ svn list file:///mes_articles/  
article1/  
article2/  
$ svn list file:///mes_articles/article1  
Makefile  
article.tex  
figures/
```

Pour récupérer une copie de travail de tous les articles :

```
svn checkout file:///mes_articles/ ./
```

Pour cela, il faut utiliser les commandes ~~svn cp~~, ~~svn rm~~, ~~svn mkdir~~, ~~svn mv~~ à la place des classiques ~~cp~~, ~~rm~~, ~~mkdir~~, ~~mv~~. Pour les mêmes raisons, il faut signaler à Subversion l'ajout d'un nouveau fichier de vos documents grâce à la commande ~~svn add~~ (<http://www.unixgardien.com/index.php/programmation/programmation-svn>). Retrouvez l'historique de vos projets grâce à la commande ~~svn add~~ (<http://www.unixgardien.com/index.php/programmation/programmation-svn>) (contrôle de version).

Note

Si on ne précise pas de fichier(s), les commandes ~~svn xxx~~ s'appliquent au répertoire courant.

Il est intéressant de noter que toutes ces actions ont un effet immédiat sur votre copie de travail, mais ne sont appliquées au référentiel qu'après l'étape de commit (y compris la suppression). Si vous changez d'avis (toujours avant le commit, vous pouvez toutes les annuler à l'aide de la commande ~~svn revert nom_du_fichier~~ qui ramène votre copie locale à l'état du précédent commit.

Voyons donc comment appliquer ces concepts à notre projet. Commençons par créer un fichier dans la copie de travail :

```
$ touch fichier.tex
```

Ajout d'un fichier au référentiel

À ce stade, un fichier (vide) existe dans la copie de travail, mais Subversion n'en a pas été informé. Nous pouvons vérifier l'état de ce fichier à l'aide de la commande ~~svn status~~:

```
$ svn status
```

```
?      fichier.tex
```

Cette commande renvoie la liste des fichiers du répertoire, précédé par leur état. Ici, le point d'interrogation signifie que Subversion ne sait que faire de ce fichier, qui n'a pas été placé sous son contrôle. Il faut donc ajouter ~~fichier.tex~~ explicitement :

```
$ svn add fichier.tex
```

```
$ svn status
```

```
A      fichier.tex
```

Copie de fichiers

Retrouvez l'historique de vos documents grâce à Subv

<http://www.unixgarden.com/index.php/programmatio...>

Nous pouvons créer une copie d'un fichier puis l'ajouter au référentiel, comme suit :

```
$ cp fichier.tex copie.tex
$ svn add copie.tex
A      copie.tex
```

Mais cela ne serait pas une bonne idée. En effet, ~~copie.tex~~ n'est pas un nouveau fichier, mais une copie. Cette méthode ajoute bien le fichier ~~copie.tex~~ au référentiel, mais ce fichier n'a pas d'historique. Avec la commande suivante, le fichier ~~copie.tex~~ sera aussi ajouté au référentiel, mais, en plus, Subversion lui attribuera l'historique du fichier ~~fichier.tex~~.

```
$ svn copy fichier.tex copie.tex
A      copie.tex
```

Suppression de fichiers

Pour supprimer un fichier de votre copie de travail et demander à Subversion de faire de même dans le référentiel lors du prochain commit, utilisez simplement la commande suivante :

```
$ svn delete fichier.tex
D      fichier.tex
```

Contrairement à son équivalent shell, la commande ~~svn delete~~ est réversible : en effet, même lorsqu'un fichier est « supprimé », son historique est conservé dans le référentiel. Il vous sera donc toujours possible d'y accéder, à condition de vous souvenir du numéro du commit où vous avez effectué la suppression. C'est là l'intérêt du journal de bord que Subversion vous demande de remplir à chaque commit et auquel vous pouvez accéder par ~~svn log~~.

Déplacement de fichiers

Cette opération peut être réalisée à l'aide des commandes ~~svn copy~~ et ~~svn delete~~ ou en une seule fois comme suit :

```
$ svn rename fichier.tex article.tex
```

Annuler des modifications

Nous allons maintenant commencer à manipuler vraiment les différentes versions de votre fichier.

Retour à une version antérieure

L'action la plus simple à effectuer, c'est de revenir à une ancienne version d'un fichier. Pour cela, il vous suffit de spécifier un numéro de révision à la commande `svn update` :

```
$ svn update fichier.tex -r 1
At revision 1.
```

Et c'est tout ! Si vous ne spécifiez pas de numéro de révision (avec le commutateur `-r`), Subversion mettra automatiquement à jour ~~fichier.tex~~ avec la dernière version disponible sur le référentiel.

Annulation d'un seul commit

Imaginons la situation suivante : vous devez rédiger un article dont le nombre de page est limité. À la fin de votre premier jet, l'article est trop long. Vous faites un commit (disons que vous passez à la version 2). Vous supprimez une partie et passez à la version 3. Vous présentez votre travail à votre ami que vous surnommez Maître Capello qui vous signale plein de fautes d'orthographe. Vous les corrigez, faites un nouveau commit (passage à la version 4) et vous présentez votre travail à votre chef.

Évidement, la première remarque de votre chef sera qu'il manque des choses (justement ce que vous avez supprimé) et qu'on peut gagner de la place ailleurs. Que faire ? Revenir à la version 2 vous permet de retrouver les parties supprimées mais vous fait perdre la correction de vos fautes d'orthographe.

La solution consiste à n'annuler que les modifications faites entre les versions 2 et 3. Voyons en

Les plus attentifs auront noté qu'il y a des fautes d'orthographe dans le texte. Corrigez-les, puis committez à nouveau :

```
$svn commit -m «corrections d'orthographe»
```

```
Sending          fichier.tex
Transmitting file data .
Committed revision 4.
```

Normalement, à ce stade, le fichier ~~fichier.tex~~ doit ressembler à cela :

```
début du fichier
conclusion
```

C'est à ce moment que votre chef vous demande de remettre la partie inintéressante, qui finalement lui plaît bien. Avant d'agir, regardons d'abord où nous en sommes. Pour cela, nous pouvons consulter le journal de bord, celui-là même que nous remplissons à chaque commit, à l'aide de la commande ~~svn log~~.

```
$ svn update
$ svn log
```

```
-----
r4 | cyril | 2005-11-23 ...
```

```
corrections d'orthographe
-----
```

```
r3 | cyril | 2005-11-23 ...
```

```
Suppression de la partie inutile
-----
```

```
r2 | cyril | 2005-11-23 ...
```

```
Suite de la rédaction de fichier.tex
-----
```

```
--- fichier.tex (revision 2)
+++ fichier.tex (revision 3)
@@ -1,3 +1,2 @@
 débu du fichié
-partie pas tré intéressante
conclusion
```

La sortie générée par ~~svn diff~~ est au format standard ~~diff~~. Le commutateur ~~-r~~ permet de définir deux numéros de révisions, ~~svn diff -r x:y~~ affiche les modifications nécessaires pour passer de la version ~~x~~ à la version ~~y~~.

Une ligne qui a été supprimée est précédée du signe ~~-~~, une ligne ajoutée est précédée d'un signe ~~+~~, une ligne modifiée est considérée comme une ligne supprimée puis comme une autre ligne ajoutée. La ligne qui précède et celle qui suit la modification sont également données de façon à retrouver le contexte.

Note

La commande ~~svn diff -r 3:2~~ permet d'afficher les opérations nécessaires pour annuler les modifications effectuées pour passer de la version 2 à la version 3.

Les personnes rompues à cet exercice savent que la suite logique d'un ~~diff~~, c'est la commande ~~patch~~. Mais Subversion fournit directement un outil « tout en un » qui applique directement le résultat du ~~diff~~ à un fichier (ou à tout un projet) : ~~svn merge~~. Son fonctionnement est identique à celui de ~~svn diff~~ :

```
$ svn merge -r 3:2 fichier.tex
```

```
U fichier.tex
```

Remarquez que l'on a spécifié le nom du fichier auquel on applique la commande. Comme dans le cas de ~~svn diff~~, lorsque l'on ne spécifie pas de nom de fichier, la commande est appliquée à tout le référentiel ! Attention donc !

Notez l'ordre des révisions (~~-r 3:2~~) : on cherche en effet à appliquer les modifications permettant de passer de la révision 3 à la révision 2 (c'est-à-dire supprimer la révision 3). Notez aussi que ~~svn merge~~ est supérieur à ~~patch~~ dans le sens où il permet par exemple d'annuler des

Retrouvage de vos documents grâce à Subversion. d'orthographe et de grammaire. [http://www.auxfardes.com/index.php/programmation...](http://www.auxfardes.com/index.php/programmation)

Retrouvage de vos documents grâce à Subversion. d'orthographe et de grammaire. <http://www.auxfardes.com/index.php/programmation...>

N'ayez pas peur d'alourdir votre référentiel en commitant trop souvent : Subversion ne stocke que les différences entre les fichiers, pas les fichiers eux-mêmes, ce qui fait que la place prise sur votre disque reste modeste.

Marquez vos documents

Avant de produire une version définitive de votre document, il vous faudra sans doute en distribuer des versions provisoires (pour relecture par des tiers, par exemple). Il vous arrivera sans doute que des personnes fassent des remarques sur la version qu'elles ont lue, qui n'est sans doute plus la version sur laquelle vous travaillez. Pour pouvoir vous y retrouver dans l'historique (et vérifier que ces remarques sont toujours valables), il est donc important que les références de version (numéro, date, auteur) soit écrites sur les documents que vous distribuez. Subversion permet de marquer automatiquement un fichier avec des renseignements comme la date, l'heure, le numéro de version, l'auteur du dernier commit, le nom du fichier ou l'adresse du référentiel. Pour cela, on utilise la propriété « mots-clefs » (keywords) [1]. Le fonctionnement est le suivant :

Tout d'abord, on signale à Subversion qu'il va devoir effectuer la substitution de mots-clefs sur un fichier (ici `fichier.tex`) :

[1] Chapitre 7 du livre [Gestion de projets avec Subversion](#)

```
$ svn propset svn:keywords «Id» fichier.tex
```

On modifie ensuite le contenu du fichier pour faire apparaître la chaîne de caractères `$Id$`. Par exemple, le contenu de `fichier.tex` peut être le suivant :

```
Contenu de mon document
version: $Id$
fin de mon document
```

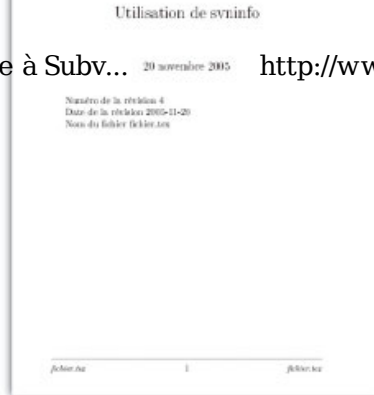


Fig. 2 : L'affichage des données de version à l'aide du paquet `svninfo`.

Heureusement, il existe un paquet spécifiquement destiné à l'affichage des renseignements de version : `svninfo`. Il vous faudra sans doute le récupérer sur le site du CTAN [2], car il est rarement inclus dans les distributions LaTeX classiques.

Son installation est aisée, puisqu'il vous suffit d'exécuter le Makefile livré avec `svninfo` (commande `make`), puis de copier les fichiers créés dans un répertoire accessible à LaTeX [3]. Pensez à rafraîchir la base des paquets avec la commande `texhash`.

L'utilisation est un peu tordue, mais relativement simple. Voici le contenu modifié de `fichier.tex` :

```
\documentclass[a5paper, 12pt]{article}
\usepackage[today]{svninfo}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[french]{babel}
\svnInfo $Id$
\title{Utilisation de svninfo}
\begin{document}
\maketitle
Numéro de la révision \svnInfoRevision
Date de la révision \svnInfoDate
Nom du fichier \svnInfoFile
\end{document}
```

Conclusion

Retrouvez l'histoire de vos documents grâce à Subv...

<http://www.unixgarden.com/index.php/programmatio...>

Nous espérons vous avoir convaincu que Subversion peut être utile à une personne travaillant seule sur des fichiers autres que du code et qu'il est simple à utiliser.

L'utilisation présentée ici est d'autant plus simple qu'elle évite la configuration du référentiel pour qu'il soit utilisé par plusieurs personnes à travers le réseau et qu'elle n'impose pas l'utilisation de branches, d'étiquettes...

Choses plus compliquées que Subversion sait très bien gérer mais qui n'ont pas d'utilité pour l'utilisation présentée.

Note

L'utilisation d'un gestionnaire de versions protège de la perte de données par effacement accidentel de fichiers à condition d'effectuer des commits réguliers. En revanche, vous n'êtes pas à l'abri de défaillances matérielles ou même de l'effacement du référentiel, il faut donc effectuer des sauvegardes de votre référentiel. La commande ~~svnadmin dump mon_referentiel~~ > ~~mon_fichier_de_sauvegarde~~ vous permettra d'effectuer une copie de sauvegarde. S'il arrivait malheur à votre référentiel, la commande ~~svnadmin load mon_nouveau_referentiel~~ < ~~mon_fichier_de_sauvegarde~~ vous tirera d'affaire.