La gestion de Version

J.-M. Bruel

IUT de Blagnac

2012-2013

bruel@irit.fr

Plan

- Pourquoi gérer les versions
- Concepts et définitions
- Quels outils
- Exemple
- Concrètement

Pourquoi?

- historique des fichiers d'un projet
- retour à une version antérieure
- historique des modifications
- accès souple aux fichiers
- travail collaboratif

Pourquoi subversion

- multiplateforme
- logiciel libre
- fonctionnement centralisé
- utilisation et administration faciles
- supporte plusieurs modes d'accès :
  - SSH
  - WebDAV
  - Apache

Pourquoi `git`

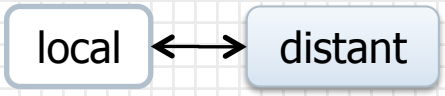
- multiplateforme
- logiciel libre + Linus Torvald
- pas de centralisation
- rapidité
- pas besoin de serveurs

subversion

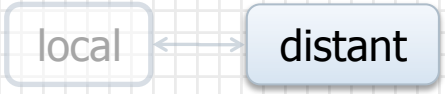
- Disponible à l'IUT

Concepts et définitions

- Termes
  - Dépôt (repository)
  - Projets
  - Copie de travail
  - Révisions
- Opérations
  - checkout
  - import
  - update
  - commit



Concepts et définitions

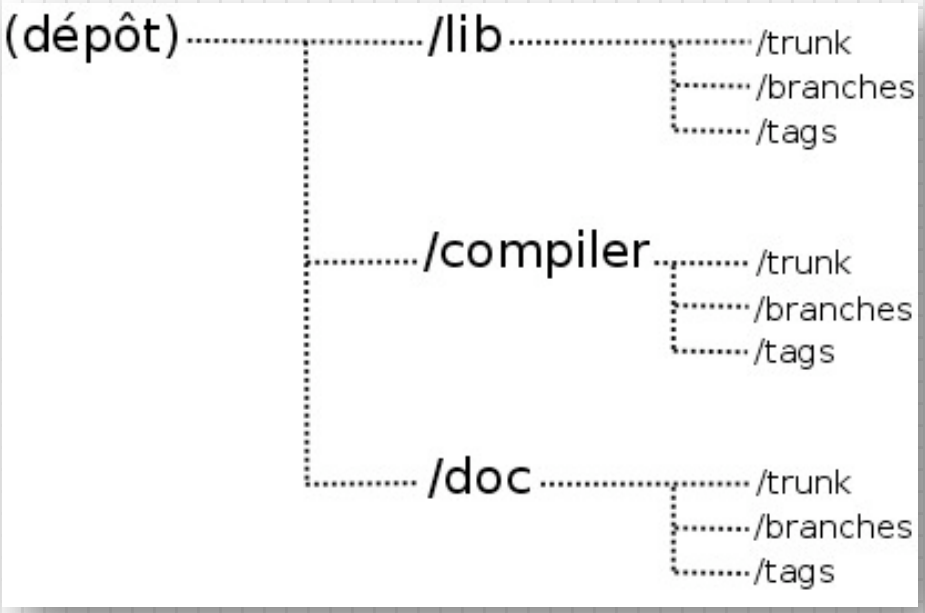


- Termes
  - Dépôt (repository)
    - emplacement central
    - contient tout :
      - données
      - historiques, versions, modifications, dates, auteurs, …

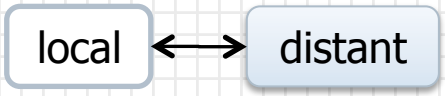
Concepts et définitions

local ⟷ distant

- Termes
  - Dépôt (exemples)



Concepts et définitions

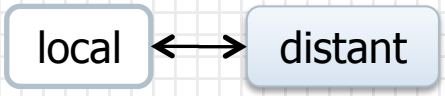


- Termes
  - Projets
    - en général un répertoire
    - contient lui-même dossiers et fichiers

Concepts et définitions

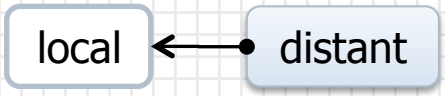
- Termes
  - Copie de travail
    - répertoire local
    - contient une copie

Concepts et définitions



- Termes
  - Révisions
    - correspond à une modification
    - à chaque modification => une révision
    - numéro de 1 à N (+1 à chaque modification)

Concepts et définitions



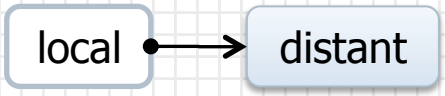
- Opérations
  - checkout

    Pour récupérer pour la 1ère fois les fichiers

    Sous-entend qu'ils sont déjà sur le serveur

    Le résultat est une copie de travail

Concepts et définitions



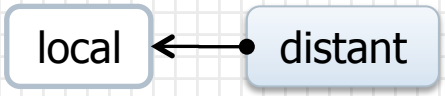
- Opérations
  - import

    Inverse du checkout

    Placer des fichier locaux dans le dépôt

    En général une fois par projet

Concepts et définitions

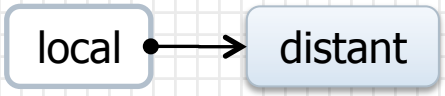


- Opérations
  - update

    Mettre à jour la copie locale (synchronisation)

    Peut provoquer des conflits

Concepts et définitions



- Opérations
  - commit

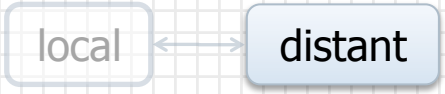
    Mettre à jour le dépôt

    Inverse du update

    Peut provoquer des conflits

    Peut nécessiter un update

Scénario type



1. Créer un dépôt

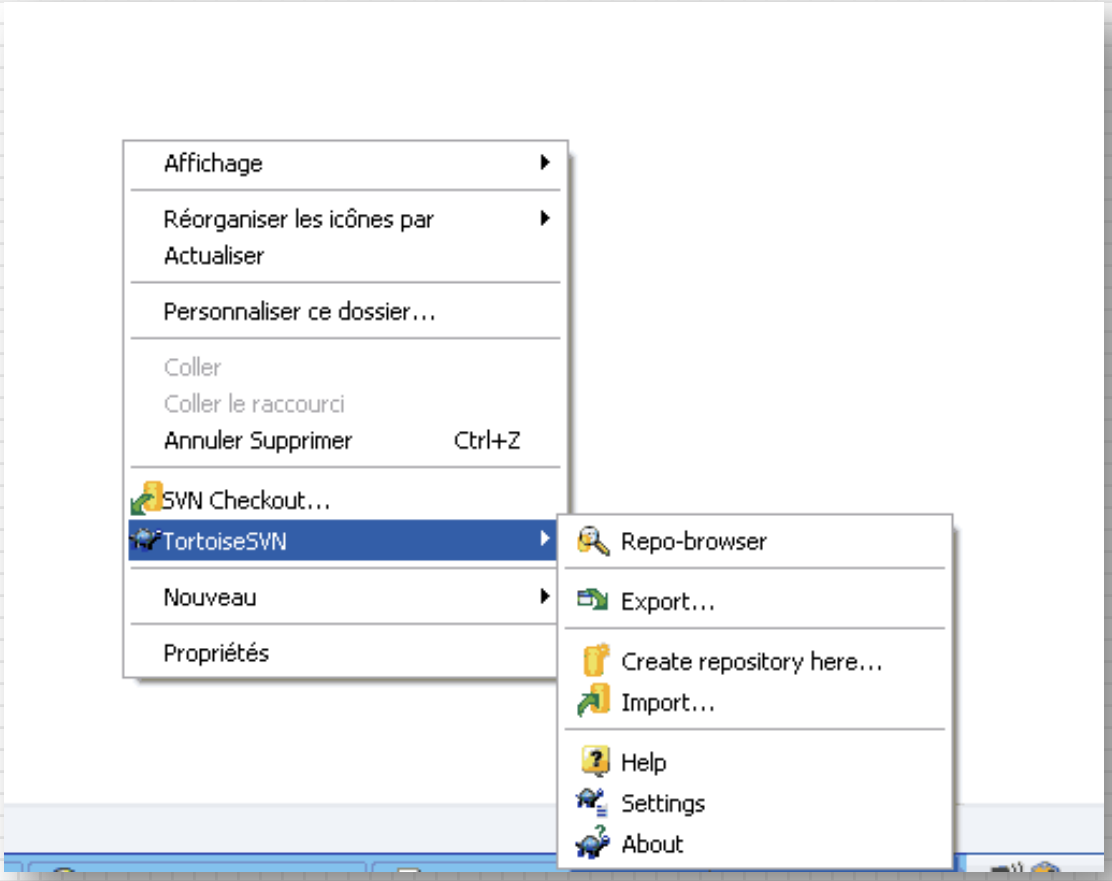
En général à partir d'un répertoire de travail local existant (droits admin)

```
svn import
```

Scénario type

local ⟷ distant

1. Créer un dépôt (windows)



Scénario type (suite)

local ← distant

2. Mettre à jour ses données locales

- Soit elles existent déjà

```
svn update
```

- Sinon un 1er checkout crée la copie locale

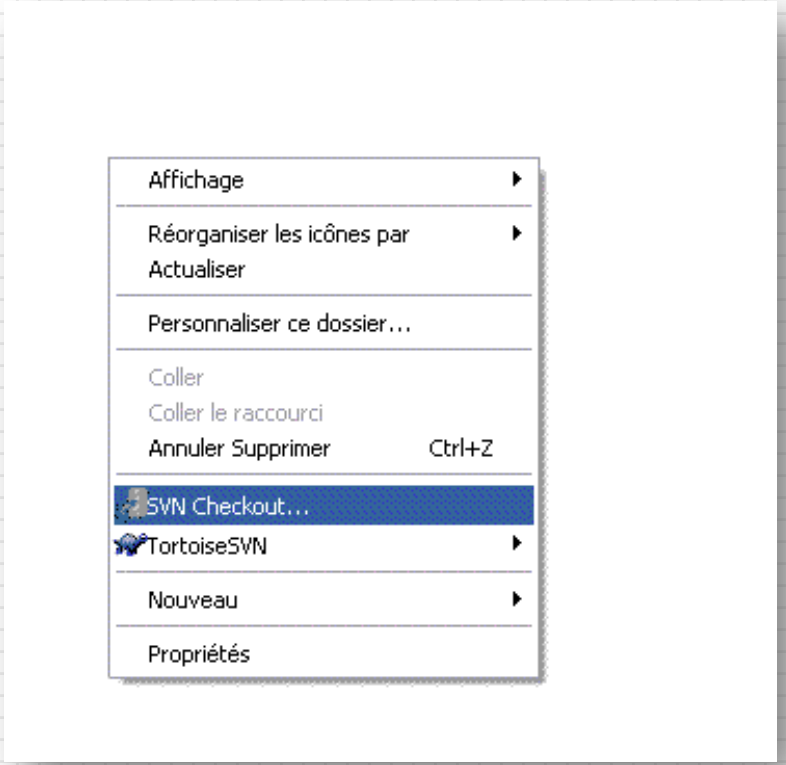
```
svn co https://test.com/svn/test .
```

Dépôt (distant)

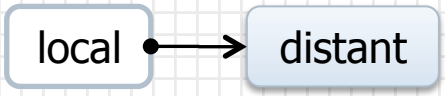
Copie (locale)

Scénario type (suite)

2. Créer/Mettre à jour ses données locales (Windows)



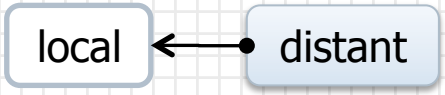
Scénario type (suite)



3. Mettre à jour ses modifications sur le serveur

```
svn commit –m "message perso"
```

Opérations particulières



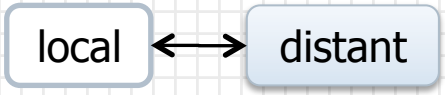
- Revenir à la dernière version "valide"

```
svn revert tp5.pl
```

- Revenir à une version particulière

```
svn update -r 12 tp5.pl
```

Opérations particulières (suite)

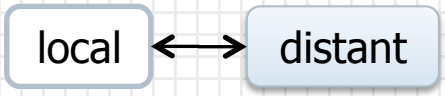


- Ajouter un fichier en local!

~~touch tp6.pl~~

```
touch tp6.pl
…
svn add tp6.pl
svn commit –m "ajout de tp6.pl"
```

Opérations particulières (suite)

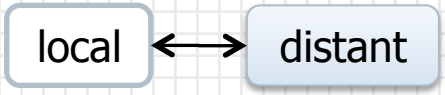


- Supprimer un fichier en local!

~~`rm tp6.pl`~~

```
svn delete tp6.pl
svn commit –m "suppression de tp6.pl"
```

Opérations particulières (suite)



- Renommer un fichier en local!

~~`rm tp5.pl tp6.pl`~~

```
svn move tp5.pl tp6.pl
svn commit –m "renommage de tp5.pl"
```

Résolution de conflits

- Travail à plusieurs => conflits!
- Exemple :
  1. Update
  2. Travail sur la copie locale du fichier X.c
  3. Commit par un collègue sur le fichier X.c
  4. Notre commit échoue!
  5. => update pour resynchroniser
  6. Commit pour intégrer notre modif

Résolution de conflits

- Exemple :

```
$ svn update
Conflict discovered in 'sandwich.txt'.
Select: (p) postpone, (df) diff-full, (e) edit,
        (h)elp for more options : p
C  sandwich.txt
Updated to revision 2.
$ ls -1
sandwich.txt
sandwich.txt.mine
sandwich.txt.r1
sandwich.txt.r2
```

Résolution de conflits (suite)

- Travail à plusieurs => conflits!
- Exemple :
  1. Update
  2. Travail sur la copie locale du fichier X
  3. Commit par un collègue sur le fichier X
  4. Notre commit échoue!
  5. => update pour resynchroniser
  6. Commit pour intégrer notre modif

Résolution de conflits

- Exemple :

```
$ svn update
Conflict discovered in 'sandwich.txt'.
Select: (p) postpone, (df) diff-full, (e) edit,
        (h)elp for more options : p
C  sandwich.txt
Updated to revision 2.
$ ls -1
sandwich.txt
sandwich.txt.mine
sandwich.txt.r1
sandwich.txt.r2
```

Résolution de conflits

- Si l'update échoue
  - 2 copies locales du fichier X.c
    - X.c.mine : copie locale avant l'update
    - X.c.r12 : version du fichier X.c pour la révision 12
    - X.c.r13 : version actuellement dans le dépôt
    - X.c : version de synthèse
  - Une fois les conflits résolus (sur X.c)

```
svn resolved X.c
```

Résolution de conflits (suite)

- Publier les modifications
  - Notion de « patch »
    1. Création du fichier de différences

```
svn -diff X.c > patch1.0
```

Résolution de conflits (suite)

Publier les modifications

Nom du fichier

Fichier dépôt

Fichier local

Conflit sur :
- 7 lignes du dépôt
- 12 lignes du local

```
$ svn diff
Index: bar.c
===================================================================
--- bar.c       (revision 3)
+++ bar.c       (working copy)
@@ -1,7 +1,12 @@
+#include <sys/types.h>
+#include <sys/stat.h>
+#include <unistd.h>
+
+#include <stdio.h>

 int main(void) {
-  printf("Sixty-four slices of American Cheese...\n");
+  printf("Sixty-five slices of American Cheese...\n");
 return 0;
 }
```

Résolution de conflits (suite)

- Publier les modifications
  - Notion de « patch »
    1. Création du fichier de différences

```
svn -diff X.c > patch1.0
```

    2. Création du patch

```
patch -p0 patch1.0
```

Résolution de conflits (suite)

- Publier les modifications

```
$ patch -p0  < patchfile
Patching file integer.c using Plan A...
Hunk #1 succeeded at 147.
Hunk #2 succeeded at 164.
Hunk #3 succeeded at 241.
Hunk #4 succeeded at 249.
done
```

Concepts avancés

- Troncs (trunks)
  - Version « officielle » en général
- Branches
  - Développement secondaire
- Tags
  - Permet de marquer une version stable
- Forges
  - Dépôts organisés (outils de suivis)

Opérations complémentaires

- Aide en ligne

  ```
  svn help
  ```

- Informations sur la copie

  ```
  svn info
  ```

- Historique d'un fichier

  ```
  svn –log X.c
  ```

- Statut de la copie

  ```
  svn status -v
  ```

- Parcours du dépôt

  ```
  svn import
  ```

Opérations complémentaires

- Qui a fait quoi?

  `svn blame X.c`

```
$ svn blame hello.c
12          steve #include <stdio.h>
10          greg
10           greg int
10           greg main (int argc, char *argv)
10           greg {
11         robert printf ("hello world \n");
10           greg return 0;
10           greg }
$
```

- Rendre un projet (en TP)

  `svn export . Tpfinal`

  => création du fichier `Tpfinal.tar.gz`

Côté serveur

- Administration du dépôt

  `svnadmin`

Côté serveur : exemple

- Création d'un « hook »
  - But : recevoir un mail à chaque commit

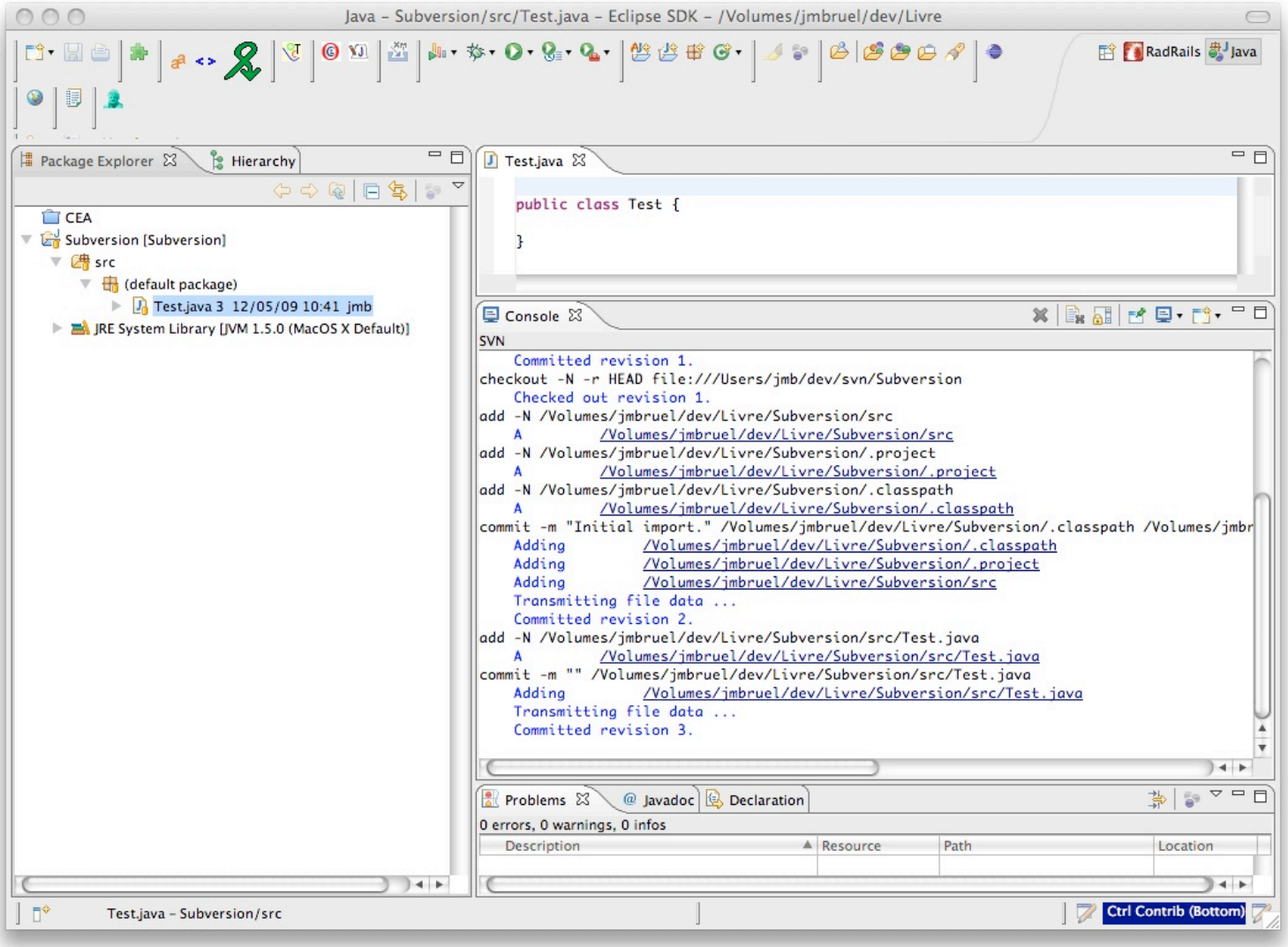
http://www.kitpages.fr/svn_overview.html

```
#!/bin/sh

REPOS="$1"
REV="$2"

SUBJECT=`\
  echo -n "[SVN commit] R-$REV Log:";\
  svnlook log --revision \
        $REV /home/webadmin/htdocs/repository | head -n1`
MESSAGE=`\
  echo -n "REPOS:$REPOS ; REV:$REV ; AUTHOR:";\
  svnlook author --revision $REV /home/webadmin/htdocs/repository; \
  echo ; echo "LOG:";\
  svnlook log --revision $REV /home/webadmin/htdocs/repository;\
  echo;echo "LISTE DES FICHIERS:";echo "-------------";\
  svnlook changed --revision $REV /home/webadmin/htdocs/repository`

/usr/local/bin/sendEmail.pl -f subversion@kitpages.com \
  -t webmaster@kitpages.com \
  -u "$SUBJECT" -m "$MESSAGE" >> /dev/null
```

Outils clients

- commande en ligne
- TortoiseSVN (Windows)
- RapidSVN
- eSVN
- JSVN
- Subclipse
- Etc!

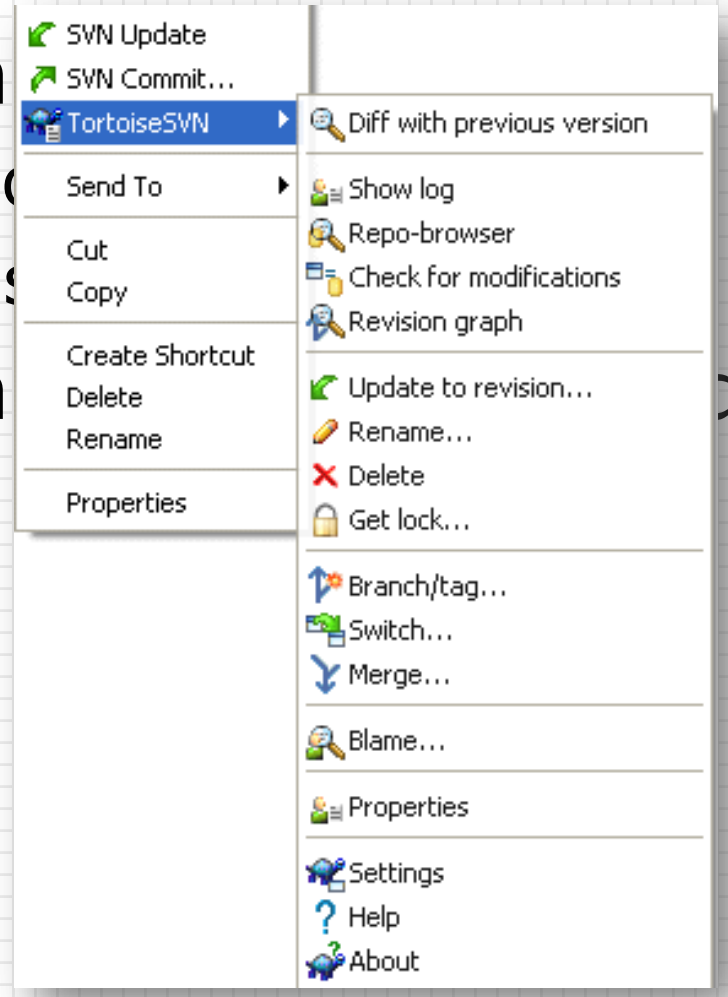
Exemple concret : eclipse



Université de Toulouse

Exemple concret : projet VB

- Utilisation
  - Générati
  - Adresse s
- Utilisation dows



GIT

- Pas disponible à l'IUT…
- … mais partout ailleurs!
- Cf. autre support Moodle

Pour finir…

- Gestion de version
  - Incontournable en entreprise
  - Un plus pour votre CV
  - Une habitude à prendre
    - Seuls en 1ère année (plusieurs machines)
    - En groupe en 2ème année (projet tut)
- CVS, Subversion, GIT, Mercury
  - Les bases sont les mêmes
  - Explorez!

Support et références

- Liens utiles
  - http://zooko.com/revision_control_quick_ref.html
  - http://better-scm.berlios.de/comparison/comparison.html
  - http://subclipse.tigris.org/update
- Quelques tutoriels :
  - http://dev.nozav.org/intro_svn.html
  - http://www.iut-arles.up.univ-mrs.fr
  - http://isaacproject.u-strasbg.fr/download/presentation_subversion.pdf
- Livres
  - En ligne : http://svnbook.red-bean.com/en/1.5/svn-book.pdf
  - Version Control with Subversion, O'Reilly.