Kévin Fardel

Tutoriel Tortoise SVN

03/07/09

Sommaire

Introduction

Ce petit tutoriel va vous apprendre les bases de l'utilisation de TortoiseSVN. De nombreux points ne seront pas abordés ou pas approfondis mais à la suite de ce tutoriel vous devriez pouvoir utiliser TortoiseSVN pour développer un projet en collaboration avec d'autres développeurs. Toutes les parties de ce tutoriel sont importantes si vous ne savez pas réaliser l'une d'entre elle n'espérez pas voir l'efficacité de TortoiseSVN dans un gros projet. Ce tutoriel n'est pas long et reste volontairement un résumé des commandes utiles pour l'utilisation qu'on en fera dans le cadre du projet Stics. Si vous souhaitez plus de détails libre à vous de vous rendre sur le site :
http://tortoisesvn.net/docs/nightly/TortoiseSVN_fr/

Cette documentation est nettement plus consistante et nettement plus longue. Par contre dans certaine partie la traduction anglais français n'est pas réalisée. Ne vous étonnez de tomber sur des paragraphes en anglais!!!

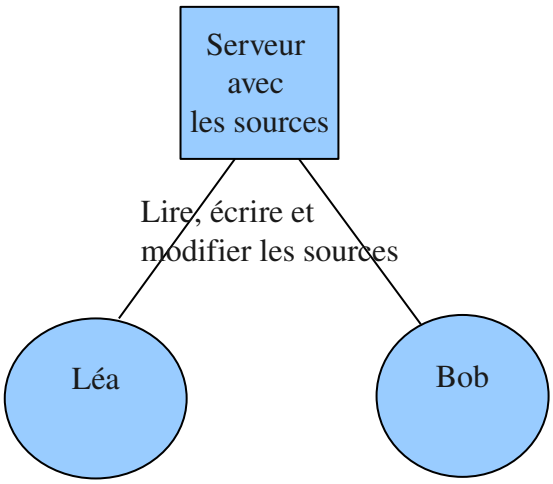
Pour toutes remarques sur ce petit tutoriel :
kevin.fardel@gmail.com

Présentation de Tortoise SVN

TortoiseSVN est un client open-source gratuit pour le système de contrôle de version *Subversion*. C'est-à-dire TortoiseSVN gère des fichiers et des répertoires à travers le temps. Les fichiers sont stockés dans un *référentiel* central. Le référentiel ressemble beaucoup à un serveur de fichiers ordinaire, sauf qu'il mémorise tout changement fait à vos fichiers et répertoires. Cela vous permet de récupérer les versions précédentes de vos fichiers et examiner l'historique de comment et quand vos données ont changé. C'est pourquoi beaucoup de personnes pensent que Subversion et les systèmes de contrôle de version en général sont une sorte de « machine à remonter le temps ».

Quelques systèmes de contrôle de version sont aussi des systèmes de gestion de configuration logicielle (GCL). Ces systèmes sont spécifiquement conçus pour gérer des arborescences de code source et ont beaucoup de fonctionnalités spécifiques au développement de logiciel - comme la compréhension de langages de programmation en natif, ou des outils d'approvisionnement pour construire le logiciel. Subversion, cependant, n'est pas un de ces systèmes, c'est un système général qui peut être utilisé pour gérer *n'importe quelle* collection de fichiers, y compris du code source.

Pour bien comprendre l'utilité de subversion (SVN) il faut prendre un exemple. Supposons que bob et léa sont deux développeurs d'un logiciel. Ces deux personnes vont travailler conjointement sur ce même projet. Mais supposons que bob travaille à Paris et Léa à Montréal. Il faut que les deux puissent utiliser les sources du projet en même temps, les modifier en même temps et surtout que les modifications de l'un soient apportées aux sources de l'autre et vice versa. Donc pour ce faire un serveur SVN est installé à Paris. Les sources sont stockés dans ce serveur.



TortoiseSVN va donc permettre à Bob et Léa de lire, écrire, et modifier les sources mais aussi de gérer différentes versions des sources.

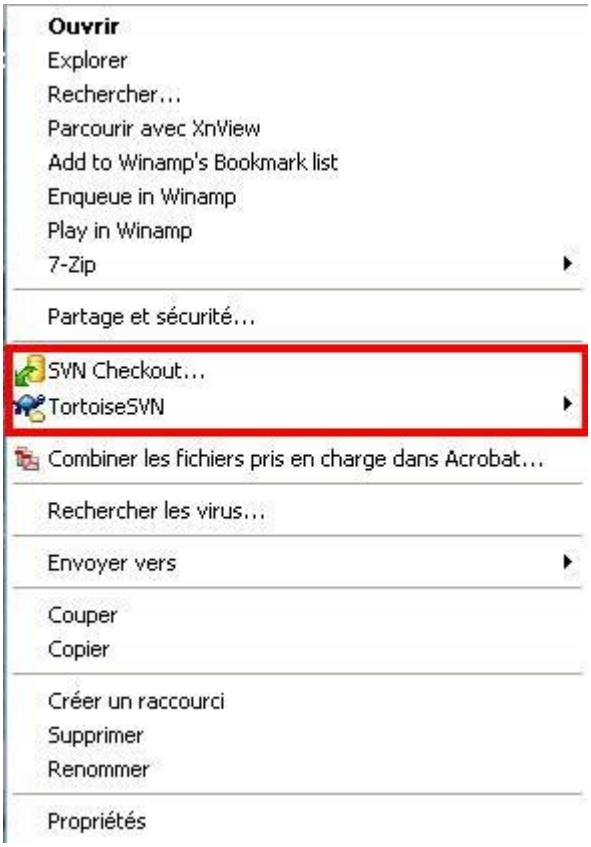
Installation

TortoiseSVN fonctionne sur Windows (2000, XP ou vista); pour des versions antérieurs de Windows il faut télécharger une version de Tortoise inférieure à la version 1.2.0.

L'installation de Tortoise est très simple, une fois le fichier d'installation lancé il faut suivre les étapes de l'assistant.

Attention : Il faut être administrateur de la machine pour pouvoir installer Tortoise. Si jamais vous n'êtes pas administrateur un message d'erreur vous l'indiquera durant l'installation.

A la fin de l'installation il vous faut redémarrer votre machine. Après ce redémarrage vous pouvez constater qu'en réalisant un clic droit sur un répertoire (dans votre explorateur Windows) un nouveau menu est apparu :



Si ce menu n'apparait pas alors l'installation c'est probablement mal déroulé. Dans ce cas aller sur le FAQ du site suivant :

http://tortoisesvn.tigris.org/faq.html.

Configuration de Tortoise SVN.

Tortoise est un CLIENT SVN donc il va vous permettre de vous connecter à un serveur SVN. Vous devez avoir en votre possession l'adresse de ce serveur.
Cette adresse peut se trouver sous plusieurs formes résumées dans le tableau suivant :

file://	Accès direct au référentiel sur disque local ou réseau.
http://	Accès via le protocole WebDAV à un serveur Apache avec Subversion.
https://	Même chose que `http://`, mais avec cryptage SSL.
svn://	Accès TCP/IP non authentifié via un protocole personnalisé à un serveur `svnserve`.
svn+ssh://	Accès TCP/IP authentifié, crypté via un protocole personnalisé à un serveur `svnserve`

Cette méthode d'accès est spécifié par l'adresse que vous fourni l'administrateur du serveur.

C'est la seule configuration absolument nécessaire pour utiliser Tortoise. Les autres configuration peuvent être réglées dans :

clique droit
TortoiseSVN -> Settings

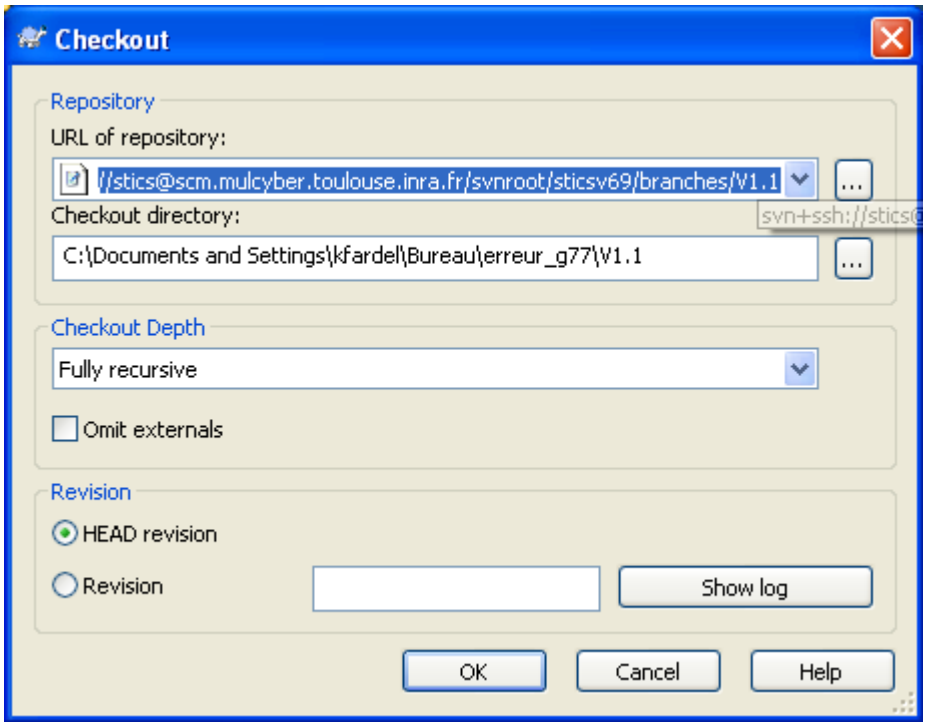
Utilisation de TortoiseSVN

1ère utilisation:

Lors de la première utilisation de Tortoise sur votre machine il faut tout d'abord rapatrier l'intégralité des sources du serveur sur votre machine locale (on suppose que des versions antérieures sont déjà en dépôt sur le serveur). Pour ce faire :

Clic droit sur un répertoire créé à cet effet
SVN Checkout...

La fenêtre suivante va s'ouvrir.



Dans le premier champs de cette fenêtre il faut renseigner l'adresse du répertoire où se trouvent les sources sur le serveur. Cette adresse doit être de l'une des forme exposées dans le tableau de la partie précédente suivit d'un nom de répertoire. Par exemple :

svn+ssh://nom_du_developpeur@scm.mulcyber.toulouse.inra.fr/svnroot/sticsv69/trunk/

Dans cet exemple nous voyons que la méthode d'accès est *svn+ssh://* ensuite il y a le login du développeur sur le serveur suivit de @, puis de l'adresse du serveur et de l'emplacement du répertoire où se trouve les sources que l'on souhaite rapatrier sur notre machine.

Dans le second champs il faut remplir l'adresse du répertoire où l'on veut rapatrier les sources sur notre machine. Par exemple :

C:\Documents and Settings\kfardel\Bureau\mulcyber

Ensuite le champs déroulant qui suit indique la profondeur du checkout que l'on veut réaliser. En gros on indique si on veut que le checkout se fasse dans le répertoire indiquer dans le champs 1 uniquement ou dans les sous répertoires du répertoire indiqué dans le champs, ou uniquement dans les sous répertoires immédiats.
Généralement on mettra la profondeur du checkout en récursif complet (fully recursive) car pour la première utilisation le mieux est probablement de récupérer la totalité des sources d'un projet (la profondeur c'est comment on va parcourir le répertoire, soit le répertoire racine et c'est tout soit le répertoire racine et l'ensemble de ces sous répertoire par exemple). Après libre à vous d'utiliser la profondeur que vous souhaitez. La dernière partie de la fenêtre indique la version que vous souhaitez récupérer du projet.
Si vous activer l'option *«HEAD revision»* vous rapatrierez la dernière version du projet.
Si vous sélectionnez *«Revision»* il vous faut indiquer quelle révision vous souhaitez rapatrier.

Prenons par exemple un projet que l'on vient de commencer. La première version du projet est sur le serveur (revision 0). Bob rapatrie avec l'option *«HEAD revision»*. Il récupère les sources qui viennent juste d'être déposées. Il réalise quelques modifications et les envois sur le serveur. Ce qui fait qu'il crée la version 1 (revision 1). Là Léa arrive et veut elle aussi travailler sur le projet mais veut partir du projet de base avant que Bob ne l'ai modifié. Alors elle sélectionne l'option Revision et écrit le numéro de la revision qu'elle souhaite dans le champs d'à côté (ici ce sera la revision 0). Un clique sur le bouton OK lancera le checkout avec les options choisies.

Durant le checkout vous verrez l'ensemble des opérations réalisées par Tortoise pendant le checkout (un bon moyen de voir si des problèmes ont eu lieu durant le checkout). A la fin du checkout quand vous avez fermé la fenêtre vous vous apercevait que le répertoire sur lequel vous venez de rapatrier les sources possède un  . Pas de panique cela signifie juste que tout c'est bien déroulé ! Ce petit symbole est rajouté par Tortoise par dessus vos icônes Windows pour vous indiquer l'état de votre version.
Au fur et à mesure de votre utilisation de Tortoise vous serez confronter à différents icône dans ce genre. La liste suivante vous indique ce que signifie les plus courants d'entre eux:

	Cet icône vous signifie que votre version local est inchangée et à jour et qu'il n'y a aucun problème visible.
	Cet icône vous signifie que vous avez modifié votre version sur votre machine et que vous n'avez pas réalisé de commit vers le serveur. Donc si d'autres utilisateurs rapatrient des sources sur leur machine ils n'auront pas vos modifications.
	Cet icône vous indique que le fichier (ou répertoire) a généré un conflit. Nous verrons plus bas comment apparaissent les conflits et comment les résoudre.

D'autres icônes peuvent apparaître je vous laisse le soin de les découvrir sur le site de TortoiseSVN : http://tortoisesvn.net/docs/nightly/TortoiseSVN_fr/tsvn-dug-wcstatus.html

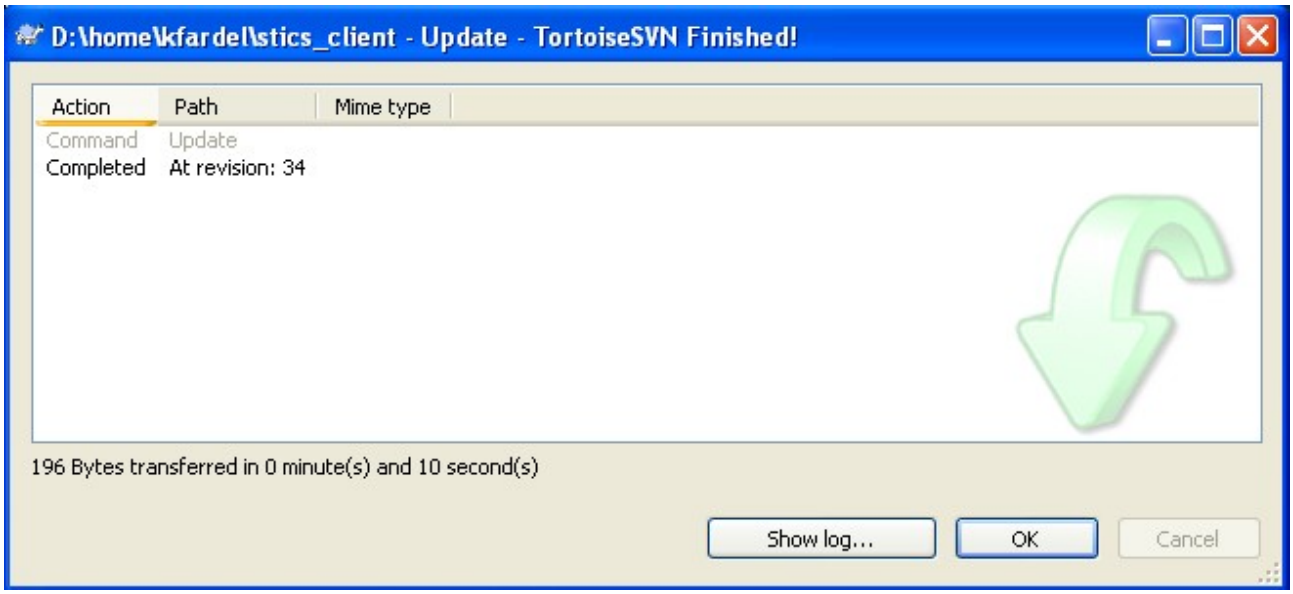
Donc à la fin de cette étape vous êtes en possession des sources du programme prêt à les utiliser.

Mise à jour

Pour mettre à jour vos sources il vous faut récupérer les sources, ayant été modifiés sur le serveur par une tiers personne. Ainsi sur votre machine vous aurez les dernières modifications apportées par les autres développeurs. La mise à jour se fait très simplement:

clique droit sur votre répertoire de travail
SVN update

Une fenêtre s'ouvre vous indiquant les fichiers qui ont été mis à jour sur votre machine:



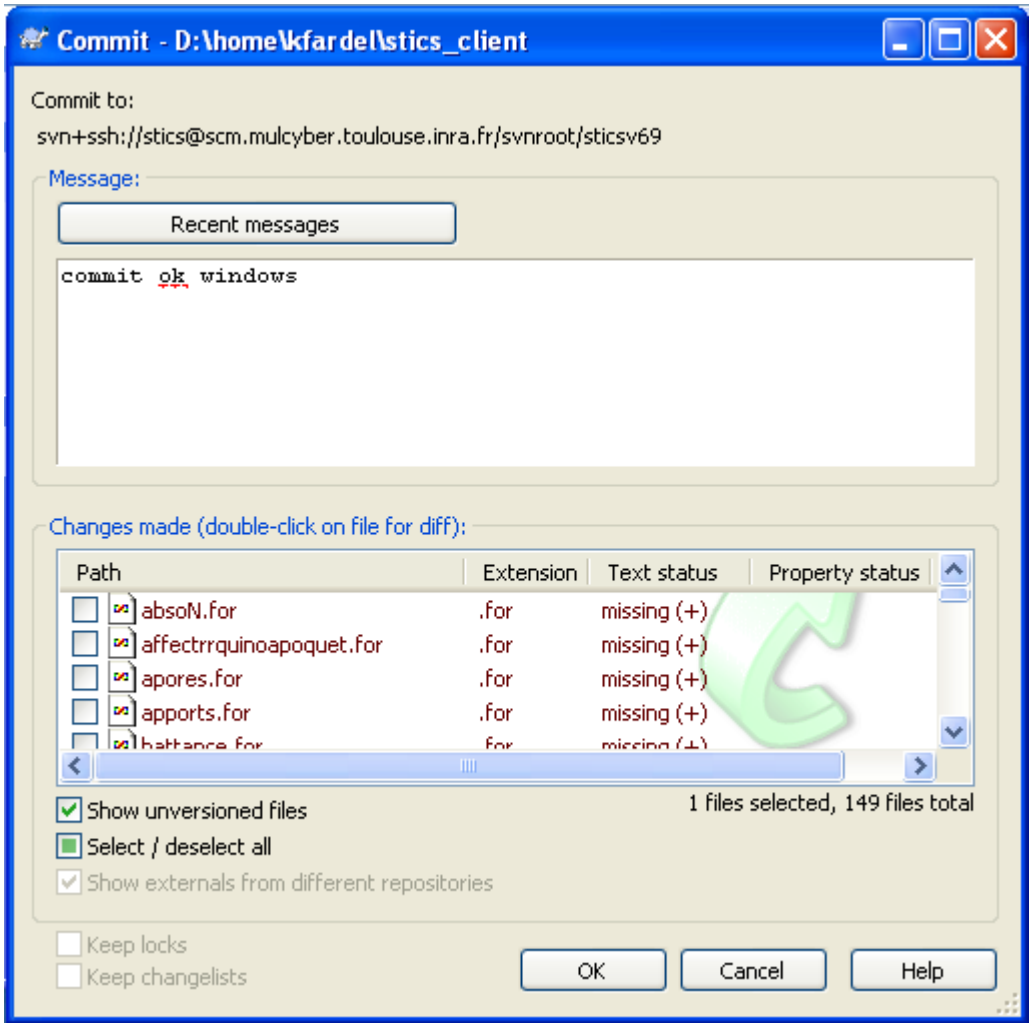
Si durant cette mise à jour des erreurs ont été affichées dans le fenêtre je vous laisse lire les deux parties suivantes pour vous expliquer comment les résoudre.

Commit

Une fois que vous aurez apporté les modifications que vous souhaitiez (ou que vous deviez) apporter au code source il faut que ces modifications soient vues de tout le monde. Cette opération s'appelle le commit. Elle se réalise par l'opération suivante :

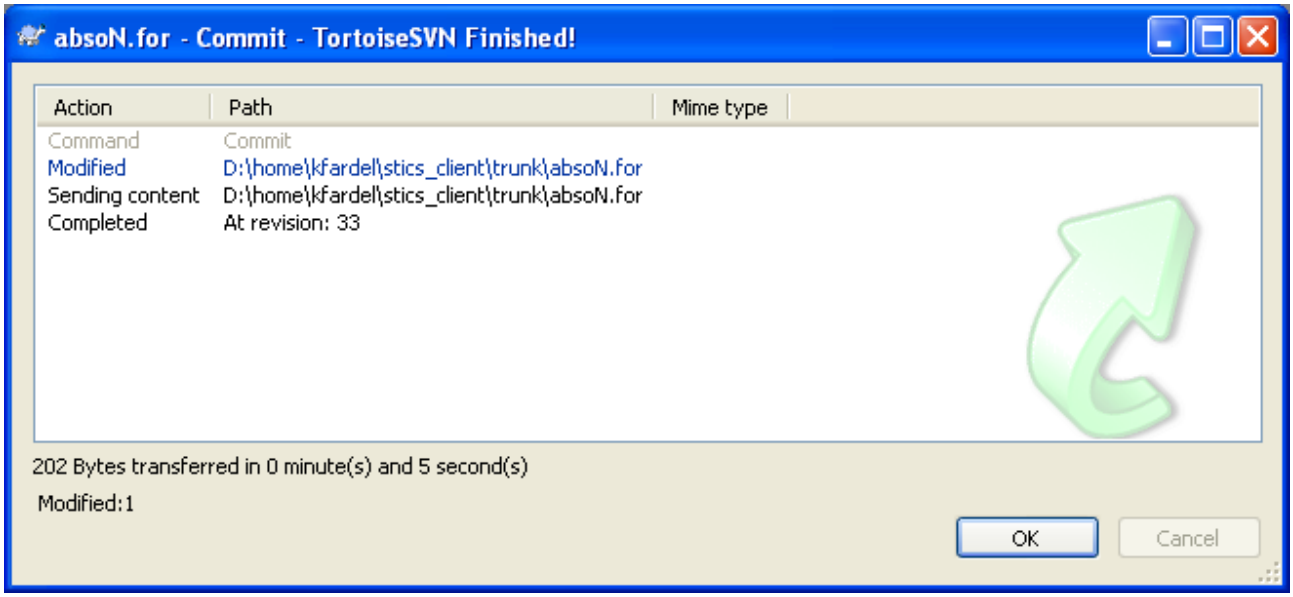
clique droit sur le répertoire que l'on souhaite mettre à jour sur le serveur.
Commit

Une première fenêtre va s'ouvrir :



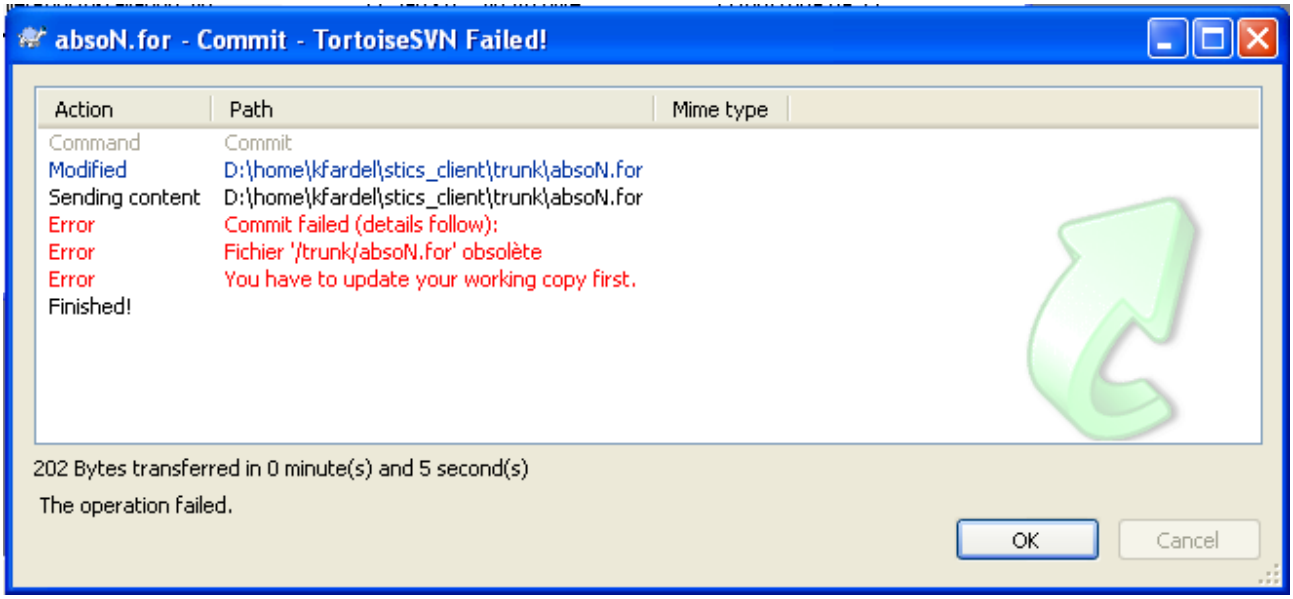
Dans le premier champs il vous faut indiquer un message (ce n'est pas obligatoire mais fortement recommandé) qui va permettre aux autres développeurs de savoir pourquoi vous avez fait ce commit. Le deuxième champs de la fenêtre vous indique tous les fichiers du répertoire qui vont être «commité». Pour chaque fichier vous avez une colonne avec le nom du fichier et une case à cocher. Une colonne indiquant l'extension du fichier et une colonne indiquant le statut du fichier. Normalement si le fichier a pour statut *«modified»* alors il est coché, ce qui signifie qu'il sera mis à jour sur le serveur. Si le fichier n'a pas été modifié entre votre dernière mise à jour sur votre machine et la mise à jour que vous réalisez maintenant, en toute logique, il n'est pas coché. C'est inutile de le mettre à jour sur le serveur.
Avant de lancer le commit penser tout de même à vérifier que le chemin d'accès au serveur (indiqué en haut de la fenêtre à la suite de *«commit to :»*) soit exact.

En cliquant sur OK une fenêtre va s'ouvrir vous indiquant les fichiers qui ont été mis à jour sur le serveur :



Dans cet exemple tout s'est bien déroulé. Donc les sources sur le serveur correspondent aux sources que vous avez sur votre machine.

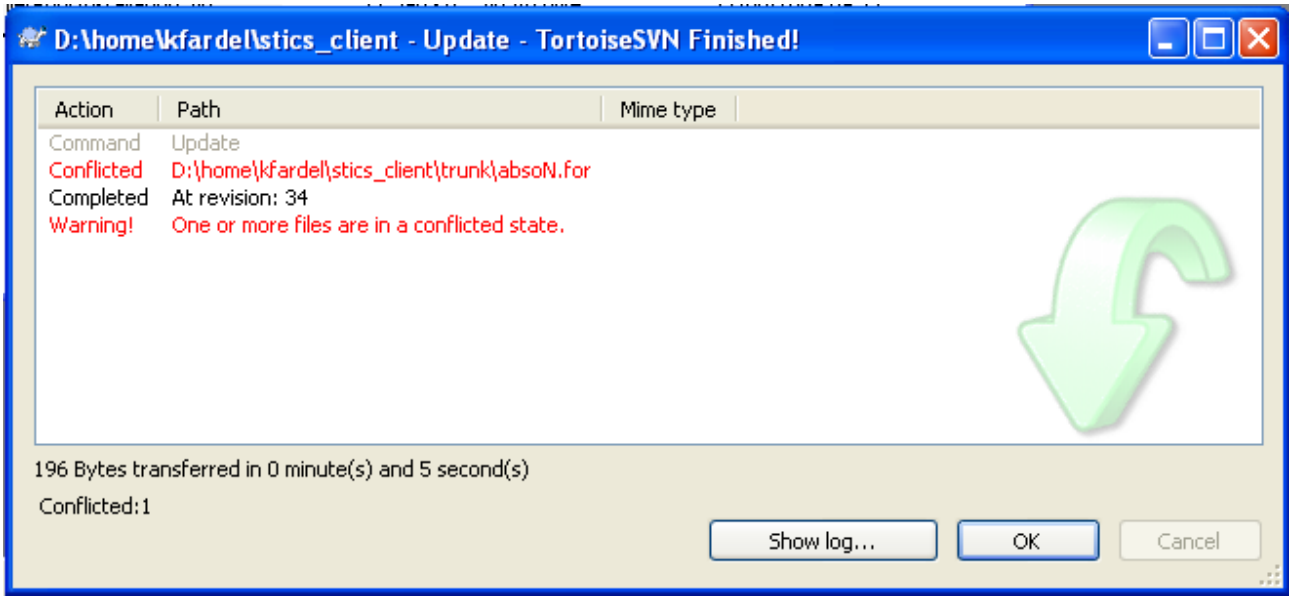
Mais durant ce commit il peut se créer des conflits. C'est lorsque les sources ont été modifiés par un autre utilisateur entre le moment où vous avez réalisé l'update (ou checkout) et le moment où vous réalisez ce commit. En cas de conflit un message dans votre fenêtre (ouverte lorsque vous avez lancé le commit) vous l'indiquera.



Pour résoudre ce conflit réalisez tout d'abord un update sur vos sources, puis faites un nouveau commit. L'update sur vos sources n'enlèvera pas les modifications que vous avez apporté à

ces dernières. Si pendant l'update aucune erreur n'est détectée alors c'est que le conflit est résolut.

Par contre si durant l'update vous obtenez un message de la sorte :



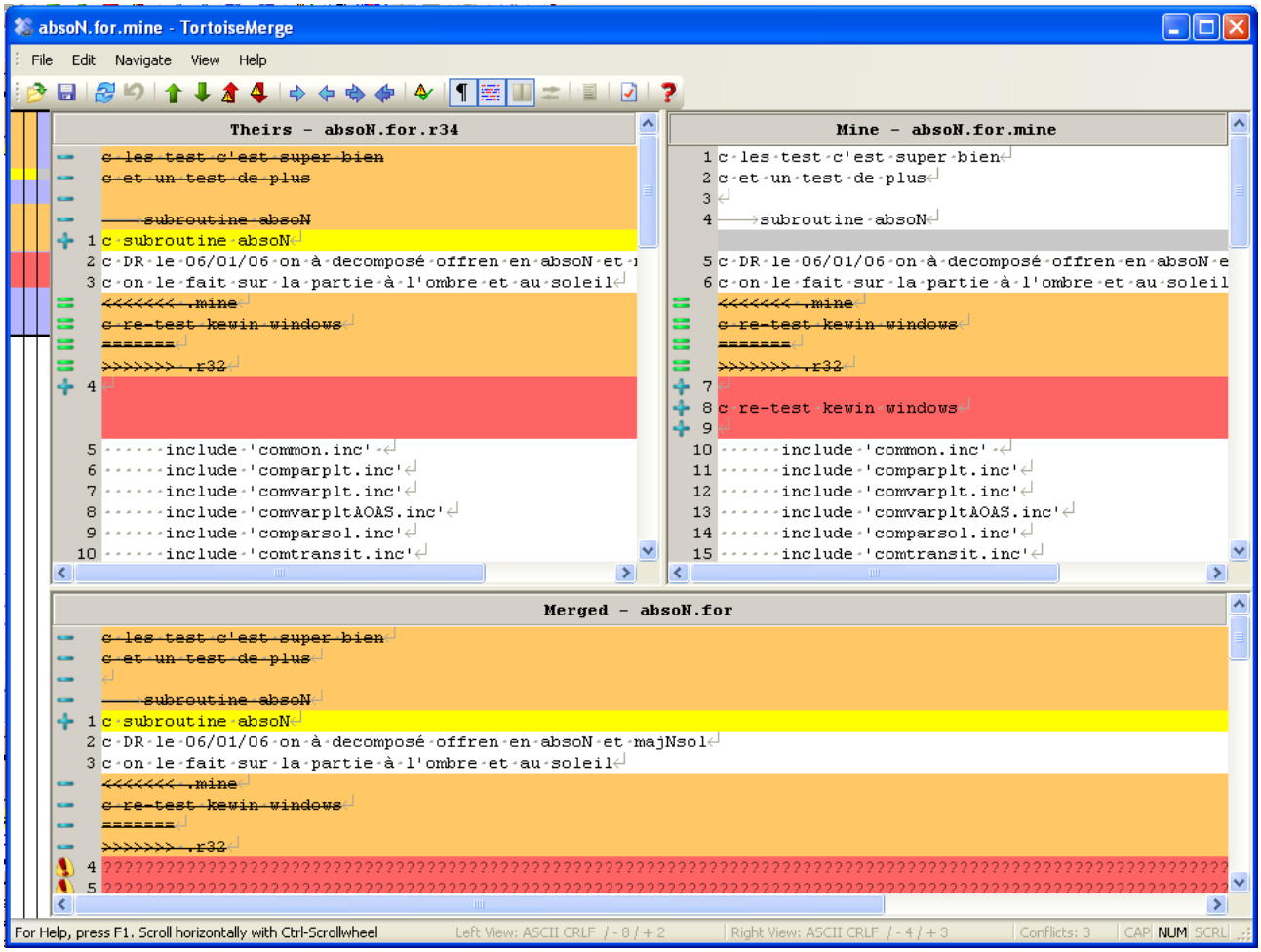
Alors cela signifie que Tortoise n'est pas en mesure de régler le conflit lui même. Ceci vient généralement du fait que la même ligne est modifiée dans le fichier qui pose problème sur le serveur et le fichier qui pose problème sur votre machine. Dans ce cas seul un des développeurs peut résoudre «manuellement» le problème.

Gestion d'un conflit

Tortoise vous offre un utilitaire pour comparer des fichiers. En cas de conflits il va vous falloir comparer le fichier sur votre machine, le fichier sur le serveur et le fichier que vous allez envoyer sur le serveur, où le conflit n'existera plus.
Si vous n'avez pas retenus le nom du fichier qui pose problème (c'est indiqué dans la fenêtre lors de l'update) pas de panique normalement l'icône de ce fichier (dans votre explorateur Windows) est mis en valeur par ⚠ .. Si encore une fois ce n'est pas le cas toujours pas de panique normalement un fichier qui pose problème est suivi par 3 fichiers du même nom (un avec l'extension « .mine » un avec l'extension « .r »+ un numéro et « .r »+un numéro).
Une fois que vous avez repéré le fichier qui pose problème. Réalisez l'opération suivante :

clique droit sur le fichier en question.
TortoiseSVN ->Edit conflicts

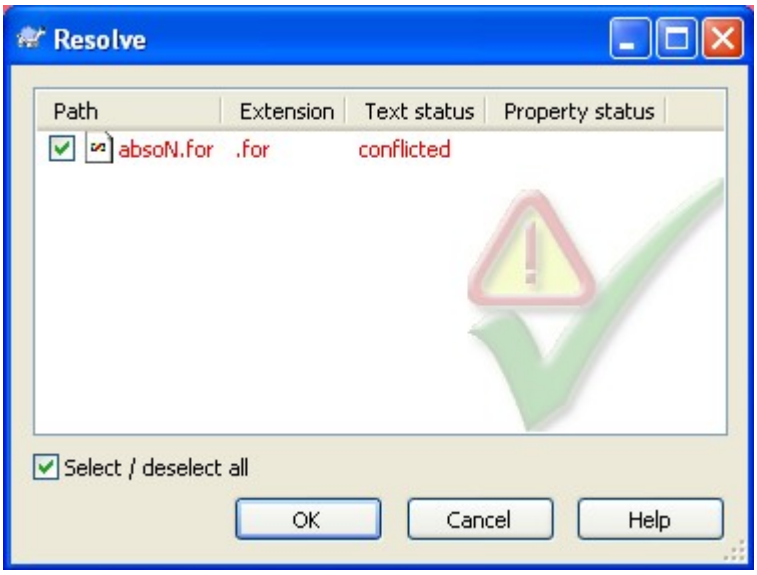
La fenêtre suivante s'ouvre :



Dans le coin en haut à gauche de la fenêtre vous voyez le fichier tel qu'il est sur le serveur, en haut à droite le fichier tel qu'il est sur votre machine, et en bas le fichier tel qu'il va être sur votre machine. Les lignes surlignées en orange sont celles qui on été modifiées par Tortoise automatiquement. Les lignes non surlignées sont celle qui n'ont pas été modifiées. Les lignes surlignées en rouge posent problème. Pour résoudre le problème sélectionnez les lignes en rouge, dans la partie basse de la fenêtre, et réalisez un clique droit dessus. Plusieurs options s'offrent alors à vous. Chaque option utilise les noms des deux bloques au dessus (le bloque « Theirs » et le bloque «Mine»). A vous de choisir l'option qui vous plait le plus. Une fois que le fichier qui pose problème ne possède plus de ligne surlignées en rouge les conflits sont résolus enregistrez, puis fermez cette fenêtre. Ensuite pour finir la résolution faites l'opération suivante :

clique droit sur le fichier qui posait problème
TortoiseSVN->Resolved

La fenêtre suivante s'ouvre alors :



Elle vous indique le fichier qui va être résolu. Une fois que vous avez cliqué sur OK une nouvelle fenêtre s'ouvre vous indiquant que le problème sur le fichier a bien été résolu.
Maintenant qu'il n'y a plus de conflit vous pouvez re-faire le commit. Et normalement tout devrait bien se dérouler.

Utilisation avancée

Gestion d'étiquettes

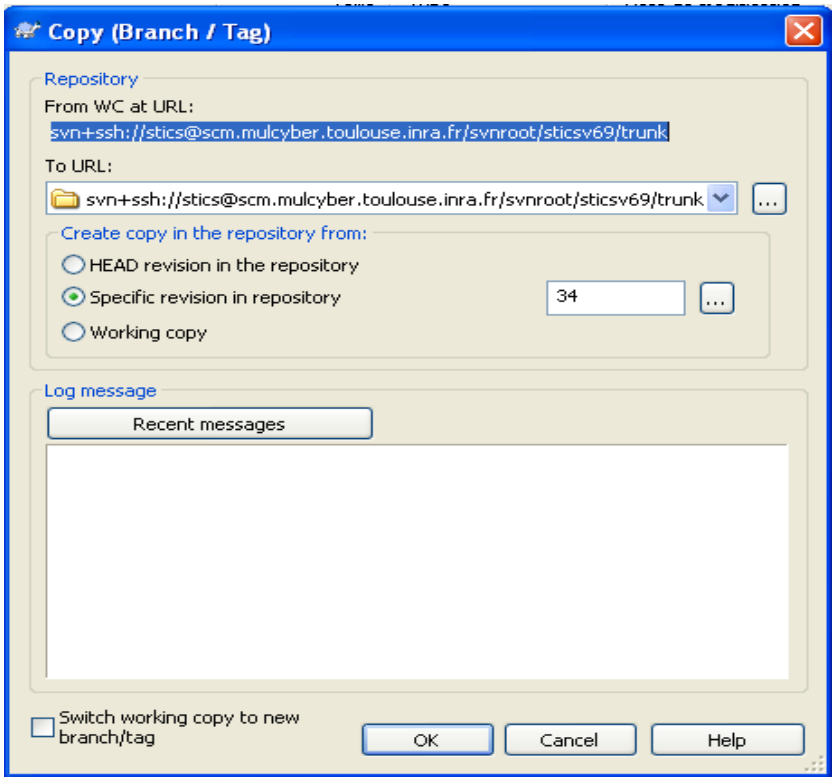
Tortoise offre la possibilité de gérer des étiquettes. Les étiquettes sont les morceaux de programme qui peuvent livrées à l'utilisateur pour qu'il puisse les utiliser. C'est par exemple une version béta d'un programme. C'est une version qui ne sera plus touchée. Pour que les étiquettes soit bien gérées il faut que sur le serveur les sources sur lesquelles les développeurs travaillent soient dans un répertoire *tronc* et qu'il y ait un autre répertoire nommé par exemple *etiquette*. La racine du serveur sera donc :

tronc/ <- Sources en développement
etiquette/ <-zone de stockage d'une version

Les développeurs travaillent sur le répertoire *tronc/* et quand une version de ces sources est livrable ils créent une étiquette, ce qui va copier toutes les sources dans le répertoire *etiquette/*.
Pour créer cette étiquette il vous faut réaliser l'opération suivante :

clique droit sur le répertoire tronc sur votre machine
TortoiseSVN -> branch/tag...

La fenêtre suivante s'ouvre :



Dans le champs à la suite de *«To URL»* il faut donner le répertoire *etiquette/* du serveur suivi

d'un répertoire indiquant le nom de l'étiquette. Dans l'exemple ci-dessus on entre :

sen+ssh://stics@scm.mulcyber.toulouse.inra.fr/svnroot/sticsv69/etiquette/nom_de_version

Ensuite on nous demande ce qu'il faut prendre pour étiquette:

- Soit la dernière version du tronc dans ce cas il faut cocher le bouton *«HEAD revision in the repository»*.
- Soit une révision dans ce cas il faut cocher *«Specific revision in repository»* et entrer dans le champs d'à côté le numéro de la révision
- Soit notre répertoire de travail. Dans ce cas il faut sélectionner *«Working copy»*.

A la suite il vous est demandé de remplir un message qui décrira la version. Ce champs n'est pas obligatoire mais fortement recommandé pour que les utilisateurs du serveur sache ce que représente cette version.
Ensuite appuyer sur OK et une fenêtre s'ouvrira vous indiquant que tout s'est bien déroulé.

Gestion des branches

Pour Tortoise une branche est équivalent à une étiquette. Mais pour réaliser un projet propre il faut différencier les deux. Une étiquette est une version livrable du projet alors qu'une branche est créée par un développeur pour qu'il code de son côté et que ses modifications ne soient pas intégrées au tronc. Par exemple si le projet n'est pas fini mais qu'il fonctionne. On peut créer une étiquette d'une version béta. Pour que les utilisateurs testent le programme. Pendant ce temps des développeurs utilisent les sources dans le tronc pour corriger les bugs. Mais un autre groupe de développeur souhaitent en parallèle ajouter une fonctionnalité au programme. Alors ils créent une branche et ajoutent leur fonctionnalité en utilisant cette branche. Une fois la fonctionnalité terminé il faut que les développeurs puissent ramener leur branche sur le tronc où les autres développeurs ont corrigé des bugs. Tortoise permet de gérer tout ce cheminement.
Pour créer une branche il faut faire comme pour créer une étiquette :

clique droit sur le tronc sur votre machine locale.
TortoiseSVN -> Branch/tag ...

Dans la fenêtre qui s'ouvre cette fois au lieu de sélectionner le répertoire etiquette/ on entre le chemin pour le répertoire branches/ suivit du nom de la branche. Par exemple :

svn+ssh://stics@scm.mulcyber.toulouse.inra.fr/svnroot/sticsv69/branche/nom_de_la_branche

Cela implique que le répertoire racine sur le serveur soit de la forme :

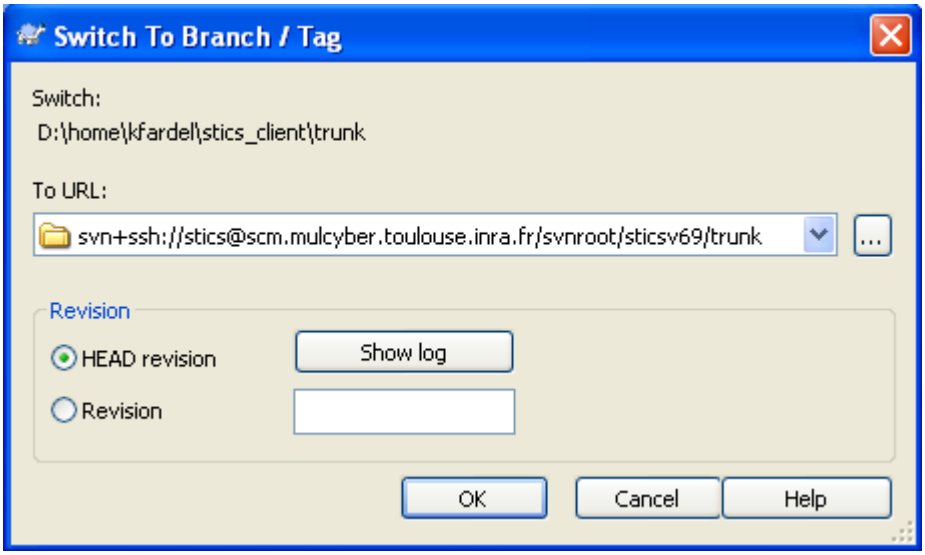
tronc/
etiquette/
branche/

Maintenant que la branche est créée il va falloir déplacer notre espace de travail (l'espace de travail est le répertoire où nous réalisons les commit et à partir duquel nous mettons à jour nos sources). En effet nous n'allons plus travailler sur le tronc mais sur notre branche.

Pour réaliser ce déplacement (ou *switch*) :

clique droit sur notre répertoire de travail (sur la machine local bien sur)
TortoiseSVN -> Switch...

Vous obtenez alors la fenêtre suivante :



Dans le premier champs il faut changer le répertoire et mettre le répertoire de la branche sur laquelle on va travailler :

svn+ssh://stics@scm.mulcyber.toulouse.inra.fr/svnroot/sticsv69/branche/nom_de_la_branche

Ensuite on peut choisir entre se placer à la révision la plus récente de la branche (sélectionner *«HEAD revision»*) ou se placer à une révision antérieure en entrant le numéro de la révision.
En cliquant sur OK une nouvelle fenêtre s'ouvre vous indiquant que tout c'est bien déroulé.
A partir de maintenant tous les update et tous les commit que vous allez réaliser se feront sur cette branche et ne toucherons plus au tronc.

Par la suite si vous voulez recoller cette branche au tronc faites :

clique droit sur votre répertoire de travail
SVN commit

Le serveur aura ainsi la dernière version des sources présentent sur votre machine.

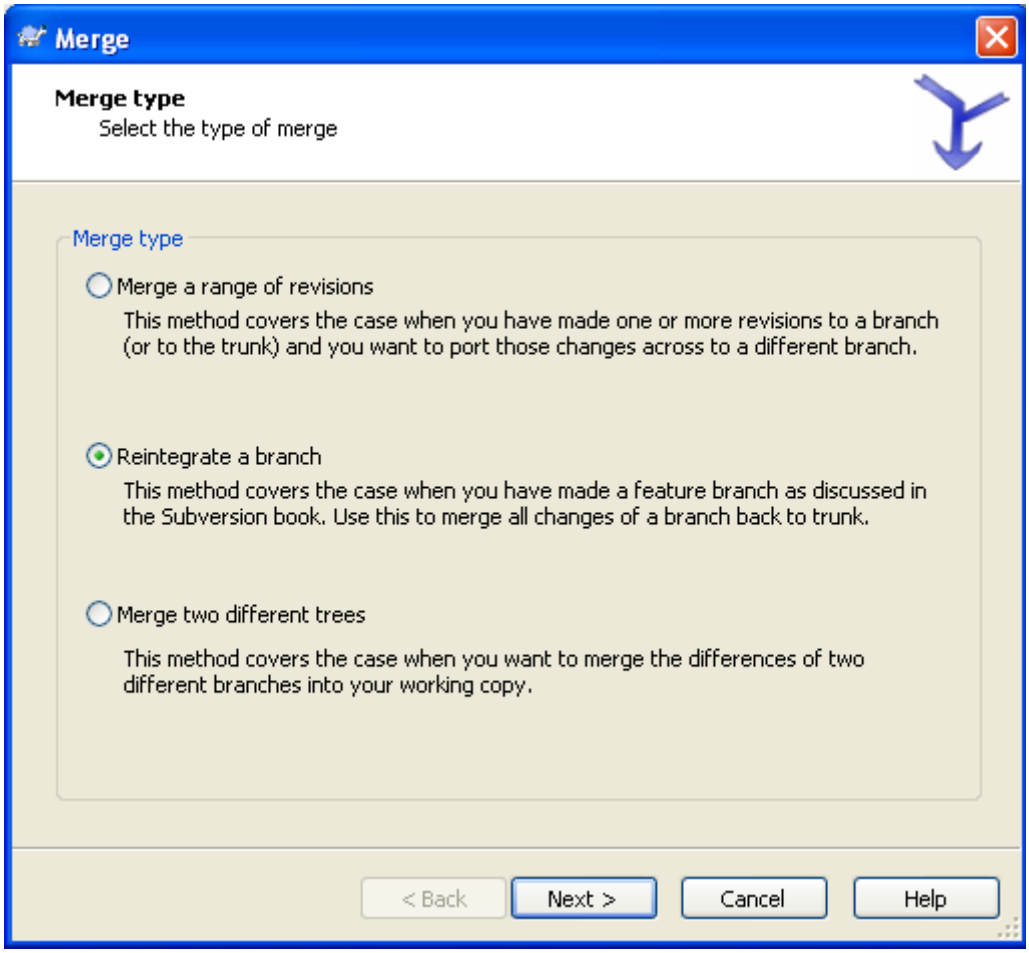
Il vous faut réaliser un nouveau *switch* vers le tronc. Vous créer un répertoire tronc sur votre machine locale puis:

clique droit sur ce dernier répertoire.
SVN update

Maintenant vous avez les sources de votre branche mise à jour sur le serveur et un répertoire tronc possédant les sources du tronc du serveur.

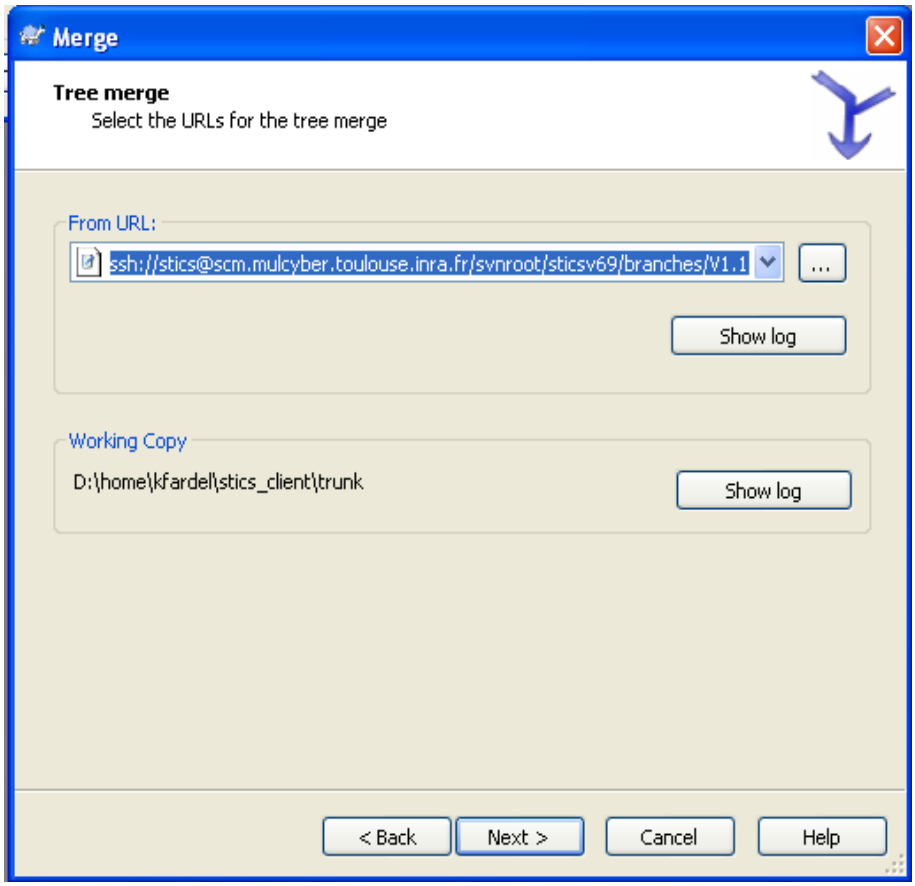
clique droit sur le répertoire tronc
TortoiseSVN -> Merge ...

La fenêtre suivante s'ouvre :



Dans cette fenêtre sélectionnez «Reintegrate a branch» (pour réintégrer une branche au tronc, pour les autres options allez sur le site :
http://tortoisesvn.net/docs/nightly/TortoiseSVN_fr/tsvn-dug-merge.html)

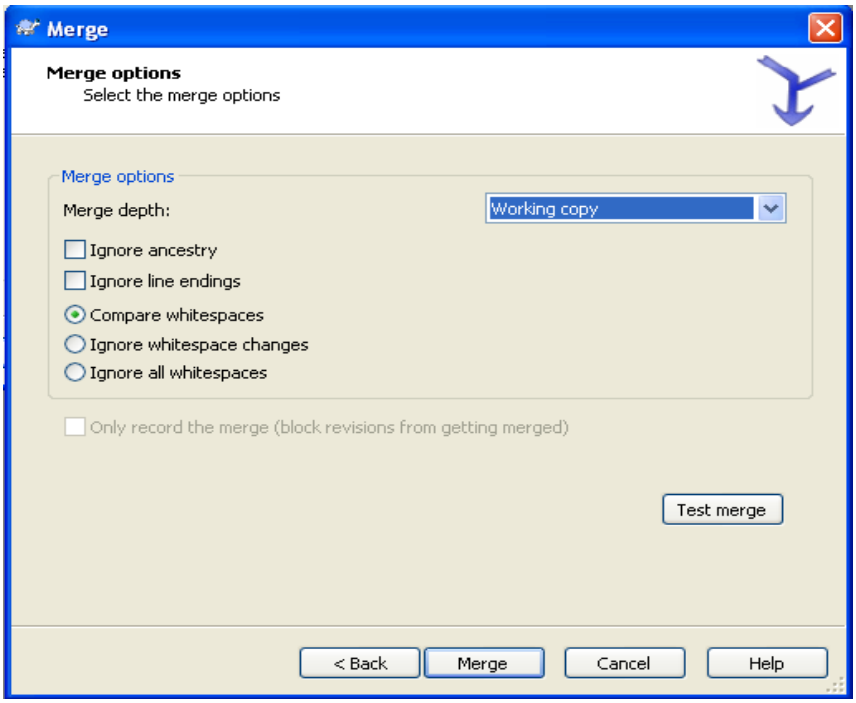
Appuyez sur Next. La fenêtre suivante apparaît :



Dans le premier champs indiquez la branche que vous souhaitez fusionner au tronc. Par exemple :

svn+ssh://stics@scm.mulcyber.toulouse.inra.fr/svnroot/sticsv69/branche/nom_de_la_branche

Appuyer sur Next. La fenêtre suivante apparaît :



Sur cette fenêtre vous n'avez pas vraiment besoin de modifier les options, seulement le champs *«Merge depth»* où vous pouvez choisir *Fully recursive.*

Une fenêtre s'ouvre vous indiquant l'avancement de la fusion. Si il y a un conflit (voir partie Gestion d'un conflit) la fenêtre suivante va s'ouvrir :

Soit vous choisissez de garder ce qui vient du tronc : *«Use local»*
Soit vous choisissez de garder ce qui vient de la branche : *«Use repository»*

Soit vous éditez les conflits : *«Edit conflict»* (et reportez vous à la partie Gestion des conflits pour savoir quoi faire devant la nouvelle fenêtre qui s'ouvre) et pensez ensuite à cliquer sur *«Resolved»*.
Soit vous choisissez de résoudre plus tard : *«Resolve later»*

Ensuite le tronc doit posséder les modifications qui avaient été apportées sur la branche en plus des modifications qui avaient été apportées à la branche.

Recommandations

- Garder un seul tronc qui sera la version en cours (les version sont en fait les révisions)
- Il faut faire une étiquette quand une version est figée
- Faire des branches quand vous êtes plusieurs développeurs
- Il faut être très rigoureux sur les commentaires et les messages lors des commit ou des checkout pour ne pas se perdre dans les versions