

Utilisation de Apache Subversion au CTIG*

Olivier Filangi^{1,2}
olivier.filangi@rennes.inra.fr

CATI SICPA

Systemes d'Informations et Calcul pour le Phénotypage Animal

² **UMR PEGASE**

Physiologie, Environnement et Génétique pour l'Animal et les Systemes
d'Élevage

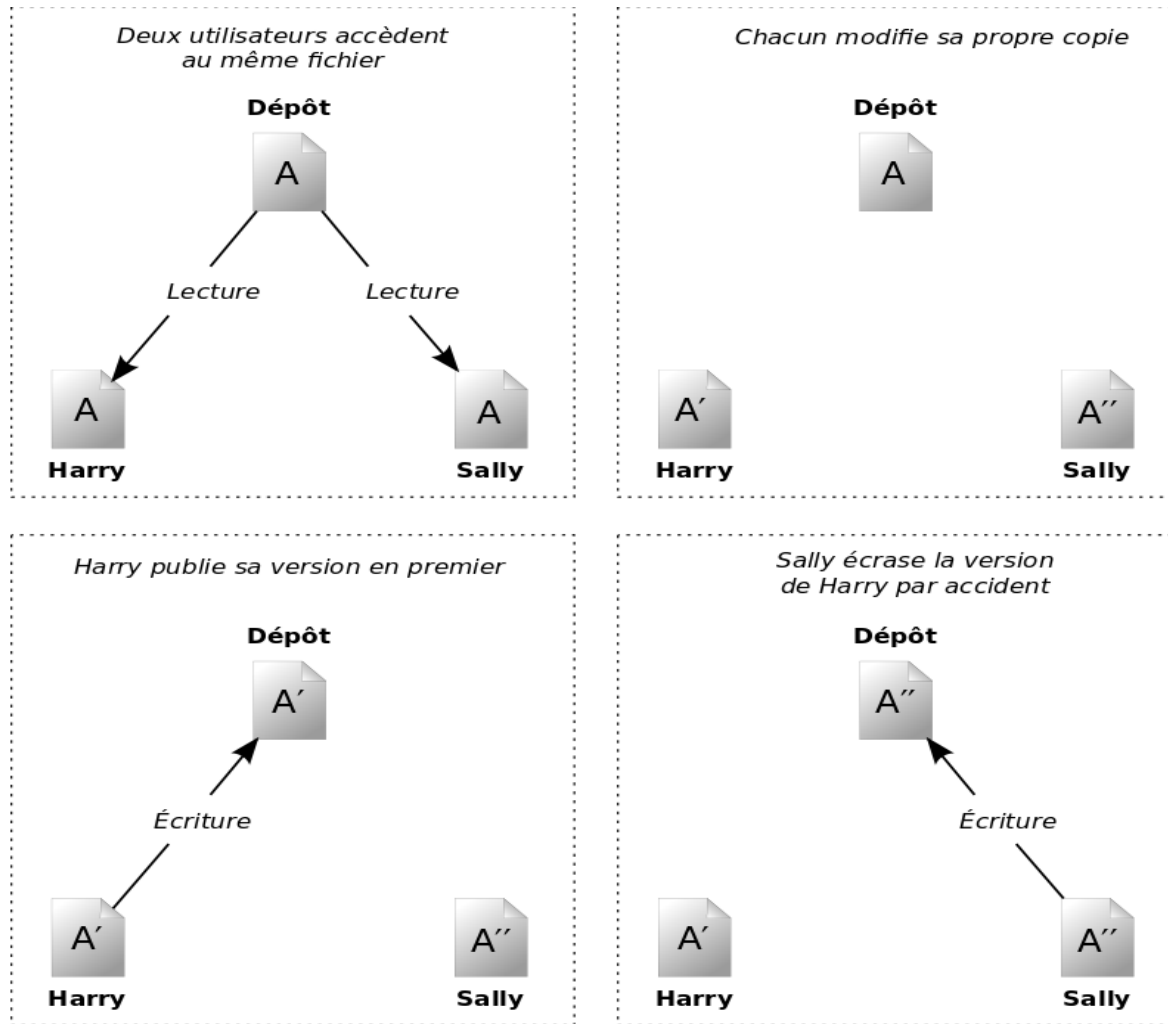
révision 23/11/2012



*Centre de Traitement de l'Information Génétique

Modèle de Gestion de Versions(1)

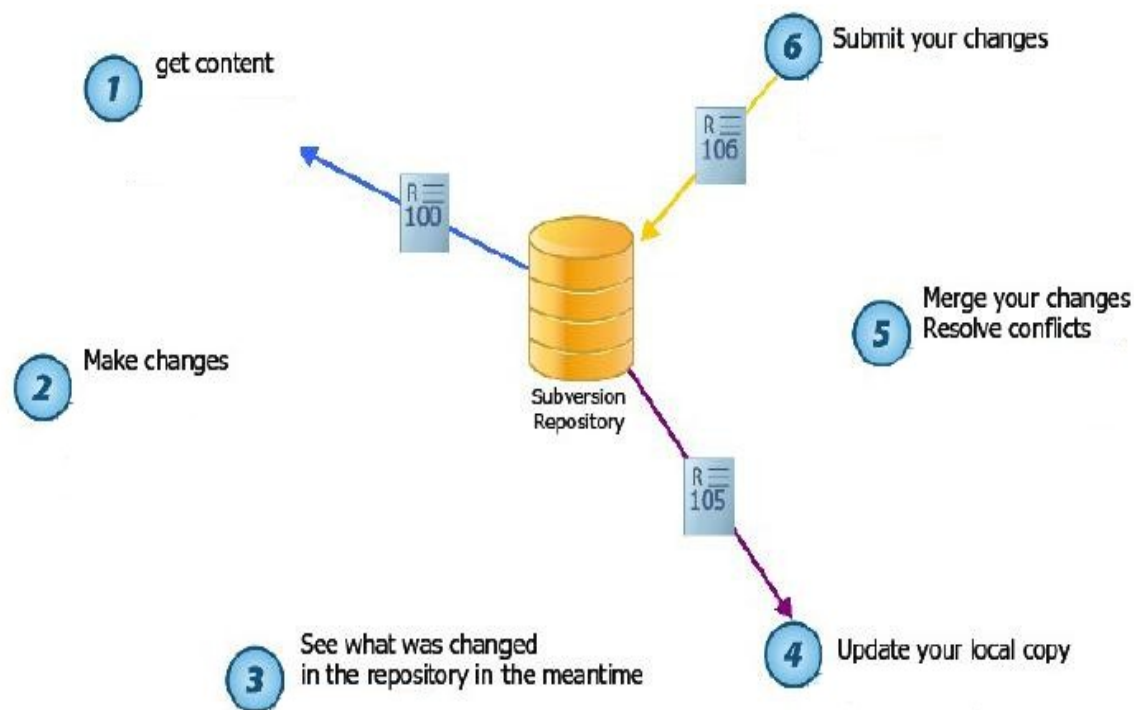
Problématique du partage de fichiers



Modèle de Gestion de Versions(2)

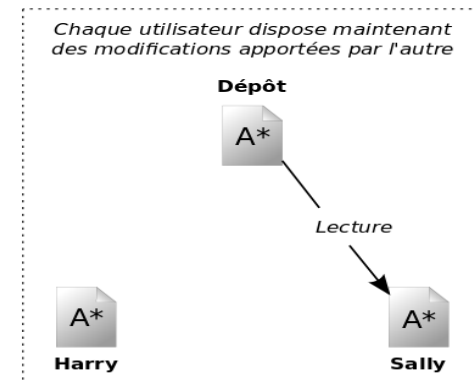
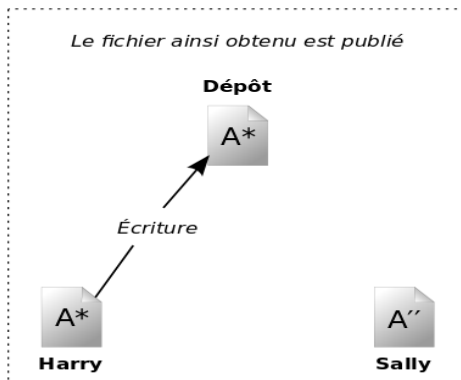
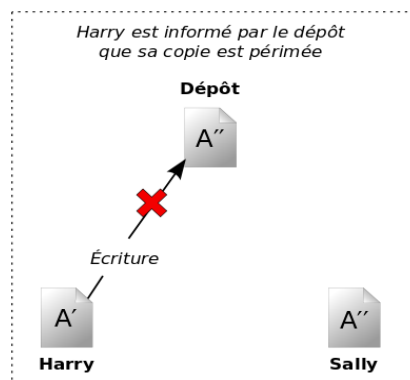
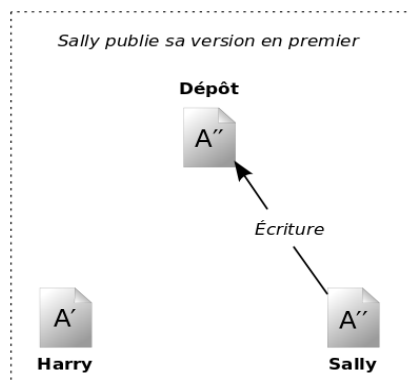
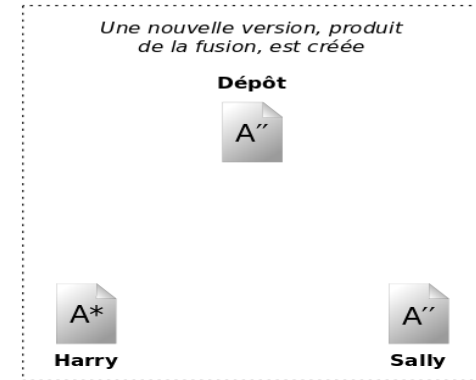
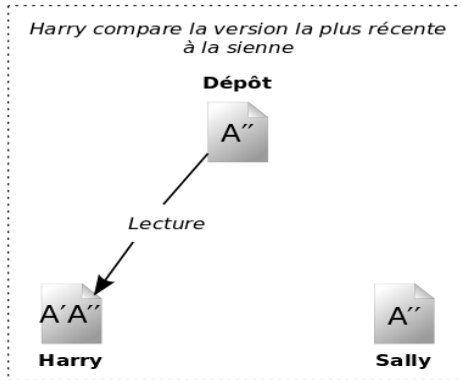
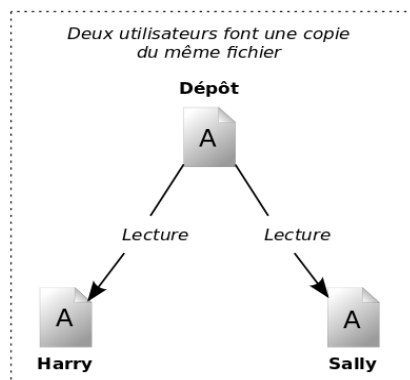
Modèle copier-modifier-fusionner implémenté par SVN

- Utilisation d'une copie de travail personnelle (CT)
- Modification en locale et indépendante pour chaque utilisateur
- Les copies privées sont fusionnées au sein d'une version finale

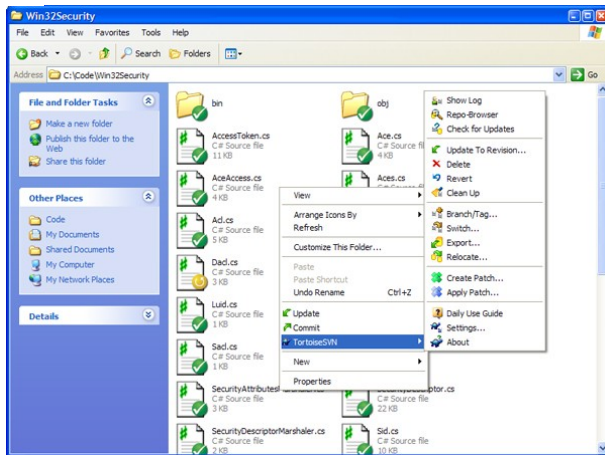


Modèle de Gestion de Versions(3)

Cas d'utilisation du modèle copier-modifier-fusionner



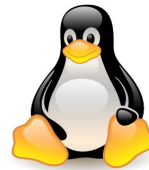
Clients graphiques SVN



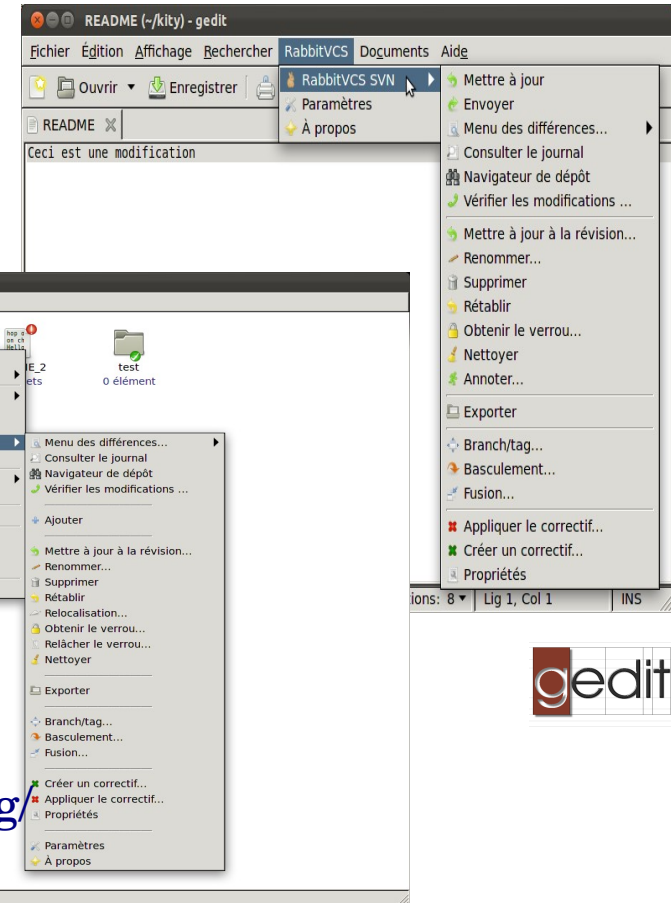
<http://tortoisesvn.net/>



RabbitVCS



<http://rabbitvcs.org/>



psvn.el (http://xsteve.nit.at/prg/vc_svn/) (mode pour Emacs)



Clients SVN et IDE

The screenshot shows the Eclipse IDE interface for a Fortran project. The Project Explorer on the left displays a directory structure with files like 'const_gpu_impl.h', 'm_qtlmap_analyse_mod', and 'm_qtlmap_map.f95'. The main editor window shows the source code for 'm_qtlmap_analyse.f95', specifically the 'check_qtl_compatibility' subroutine. A context menu is open over the code, with the 'Team' option highlighted. The right sidebar shows a list of variables and a table of warnings.

Resource	Path	Location	T
qtlmap_interface.c	/QTLMap-RedMine/	line 106	C
qtlmap_interface.c	/QTLMap-RedMine/	line 26	C
QTLMap-RedMine			C
QTLMap-RedMine			C
[V2.F95.o.provides] Erreur 2			C
QTLMap-RedMine			C
QTLMap-RedMine			C
Warning: Deleted feature: Assigned GOTO statement at (1)	dcdflib.f95	/QTLMap-RedMine/ line 7606	C
Warning: Deleted feature: ASSIGN statement at (1)	dcdflib.f95	/QTLMap-RedMine/ line 7626	C
Warning: Deleted feature: ASSIGN statement at (1)	dcdflib.f95	/QTLMap-RedMine/ line 7637	C
Warning: Deleted feature: ASSIGN statement at (1)	dcdflib.f95	/QTLMap-RedMine/ line 7687	C
Warning: Deleted feature: ASSIGN statement at (1)	dcdflib.f95	/QTLMap-RedMine/ line 7729	C
Warning: Deleted feature: ASSIGN statement at (1)	dcdflib.f95	/QTLMap-RedMine/ line 7785	C
Warning: Deleted feature: ASSIGN statement at (1)	dcdflib.f95	/QTLMap-RedMine/ line 7842	C

- ✓ **Subclipse (<http://subclipse.tigris.org/>) (plugin pour Eclipse)**
- ✓ **AnkhSVN (<http://ankhsvn.tigris.org/>) (plugin pour Visual Studio .Net)**



Client en ligne de commande

livré avec les sources/exécutables de SVN

```
svn <commande> <sous-commande> | <url> | <file> | <options>
```

La seule commande à savoir par cœur....

svn help : liste toute les sous-commandes de SVN

svn help <sous-commande> : syntaxe, les options et le comportement de la sous-commande



SVN et la forge Redmine DGA

Initialisation du dépôt

Cf FAQ* (Création d'un projet) : Remarque importante : **Si vous voulez disposez d'un dépôt (svn), il suffit de le choisir dans le menu déroulant nommé SCM. Ne rien renseigner dans l'onglet "dépôt".**

Nouveau projet

Nom *

Sous-projet de

Description

Identifiant *
Longueur comprise entre 1 et 100 caractères. Seuls les lettres minuscules (a-z), chiffres, tirets et underscore sont autorisés.
Un fois sauvegardé, l'identifiant ne pourra plus être modifié.

Site web

Public

SCM

Modules

<input checked="" type="checkbox"/> Suivi des demandes	<input checked="" type="checkbox"/> Suivi du temps passé	<input checked="" type="checkbox"/> Publication d'annonces	<input checked="" type="checkbox"/> Publicatio
<input checked="" type="checkbox"/> Dépôt de sources	<input checked="" type="checkbox"/> Forums de discussion	<input checked="" type="checkbox"/> Calendrier	<input checked="" type="checkbox"/> Gantt

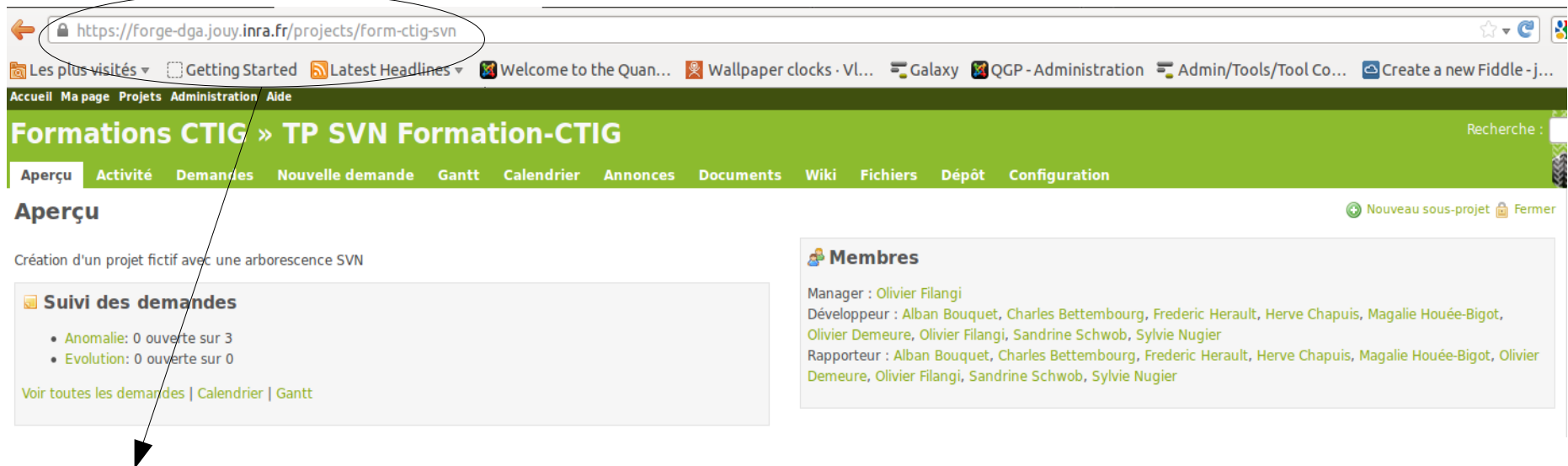
Trackers

<input checked="" type="checkbox"/> Anomalie	<input checked="" type="checkbox"/> Evolution	<input checked="" type="checkbox"/> Assistance	<input checked="" type="checkbox"/> Test
--	---	--	--

* <https://forge-dga.jouy.inra.fr/projects/docforge/wiki/FAQ>

Obtenir l'URL SVN

- Protocole HTTPS (SSL) - les mots de passe sont cryptés



https://forge-dga.jouy.inra.fr/projects/form-ctig-svn

Formations CTIG » TP SVN Formation-CTIG

Aperçu

Création d'un projet fictif avec une arborescence SVN

Suivi des demandes

- Anomalie: 0 ouverte sur 3
- Evolution: 0 ouverte sur 0

Voir toutes les demandes | Calendrier | Gantt

Membres

Manager : Olivier Filangi
Développeur : Alban Bouquet, Charles Bettembourg, Frederic Herault, Herve Chapuis, Magalie Houée-Bigot, Olivier Demeure, Olivier Filangi, Sandrine Schwob, Sylvie Nugier
Rapporteur : Alban Bouquet, Charles Bettembourg, Frederic Herault, Herve Chapuis, Magalie Houée-Bigot, Olivier Demeure, Olivier Filangi, Sandrine Schwob, Sylvie Nugier

https://forge-dga.jouy.inra.fr/projects/monprojet



https://forge-dga.jouy.inra.fr/svn/monprojet



Organisation d'une arborescence de développement logiciel

trunk : le tronc, dernière version en développement de l'application

branches : versions dérivées de la version trunk et/ou des versions tags

tags : versions figées de l'application avec un accès seulement en écriture

exemple d'architecture d'un depot <https://forge-dga.jouy.inra.fr/svn/monprojet> :

- trunk
 - src
 - main.f90
 - README
 - Makefile
- branches
 - 1.0.1
 - src
 - main.f90
 - README
 - Makefile
- tags
 - 1.0.0
 - src
 - main.f90
 - README
 - Makefile

Caractéristiques d'une branche développement :

- Maturité d'une fonctionnalité
 - **Ex** : *Une nouvelle analyse en cours d'évaluation à ne pas intégrer aux prochaines releases*
- Fonctionnalités distinctes du développement de la branche principale
 - **Ex** : *GUI*

Caractéristiques d'un TAG:

- Snapshot d'une révision (release) et assignation d'un numéro de version

créer un tag : `svn copy https://forge-dga.jouy.inra.fr/svn/monprojet/trunk https://forge-dga.jouy.inra.fr/svn/monprojet/tags/1.0.0 -m "TAG 1.0.0"`

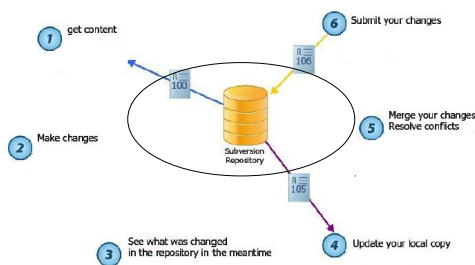
créer une branche : `svn copy https://forge-dga.jouy.inra.fr/svn/monprojet/tags/1.0.0 https://forge-dga.jouy.inra.fr/svn/monprojet/branches/1.0.1 -m "BRANCHE 1.0.1"`



Création du dépôt

La première étape consiste à importer l'ensemble de vos sources sur le serveur distant (commande **import**) .

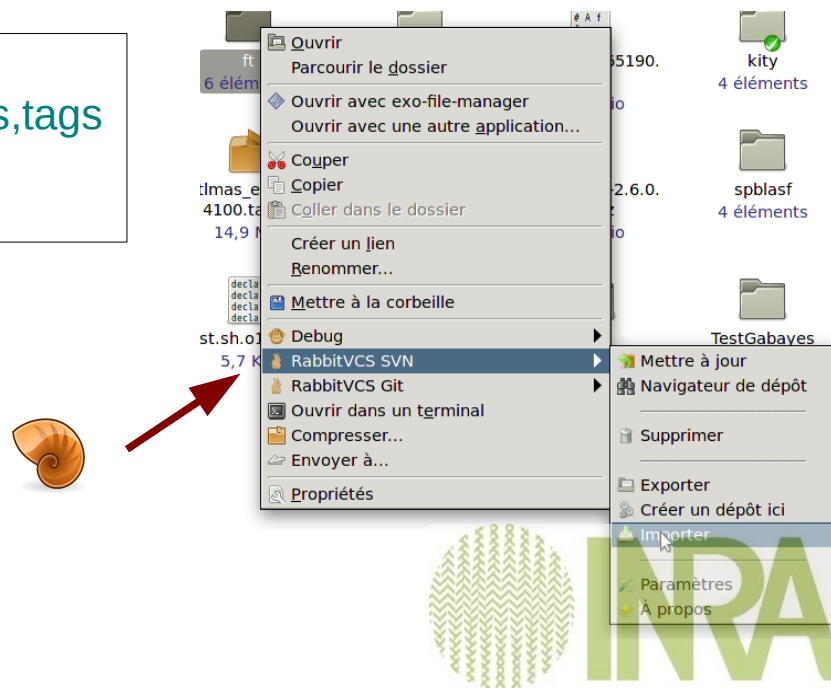
La commande **svn import** est un moyen rapide de copier une arborescence non-suivie en versions dans le dépôt, créant des dossiers intermédiaires si nécessaire.



```
svn import . https://forge-dga.jouy.inra.fr/svn/monprojet/
```

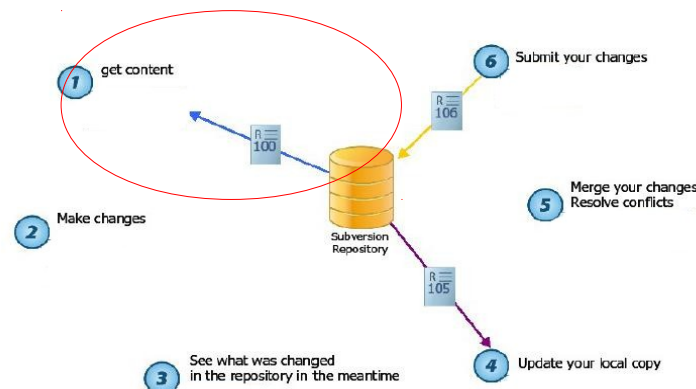
Proposition pour initialiser votre dépôt :

1. Créez une arborescence de développement **trunk,branches,tags**
2. Copiez vos sources dans le répertoire **trunk**
3. Exécutez la commande **svn import**



La copie de travail

- Arborescence classique de votre système local contenant un ensemble de fichiers
- Espace de travail personnel privé où vous pouvez éditer, compiler,...
- Chaque répertoire contient un sous-répertoire appelé *.svn* (répertoire *administratif* de la copie de travail)



Créer une copie de travail

Vous pouvez exporter votre projet entièrement :

```
svn checkout https://forge-dga.jouy.inra.fr/svn/monprojet/
```

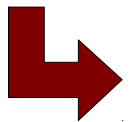
Dans la pratique :

```
svn checkout https://forge-dga.jouy.inra.fr/svn/monprojet/trunk
```

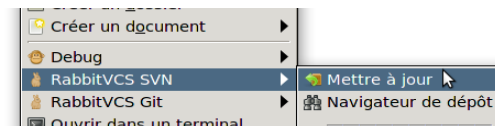
- ✓ Exporter seulement le répertoire `trunk` pour le développement de la branche principale
- ✓ Créez une branche de développement pour un développement spécifique (stagiaire,CDD).

Caractéristiques d'une branche développement :

- Maturité de la fonctionnalité
 - Ex : Une nouvelle analyse en cours d'évaluation à ne pas intégrer aux prochaines releases
- Fonctionnalités distinctes du développement de la branche principale
 - Ex : GUI

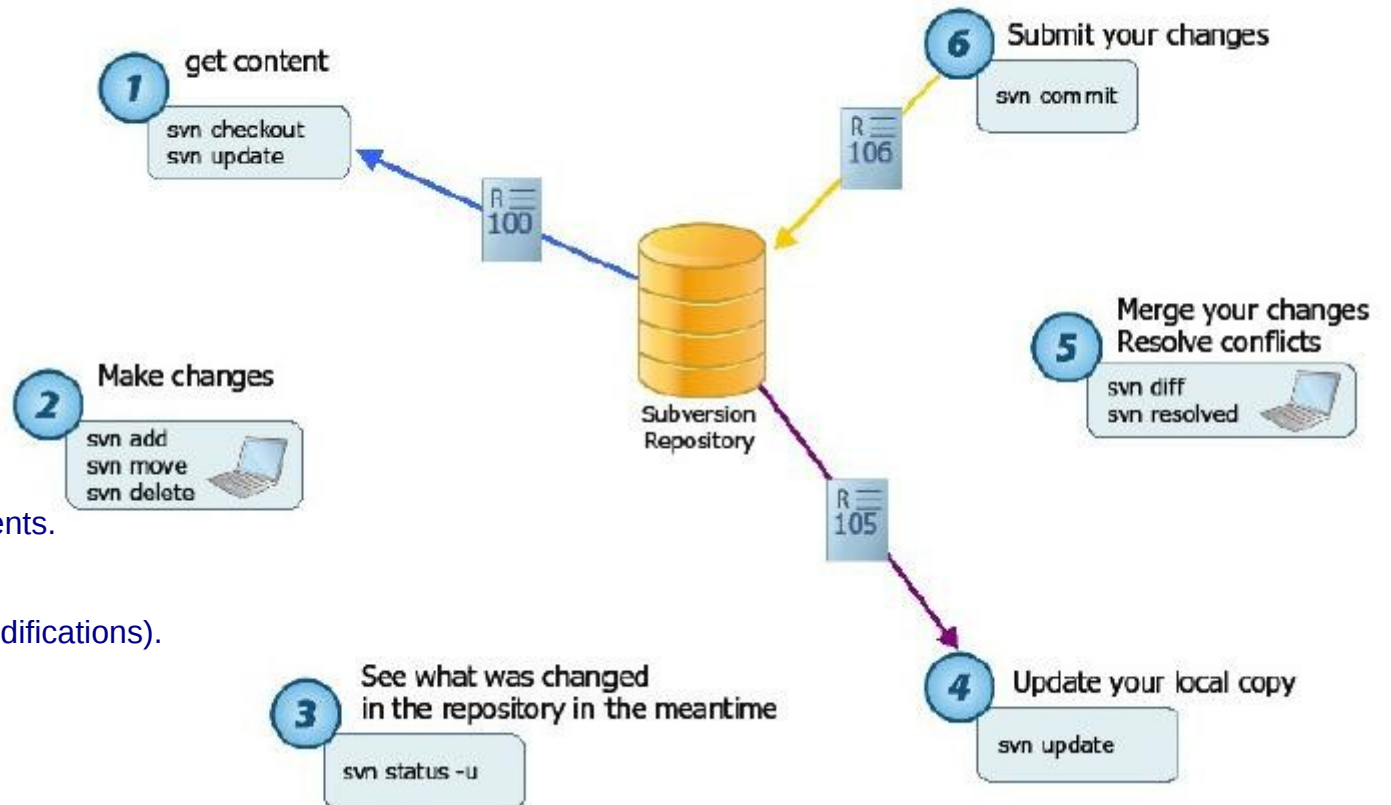


L'objectif final est de mutualiser (merge) les nouveaux développements



SVN cycle

- Mettre à jour votre copie de travail.
 - `svn update`
- Faire des changements.
 - `svn add`
 - `svn delete`
 - `svn copy`
 - `svn move`
 - `svn mkdir`
- Examiner les changements effectués.
 - `svn status`
 - `svn diff`
- Éventuellement annuler des changements.
 - `svn revert`
- Résoudre les conflits (fusionner les modifications).
 - `svn update`
 - `svn resolve`
- Propager les changements.
 - `svn commit`



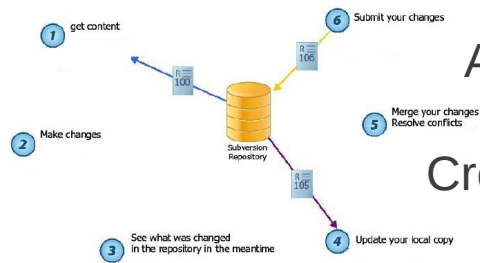
SVN au quotidien

mettre à jour votre répertoire de travail :

```
svn update
```

OU

```
svn update file1 file2 ...
```



Ajouter un fichier :

```
svn add file1
```

Créer un répertoire :

```
svn mkdir file1
```

OU

```
svn mkdir http://...
```

Effacer un fichier/répertoire :

```
svn del file1
```

Effacer un fichier/répertoire du dépôt :

```
svn del http://...
```

Copier :

```
svn copy file http://...
```

```
svn copy http://... http://...
```

```
svn copy http://... file
```

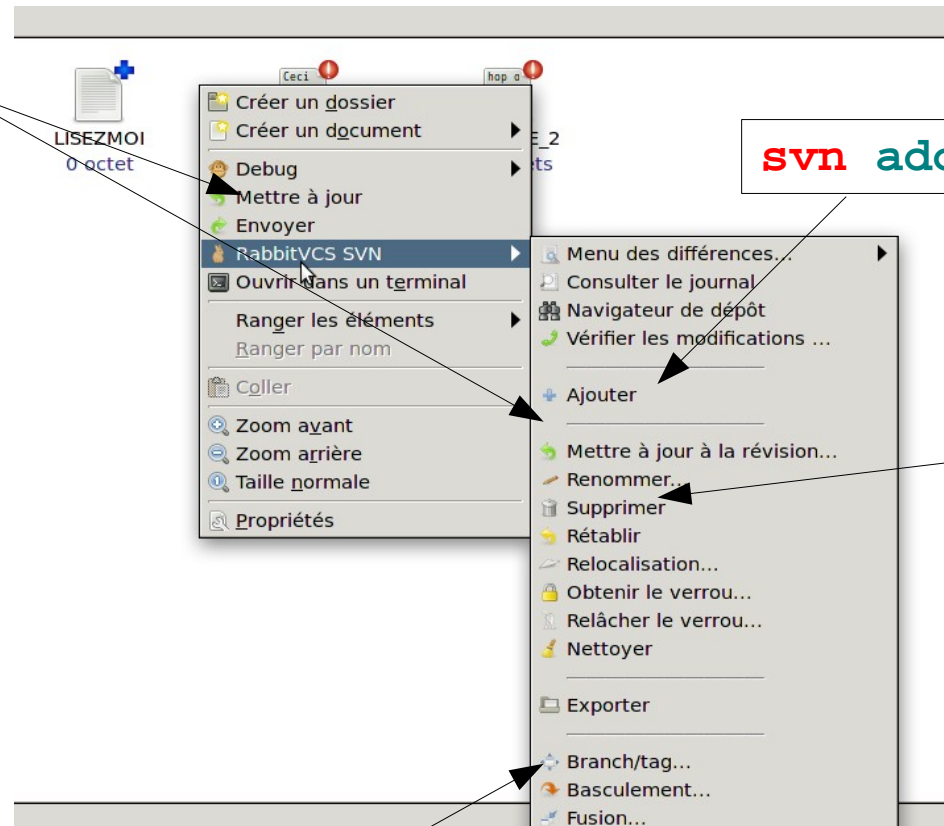
```
svn copy file1 file
```





SVN au quotidien

`svn update`



`svn add|mkdir file|rep`

`svn del file1`

`svn copy file http://...`



Propagez vos modifications



La copie de travail est une **zone de transit**.
Via une URL=> **les actions sont immédiates**

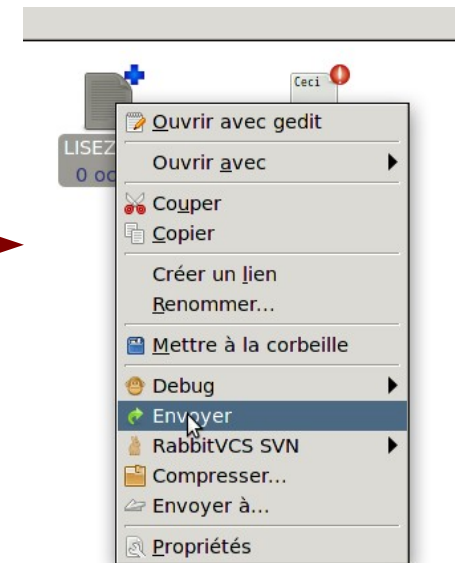
La commande `svn commit` envoie vos changements au dépôt. Quand vous propagez un changement, vous devez l'accompagner d'un message de propagation qui décrit ce changement.

```
svn commit -m"J'ai corrigé le bug."
```

Sans l'option `-m` subversion ouvre un éditeur de texte avant la propagation des modifications

Positionnez la variable d'environnement `SVN_EDITOR` pour utiliser votre éditeur préféré

```
export SVN_EDITOR = emacs
```



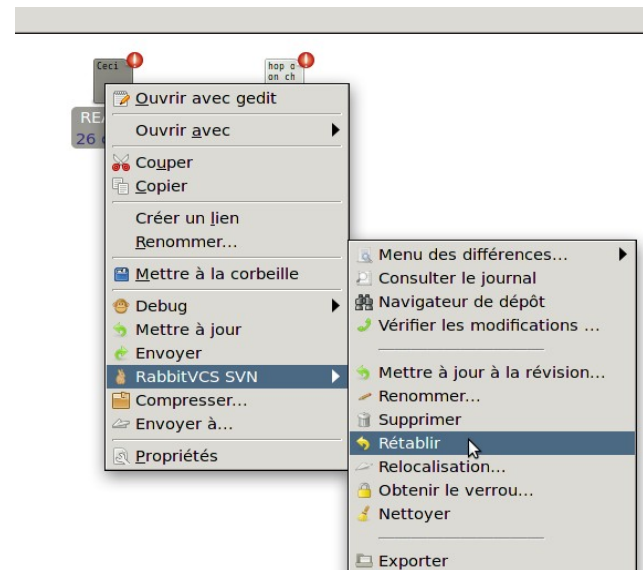
Annuler les changements sur la copie de travail

`svn revert` peut annuler n'importe quelle opération et fonctionne sur `add` et `del`

```
svn revert file
```

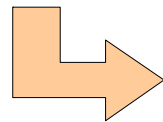
```
svn revert repertoire
```

Note : cette sous-commande n'a pas besoin d'accès réseau, et résout les conflits. Elle ne restaure cependant pas les répertoires supprimés.



Les révisions

Une opération *svn commit* publie les modifications d'un nombre quelconque de fichiers et de répertoires en **une seule opération atomique**.



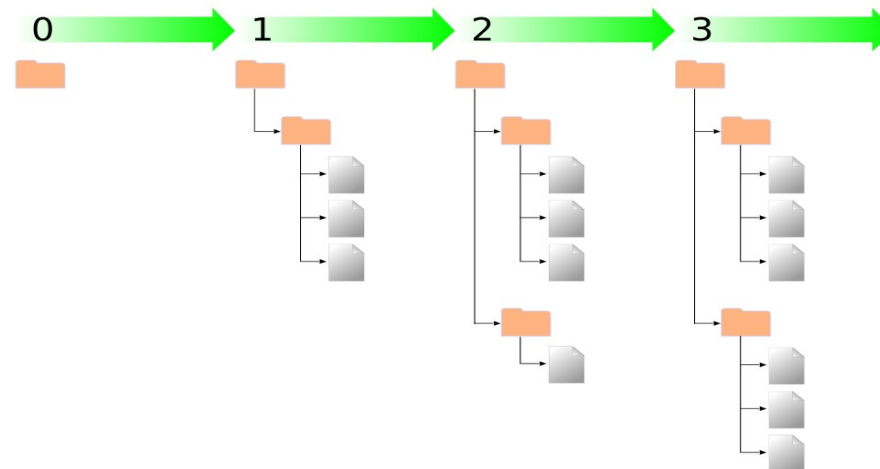
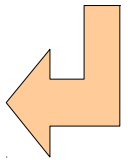
soit toutes les modifications sont propagées dans le dépôt,
soit aucune ne l'est

la révision N représente l'état du système de fichiers du dépôt après la N-ième propagation.

Quand on parle de la « révision 5 de truc.c », on veut en fait parler de « truc.c tel qu'il apparaît dans la révision 5 »

révision :

- Numéro unique
- Snapshot (photo instantanée de l'arborescence de système de fichier)



Vue d'ensemble des modifications

svn status

Si vous lancez `svn status` sans argument à la racine de votre copie de travail, Subversion détecte toutes les modifications effectuées sur les fichiers et sur l'arborescence.

```
?      gribouillage.c           # le fichier n'est pas suivi en versions
A      bazar/hello/machin.h  # le fichier sera Ajouté
C      bazar/hello/tas.c    # le fichier entre en Conflit avec une mise à jour
D      bazar/poisson.c      # le fichier sera supprimé(Deletion en anglais)
M      truc.c               # le contenu de truc.c a subi des Modifications
```

`svn status -v`



```
17:00 ofilangi@dgal12:~/kity# svn status -v
      35      35 ofilangi      .
      35      34 ofilangi      test
A     0       ?  ?          LISEZMOI
M     35      34 ofilangi      README
      35      35 ofilangi      README_2
```

Révision courante du fichier

`svn status -u`



```
17:02 ofilangi@dgal12:~/kity# svn status -u
A     0       LISEZMOI
M     35      README
      *      35      README_2
État par rapport à la révision      37
```

Dernière révision qui a affectée le fichier

Le fichier est obsolète : une nouvelle version existe dans le dépôt

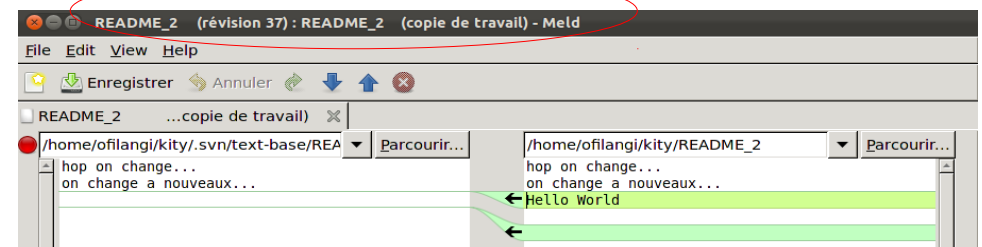
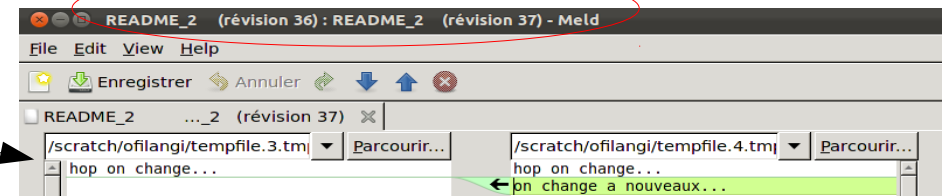


Examiner les changements

`svn diff`

`svn diff` permet de voir en détail les modifications apportées à un fichier
Par défaut , entre le fichier local et le dépôt sinon entre révisions (`-r`)

```
svn diff prog.f
svn diff --revision 1:2 prog.f
svn diff --revision PREV:COMMITTED prog.f --diff-cmd=meld
svn diff prog.f --diff-cmd=meld
```



HEAD : dernière révision du **dépôt**
BASE : rév. de base de la **copie de travail**
COMMITTED : dernière propagation (**dépôt**)
PREV : révision juste avant COMMITTED (**dépôt**)



Interrogation du dépôt par la ligne de commande

La commande `svn list` liste les fichiers présents dans le dépôt :

```
svn list https://forge-dga.jouy.inra.fr/svn/form-ctig-svn
```

Examiner une version antérieure d'un fichier :

```
svn cat -r R LISEZMOI.txt
```

Ou dans le dépôt :

```
svn cat https://forge-dga.jouy.inra.fr/svn/form-ctig-svn/all/LISEZMOI.txt
```

Afficher l'historique d'un fichier (login/date/révision affectée) :

```
svn log https://forge-dga.jouy.inra.fr/svn/form-ctig-svn/all/LISEZMOI.txt
```

Afficher les informations de révisions et d'auteurs en plus du contenu :

```
svn blame https://forge-dga.jouy.inra.fr/svn/form-ctig-svn/all/LISEZMOI.txt
```





TP

Les stagiaires doivent être membre du projet [form-ctig-svn](#)

TP Cycle Base

1. Créez un répertoire à votre nom, sans copie de travail (CT), dans le répertoire [all](#) du projet [form-ctig-svn](#)
2. Créez les répertoires [trunk](#), [branches](#) et [tags](#) dans ce répertoire (sans CT)
3. Créez une CT du répertoire [trunk](#) sur la machine DGA12
4. Créez dans votre CT, les fichiers suivants :
[README](#),[hello_world.f90](#)
5. Propagez ces modifications dans le dépôt
6. Vérifiez l'existence des fichiers [README](#),[hello_world.f90](#) de votre voisin dans son répertoire [trunk](#)
7. Créez une CT du répertoire [trunk](#) de votre voisin, modifiez le fichier [README](#) et propagez cette modification

```
program Hello_World
    print *, "Hello World !"
end program Hello_World
```



Résoudre les conflits après un update qui tourne mal...

Updated

```
$ svn update
```

```
U INSTALL
```

```
G LISEZMOI
```

merGed

Conflit découvert dans 'machin.c'.

Sélectionner : (p) report, (df) diff complet, (e) édite,
(h) aide pour plus d'options :

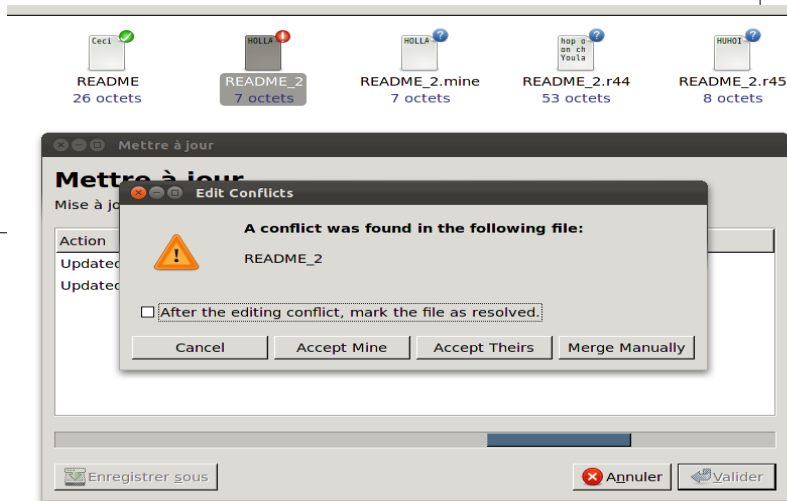
(p) report	- marque ce conflit pour résolution ultérieure
(df) diff-complet	- montre toutes les différences du fichier fusionné
(e) édite	- résout manuellement le conflit avec un éditeur
(r) résolu	- utilise la version fusionnée
(mf) mien complet	- utilise ma version (ignore les autres éditions)
(tf) autre complet	- prends la version du dépôt (perds mes éditions)
(l) lance	- utilise un outil externe pour résoudre le conflit
(h) aide	- affiche cette liste

```
...
Sélectionner : (p) report, (df) diff-complet, (e) édite,
(h) aide pour plus d'options : d
--- .svn/text-base/sandwich.txt.svn-base      mar. 11 déc. 2007, 21:33:57
+++ .svn/tmp/tempfile.32.tmp                  mar. 11 déc. 2007, 21:34:33
@@ -1 +1,5 @@
-Achète-moi un sandwich.
+<<<<<<< .mine
+Va chercher un hamburger.
+=====
+Apporte moi un taco !
+>>>>>>> .r32
...
```

Pour chaque fichier en conflit, Subversion place trois fichiers supplémentaires non-suivis en versions dans votre copie de travail :

- *nom_du_fichier.mine*
- *nom_du_fichier.rANCIENNE_REVISION*
- *nom_du_fichier.rNOUVELLE_REVISION*

```
$ ls -l
sandwich.txt
sandwich.txt.mine
sandwich.txt.r1
sandwich.txt.r2
```



Résoudre les conflits à la main

Exemple :

Par manque de communication entre Sally et Harry, le fichier *sandwich.txt* est édité en même temps . Sally propage ses changements et, quand Harry met à jour sa copie de travail, un conflit apparaît, que Harry doit résoudre en éditant *sandwich.txt*.

```
$ cat sandwich.txt
Tranche de pain supérieure
Mayonnaise
Laitue
Tomate
Comté
<<<<<<< .mine
Saucisson
Mortadelle
Jambon
=====
Choucroute
Poulet rôti
>>>>>>> .r2
Moutarde
Tranche de pain inférieure
```



```
Tranche de pain supérieure
Mayonnaise
Laitue
Tomate
Comté
Saucisson
Mortadelle
Jambon
Moutarde
Tranche de pain inférieure
```

```
$ svn resolve --accept working sandwich.txt
Conflit sur 'sandwich.txt' résolu
$ svn commit -m "Va pour le sandwich de Harry et au diable celui de Sally !"
```



Abandonner ou revenir en arrière

Abandonner vos modifications au profit de la révision la plus récente

```
svn resolve --accept theirs-full CHEMIN-DU-CONFLIT
```

```
$ svn update
Conflit découvert dans 'machin.c'.
Sélectionner : (p) report, (df) diff complet, (e) édite,
              (h) aide pour plus d'options :
C   sandwich.txt
Actualisé à la révision 2.
$ ls sandwich.*
sandwich.txt sandwich.txt.mine sandwich.txt.r2 sandwich.txt.r1
$ svn resolve --accept theirs-full sandwich.txt
Conflit sur 'sandwich.txt' résolu
```

Revenir en arrière : utiliser svn revert (restaure l'état initial d'un fichier)

```
$ svn revert sandwich.txt
'sandwich.txt' réinitialisé
$ ls sandwich.*
sandwich.txt
```





Résoudre les conflits à la main

1

A context menu is open over a file named 'README_2.r44'. The menu items include 'Ouvrir avec gedit', 'Ouvrir avec', 'Couper', 'Copier', 'Créer un lien', 'Renommer...', 'Mettre à la corbeille', 'Debug', 'Mettre à jour', 'RabbitVCS SVN', 'Compresser...', 'Envoyer à...', and 'Propriétés'. The 'RabbitVCS SVN' sub-menu is expanded, showing options like 'Menu des différences...', 'Consulter le journal', 'Navigateur de dépôt', 'Vérifier les modifications...', 'Mettre à jour à la révision...', 'Renommer...', 'Supprimer', 'Edit conflicts', 'Mark as Resolved', 'Relocalisation...', 'Obtenir le verrou...', and 'Nettoyer'. The 'Edit conflicts' option is highlighted.



2

A 'Edit Conflicts' dialog box is shown with the message: 'A conflict was found in the following file: README_2'. Below the message are checkboxes for 'After the editing conflict, mark the file as resolved.' and buttons for 'Cancel', 'Accept Mine', 'Accept Theirs', and 'Merge Manually'. To the right, a 'Meld' window is open, showing a side-by-side comparison of 'README_2*.README_2.head*' and 'README_2*.README_2.head*'. The left pane shows 'Titre1' and the right pane shows 'Titre2'. The 'Meld' window has a menu bar with 'File', 'Edit', 'View', and 'Help', and buttons for 'Enregistrer', 'Annuler', and navigation arrows.

3

A context menu is open over a file named 'README_2.r44'. The menu items are similar to the first menu. The 'RabbitVCS SVN' sub-menu is expanded, and the 'Mark as Resolved' option is highlighted.

4

A context menu is open over a file named 'README_2.r44'. The menu items are similar to the first menu. The 'Envoyer' option is highlighted.





TP

- 1) Modifier votre programme pour afficher « Hello World. My name is <votre nom> »
- 2) Propager cette modification
- 3) Modifier le programme de votre voisin avec le même affichage.
- 4) Propager cette modification(*)

(*) Exécutez cette action chez un autre voisin si vous n'avez pas de conflit....

Reprendre après une interruption

Quand Subversion modifie votre copie de travail, il effectue les actions suivantes :

- Écrit dans un fichier de trace ses intentions
- Place un verrou
- applique les modifications demandées
- Supprime le fichier de trace et le verrou

En cas de plantage (du serveur, de votre machine), SVN peut terminer ces opérations
En utilisant la commande `cleanup`

```
$ svn status
L    un-repertoire
M    un-repertoire/machin.c

$ svn cleanup
$ svn status
M    un-repertoire/machin.c
```



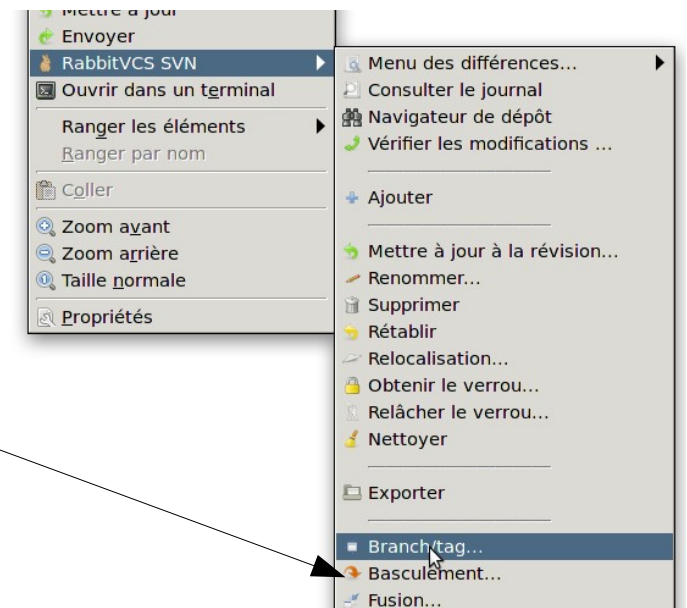
Gestion des branches et release

```
svn copy https://forge-dga.jouy.inra.fr/svn/monprojet/trunk  
https://forge-dga.jouy.inra.fr/svn/monprojet/tags/1.0 -m « version-1.0 »
```

Subversion n'a pas de notion interne de branche ou d'étiquette :
il sait seulement faire des copies.

Garder une branche synchronisée :

```
$ pwd  
/home/user/ma-branche-calc  
  
$ svn merge http://svn.exemple.com/depot/calc/trunk  
--- Fusion de r345 à r356 dans '.':  
U bouton.c  
U entier.c
```



Export d'une version sans prise en charge SVN

Créer une copie non versionnée d'une arborescence.

```
svn export https://forge-dga.jouy.inra.fr/svn/monprojet/trunk
```

On peut obtenir les sources d'un logiciel si celui-ci est un projet public de la forge en utilisant cette commande



Ignorer des fichiers et des répertoires

Utile pour ne pas se retrouver avec beaucoup de ? lors des `svn status`

Pour ignorer les fichiers d'extension .o, utilisez la commande `svn propset` avec la propriété `svn:ignore`

```
$ svn propset svn:ignore «*.o» .  
Propriété 'svn:ignore' définie sur '.'
```

Pour ignorer un répertoire :

```
$ svn propset svn:ignore tmp .
```

Via la ligne de commande et votre éditeur préféré :

```
$ svn propedit svn:ignore
```




svn:keyword property

```
$ svn propset svn:keywords «IdName» <fichier>
```

La chaîne \$Id\$ sera remplacé par le nom du fichier, le numéro de révision, l'auteur et la date de modification

```
! simple BayesCpi implementation
! $Id$
program main
  use mod_struct
  use mod_lecture
  use mod_calcul
  use mod_random
#ifdef MANAGE_OMP
  use omp_lib
```

- Date
- Revision
- Author
- HeadURL
- Id



```
1 ! simple BayesCpi implementation
2 !
3 ! $Id: main.F95 8 2012-11-23 10:42:57Z ofilangi $
4
5 program main
6   use mod_struct
7   use mod_lecture
8   use mod_calcul
9   use mod_random
10
11 #ifdef MANAGE_OMP
```

```
11:41 ofilangi@URUBU:~/workspace-new/gabayes-svn/src$ svn propset svn:keywords "Id" main.F95
Propriété 'svn:keywords' définie sur 'main.F95'
11:41 ofilangi@URUBU:~/workspace-new/gabayes-svn/src$ svn commit . -m"Ajout de la propriété id dans main.F95"
Envoi      src/main.F95
Transmission des données .
Révision 7 propagée.
```



Ce qu'on ne verra pas...

- Les verrous
- Gestion des correctifs (patch)

