Programmer avec Subversion

Félix-Antoine Bourbonnais

fbourbonnais@rubico.info

12 novembre 2008

Université Laval,
Québec

Partie I

Le travail collaboratif

Définitions

δ Travail concurrent :

Plusieurs personnes travaillent sur une même ressource (code ou document) dans un intervalle de temps commun.

δ Travail collaboratif :

Se dit de ce qui, dans un environnement informatisé ou en ligne, vise à favoriser la collaboration entre pairs, en permettant d'échanger et de partager des compétences pour mieux réussir un projet commun. [GDT]

√ Le ***grand dictionnaire terminologique*** (GDT) de l'Office québécois de la langue française est utilisé à •titre de référence ainsi que pour les traductions linguistiques. [www.granddictionnaire.com]

Introduction

- Les équipes de travail deviennent de plus en plus grandes
  - Difficile de fusionner et intégrer manuellement toutes les modifications
  - Les modifications concurrentes sur une même partie sont fréquentes
- Le personnel et les membres sont dispersés géographiquement
  - Impensable de prendre le téléphone pour parler à son collègue
  - Disparité des heures de travail

Introduction (2)

- La taille des projets peut être énorme
  - Difficile de gérer efficacement tous ces documents ou sources
  - Il est même parfois impossible qu'un seul être humain puisse connaître l'ensemble d'un projet (ex.: Mozilla)
- Les projets à source ouvert (OpenSource)
  - Regroupent de plus en plus de développeurs
  - Sont de plus en plus utilisés ==> augmentation de l'activité

Les besoins

- Permettre à un grand nombre de personnes de travailler :
  - Sur une même ressource (document ou bout de code)
  - En même temps (de manière concurrente)
- Avoir un système qui :
  - Fusionne automatiquement les travaux
  - Permet de revenir en arrière
  - Garde des traces des modifications

Solutions

CVS

www.cvshome.org

- **C**oncurrent **V**ersions **S**ystem
- Créé par Dick Dune en 1984-1986
- Très largement utilisé et répandu
- Gratuit / Source ouvert (OpenSource)

Subversion

subversion.tigris.org

- Remplacement à CVS (nouvelle génération)
- CVS + améliorations (principalement compatible)
- Gratuit / Source ouvert (OpenSource)

Qui utilise ces outils ?

Subversion / CVS

- La majorité des projets libres (Free/OpenSource)
  - Mozilla (Thunderbird, Firefox, ...) [CVS]
    - ~ 38800 fichiers versionnés (1/02/06)
  - KDE [SVN]
    - Révision HEAD 586883 (20/09/06)
  - OpenOffice [CVS]
    - *~700 000 « commit » dans son CVS entre octobre 2001 et janvier 2004 (1)*
  - *Subversion*
- Samba [SVN]
  - Des développeurs (ou entreprises) de logiciels
- Des particuliers pour gérer leurs documents

(1) http://stats.openoffice.org/spreadsheet/index.html

Partie II

Présentation de SVN

Qu'est-ce que SVN

Subversion

- Permet de partager des documents entre plusieurs personnes
  - Plus besoin d'envoyer les modifications par courriel!
  - Dépôt unique pour tous les documents et toutes les versions
- Conserve une copie de toutes les versions du document
  - Les différences sont conservées par rustines (patches)
  - Détection automatique des binaires Amélioration!
  - Possibilité de revenir en arrière tel qu'était le document ou le projet à une certaine date ou à une certaine révision précise
  - Permet de consulter les révisions antérieures

Qu'est ce que SVN (2)

Subversion

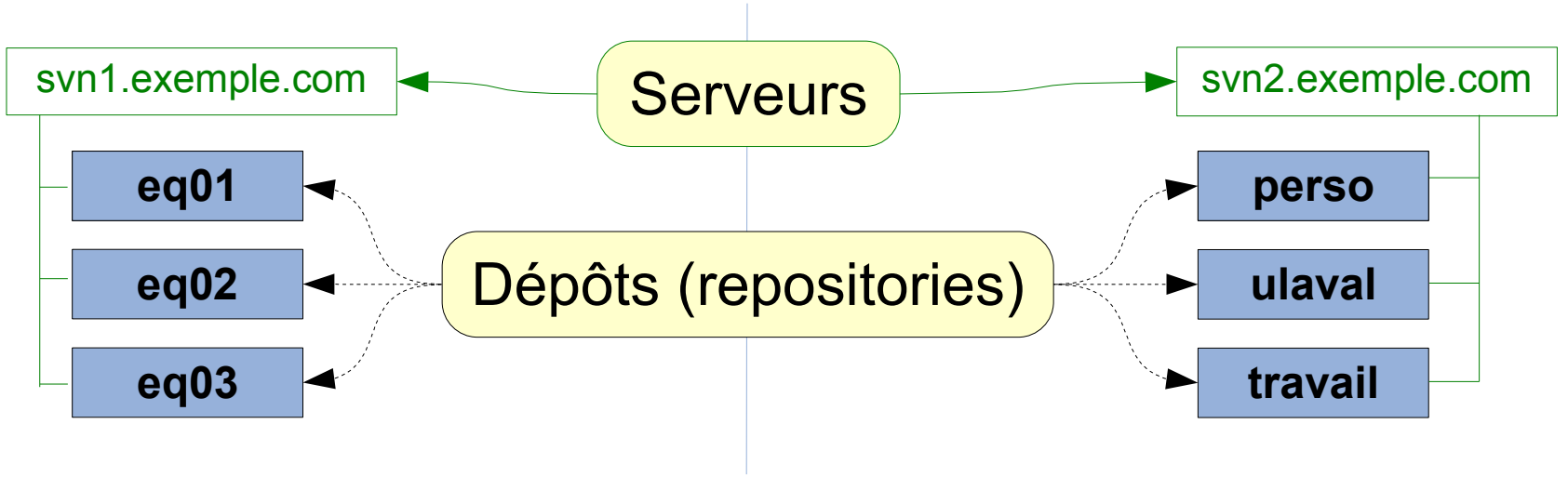
- Fusion automatisée
  - Tous les membres d'une équipe peuvent travailler sur le projet en même temps et sur les mêmes fichiers
  - La fusion est automatique si les modifications ne sont pas en conflit
  - S'il y a conflit, Subversion aide à repérer et à résoudre ce dernier
- Permet de renommer/déplacer les fichiers Amélioration!
  - Attention: Il faut éviter de déplacer ou renommer les fichiers à partir d'autres applications (ex.: Eclipse sans un plugin; directement dans Windows)
- Les répertoires sont aussi versionnés Amélioration!
- Possibilité de marquer une étape dans le développement (Tag)
- Permet de scinder temporairement ou non un projet (Branch)

Partie III

Les concepts clés

Le dépôt

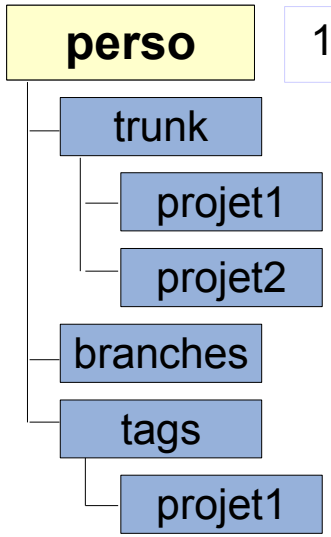
- Le dépôt (repository) :
  - Racine du Subversion (dossier parent)
  - Espace de travail dans lequel on place des projets ou sous-projets
  - Structure qui contient des fichiers, dossiers, et informations



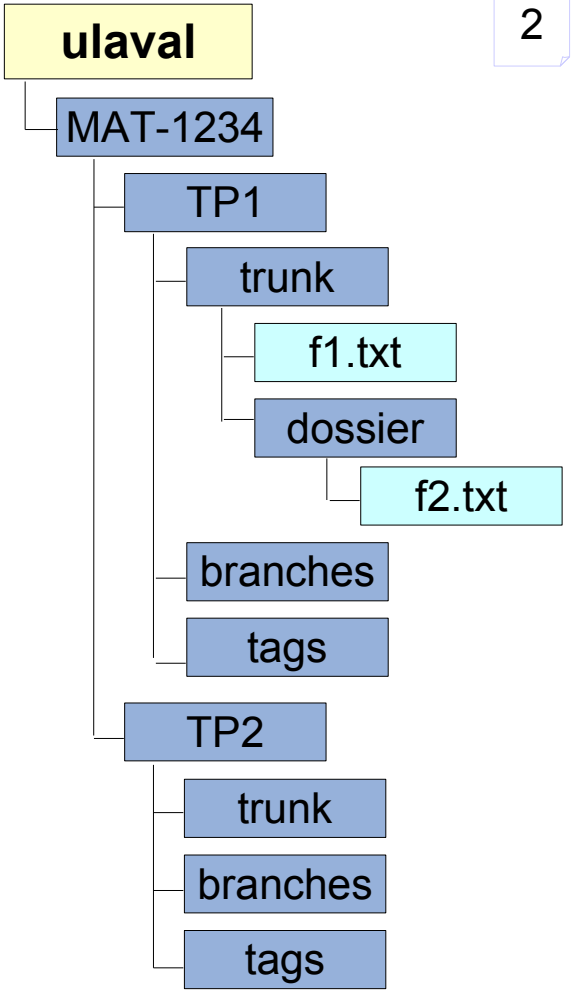
L'arborescence

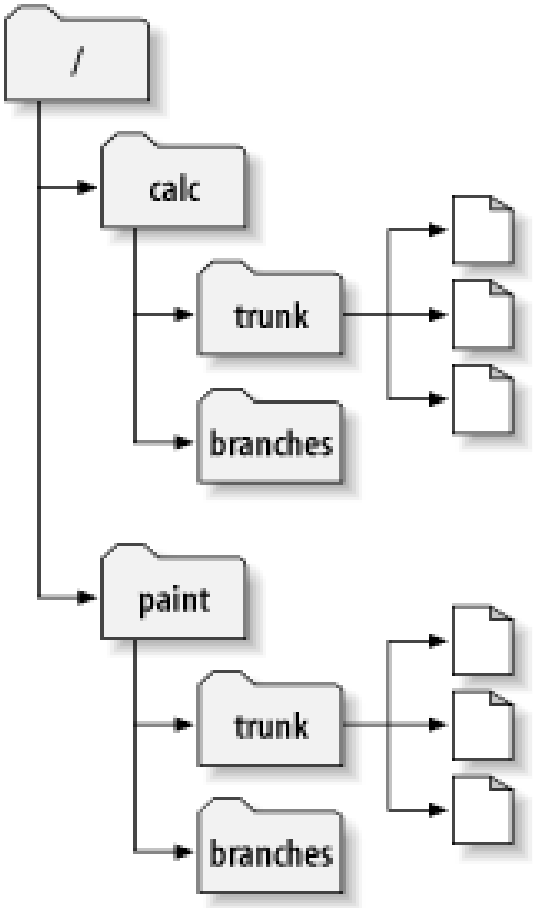
- Aucune hiérarchie n'est obligatoire pour Subversion (contrairement à CVS)
- Il y a cependant des formules recommandées:

1



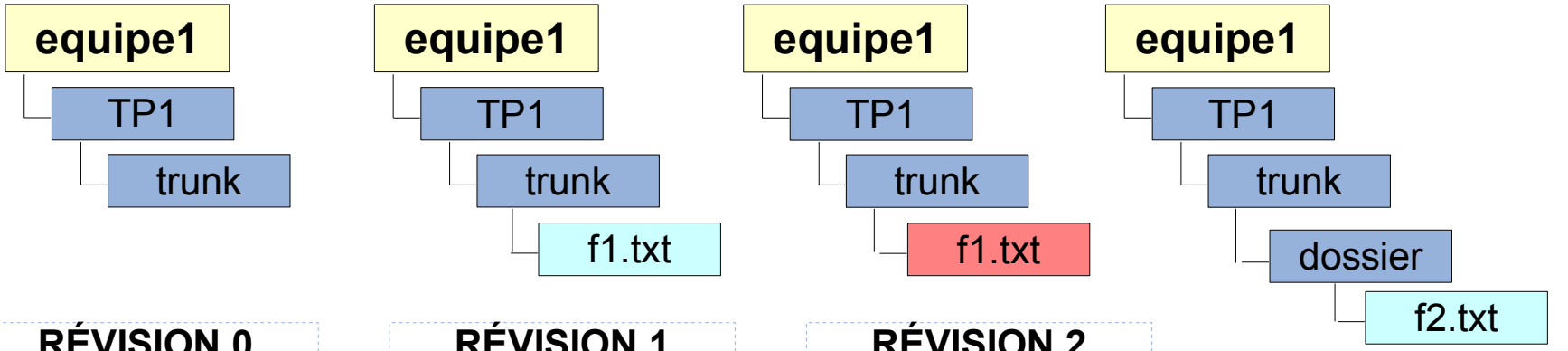
2





Les révisions

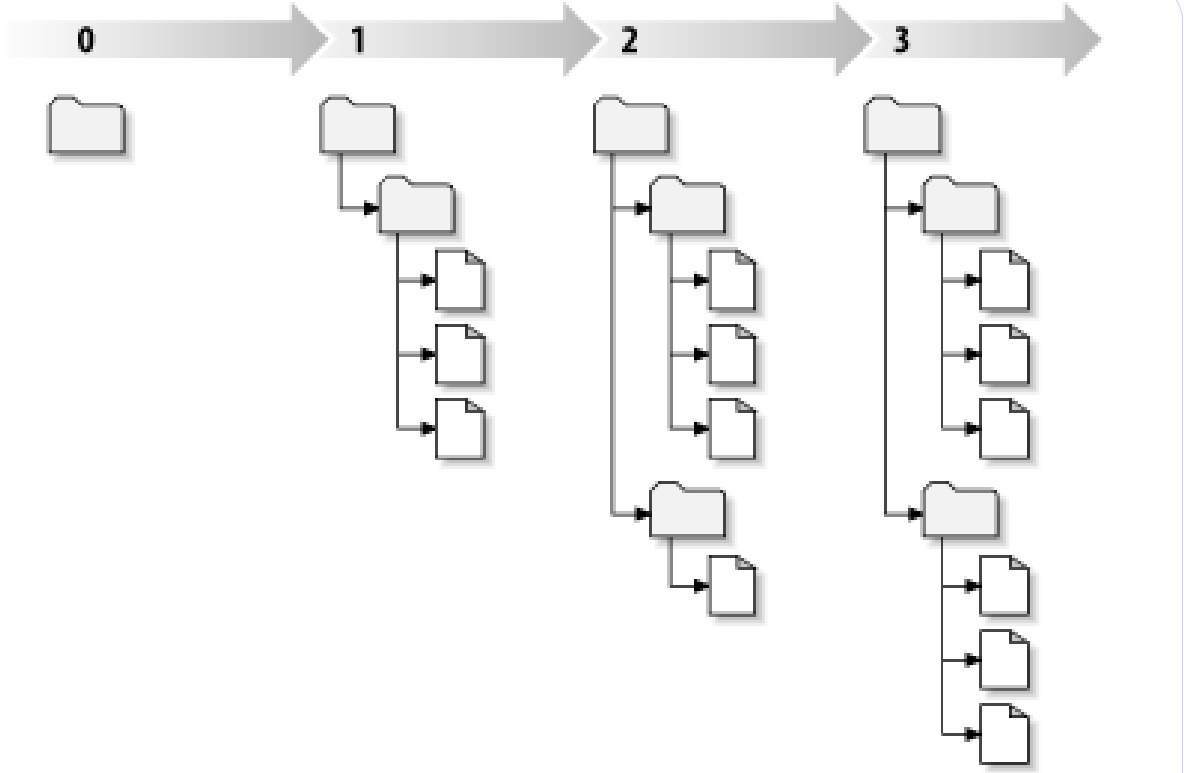
- Révisons:
  - Image de l'état du dépôt à un moment précis
  - Propre à tout le dépôt (Différent de CVS!)
  - Une nouvelle révision est créée après chaque commit accepté
  - Versions spéciales: HEAD, BASE (pristine), PREV, COMMITTED

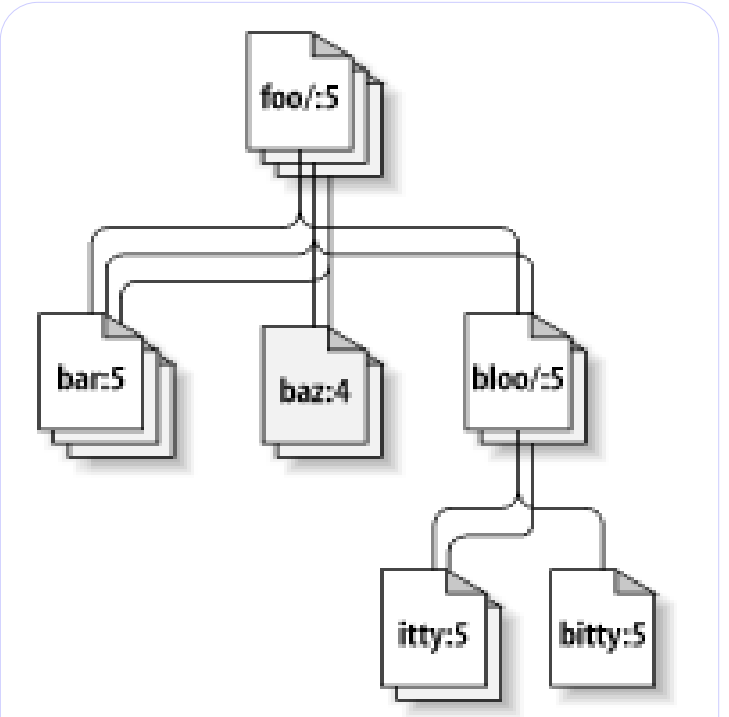


RÉVISION 0	RÉVISION 1	RÉVISION 2	RÉVISION 3
Création initiale (import)	Ajout du fichier f1.txt	Modification du fichier f1.txt	1) Effacement du fichier f1.txt 2) Ajout de dossier 3) Ajout de f2.txt

Programmer avec SVN

Révision (2)





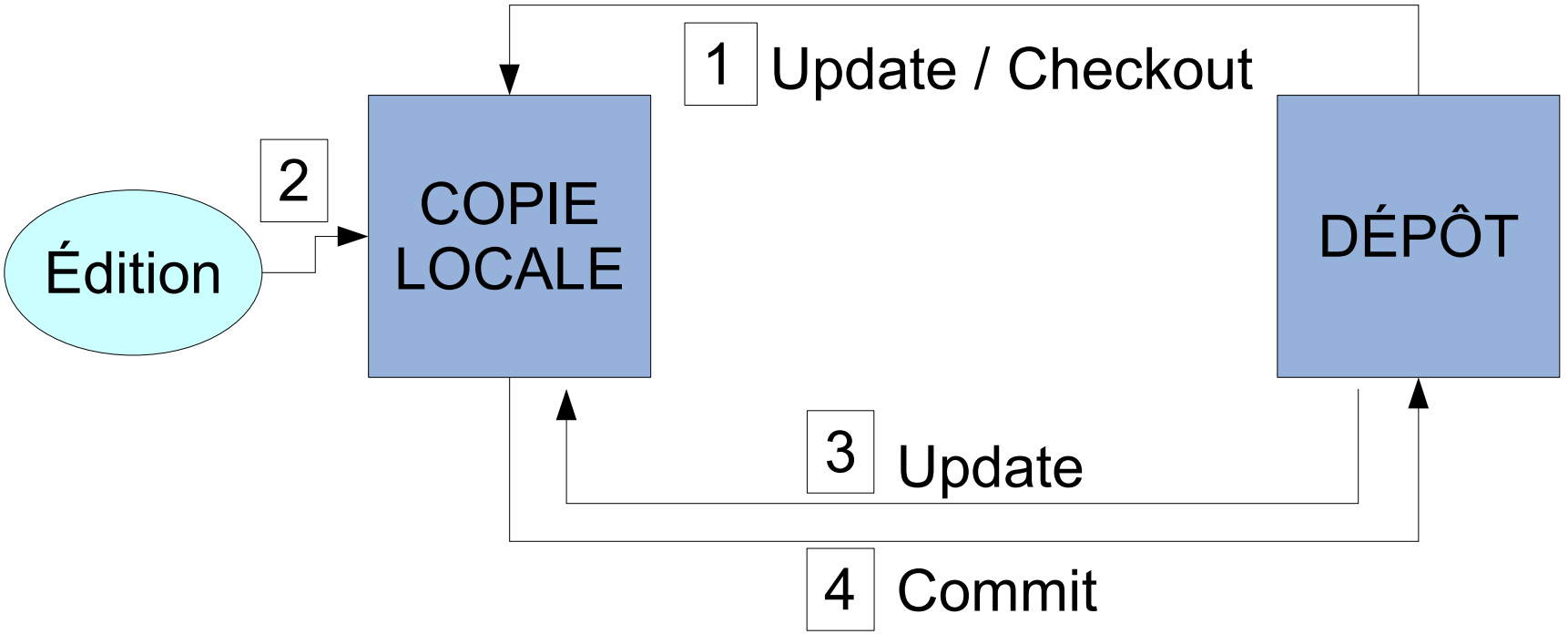
- Tags (étiquettes):
  - Permet de fixer l'état des fichiers à un instant donné par une étiquette
  - Une copie du dépôt placée (par convention) dans tags/
  - Exemple: La sortie (v1.0.0) d'une version d'un logiciel:

    logiciel1/trunk/ [Rev 586] ==COPIE==> logiciel1/branches/VERSION_1.0.0/

Différent de CVS!

Opérations

- On ne modifie jamais les fichiers directement dans le dépôt (serveur).
- Opérations à effectuer pour utiliser un dépôt existant:
  - 1- On télécharge une copie du dépôt [copie locale] 〈 Checkout/Update 〉
  - 2- On travaille (lecture et modifications) dans **la copie locale**
  - 3- On se synchronise avec le dépôt (s'il y a lieu) 〈 Update 〉
  - 4- On envoie les modifications (effectuées sur la copie locale) vers le dépôt 〈 Commit 〉

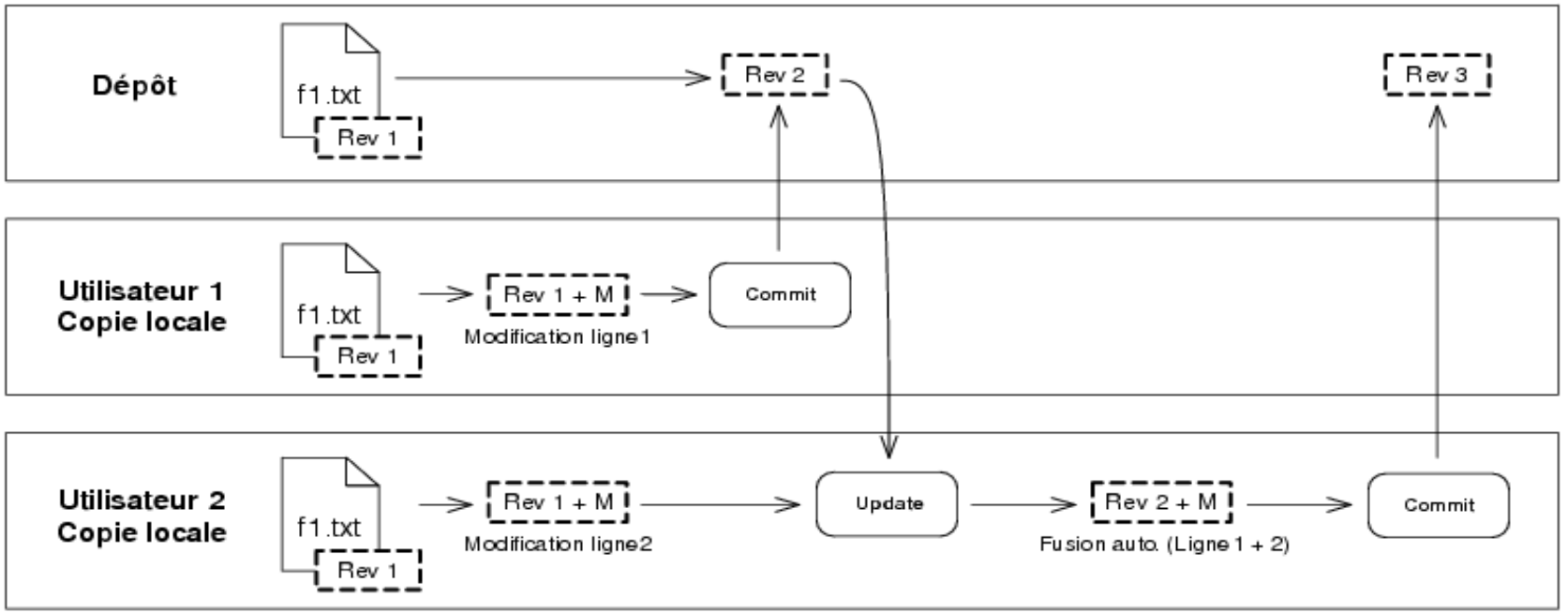


Conflits

Problème: deux personnes ont modifié le même fichier

CAS 1: Les modifications ne se recouvrent pas -et- c'est un fichier texte

- 1) Fusion automatique par Subversion
- 2) Vérifier si la fusion a du sens
- 3) Faire un commit afin d'envoyer le résultat de la fusion



Conflits (2)

CAS 2: Les modifications se recouvrent -ou- fichier binaire

- Fusion manuelle
- Ajout et modification de fichiers par SVN
  - fichier.mine (uniquement pour les fichiers textes):
    - Fichier tel qu'il était avant update (modifications locales)
  - fichier.r102:
    - Dernière révision disponible dans le dépôt (révision 102)
  - fichier.r100:
    - Révision à partir de laquelle les modifications ont été effectuées (révision 100)
  - ... suite page suivante ...

Conflits (3)

CAS 2: (suite)

- fichier:
  - [BIN] Fichier tel qu'il était avant update (modifications locales)
  - [TXT] Fusion des modifications locales avec la version du dépôt. Les conflits sont marqués avec des marqueurs.

```
Ligne 1
<<<<<<< .mine
Ligne 2 : Modification locale
=======
Ligne 2 : Modification du dépôt (r102)
>>>>>>> .r102
Ligne 3
```

Conflits (4)

- Résoudre un conflit (plusieurs possibilités):
  - a) Utiliser un utilitaire pour visualiser les modifications
    (ex.: TortoiseMerge avec TortoiseSVN, kompare, etc.)
  - b) Comparer les fichiers .rXYZ et .mine, etc.
  - c) Modification manuelle du fichier (f1.txt)
  - d) Remplacer le fichier (f1.txt) par l'un des fichiers .rXYZ ou .mine
    [Annuler les modifications ... utiliser avec prudence]
- On marque le conflit comme étant résolu (svn resolved)
  (Peut différer selon le client utilisé)

Commandes

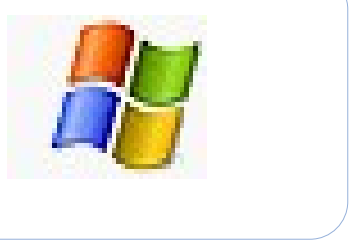
- Les principales commandes (il existe des alias de ces commandes) :
  - import : Envoi (commit) récursif avec création des parents [SVN]
  - add : Ajouter un élément dans le dépôt
  - remove: Effacer un fichier du dépôt (les anciennes versions restent!)
  - checkout: Créer la copie locale (première fois)
  - update: Mettre à jour la copie locale à partir du dépôt
  - commit: Mettre à jour le dépôt avec les modifications locales (envoi)
  - status: État des fichiers dans la copie locale (sans connexion au dépôt)

    status -u : État par rapport au dépôt (avec connexion au dépôt)
  - log: Affiche les messages de chacun des envois (commit) affectant l'élément
  - diff: Affiche les différences entre deux révisions (ou chemins pour SVN)
  - move, copy: Déplacer, renommer ou copier un élément [SVN]
  - revert: Annuler les modifications locales [SVN]

Partie IV

Utilisation de SVN

Outils



Windows

- Ligne de commande
- **TortoiseSVN** [www.tortoisesvn.org]
- Subversive (pour Eclipse) [www.polarion.org]

GNU/Linux

- **Ligne de commande**
- **KdeSVN [http://www.alwins-world.de/programs/kdesvn/]**
- RapidSVN [rapidsvn.tigris.org]
- Subversive (pour Eclipse) [www.polarion.org]

Mac OS X

- **Ligne de commande**
- svnX
- Xcode (intégration)

Outils (2)

- Il est possible de consulter (lecture) le contenu d'un dépôt via un navigateur web [HEAD uniquement]
  - Exemple: https://svn.example.com/svn/mondepot/
  - Exemple: http://svn.example.com/svn/mondepot/
- Il est possible de consulter et modifier (Subversion >= 1.2.0) via un lecteur DAV [HEAD uniquement]
  - Exemple: https://svn.example.com/svn/mondepot/ [Windows Web Folder]
  - Exemple: webdavs://svn.example.com/svn/mondepot/ [Konqueror (KDE)]
  - ...

Conseils

- Les majuscules/minuscules sont importantes (serveur unix)
  - Un fichier a.txt != A.txt sous Un*x
- Toujours entrer des messages détaillés à chaque commit
- N'attendez pas trop avant de faire un commit (une idée = un commit)
- Il est possible d'uniformiser les retours de chariots (Windows/Unix):
  - svn propset svn:eol-style "CRLF" tp2.lyx
- Il est possible d'ignorer des fichiers (ne pas les versionner) :
  - svn propset svn:ignore dossier/
- Soyez très disciplinés
  - Respectez et définissez des conventions entre les membres
  - Respectez l'ordre des opérations pour éviter des problèmes
- Éviter les caractères spéciaux et accentués dans les noms de fichiers/dossiers
- Ne placez pas des dossiers déjà versionnés dans un autre dépôt

Conseils (2)

- Quelques conseils tirés du KDE Commit Policy:

1. Think twice before committing.
2. Never commit code that doesn't compile.
3. Test your changes before committing.
4. Double check what you commit.
   [Do a "svn update" and a "svn diff" before committing]
5. Always add descriptive log messages.
6. When you plan to make changes which affect a lot of different code in CVS, announce them on the relevant mailing list in advance.
7. Take responsibilty for your commits.
   [If your commit breaks something or has side effects on other code, take the responsibility to fix or help fix the problems.]
8. Don't commit code you don't understand.
9. Don't add generated files to the repository.

Source: http://developer.kde.org/policies/commitpolicy.html

Résumé

- Vous travaillez TOUJOURS dans votre copie locale
- Quand vous ne savez plus quoi faire, demandez-vous ce que vous avez et ce que le dépôt a
- Toutes les actions et opérations effectuées sur la copie locale seront propagées (sur le dépôt) uniquement après un commit
  - Ceci inclut les opérations add et remove
  - Le tag est une opération sur le module (pas sur la copie locale)
- Vous pouvez revenir à l'état initial (dernier checkout) avec revert
- Les révisions sont globales à tout le dépôt
- Si deux personnes ont travaillé sur un même fichier:
  - Une fusion est possible: Subversion fusionne automatiquement
  - Un conflit existe: Il faut le corriger « à la main » en utilisant des outils pour faire des différences de fichiers

Partie V

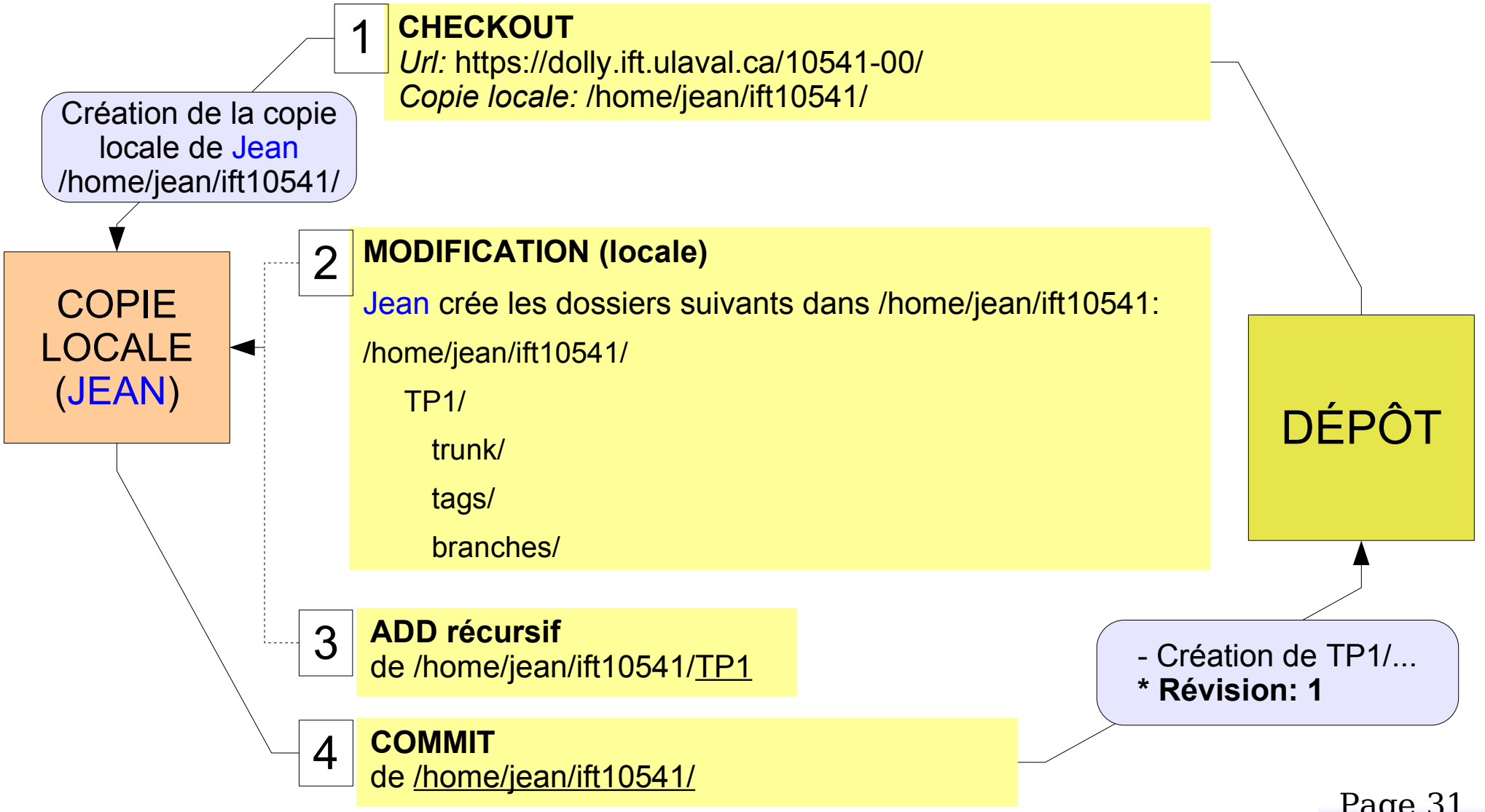
Étude de cas

Étude de cas

- Serveur:
  - Dépôt: https://dolly.ift.ulaval.ca/10541-00/
  - Méthode: HTTPS
- Utilisateurs (équipe 00) :
  - Jean
  - Paul

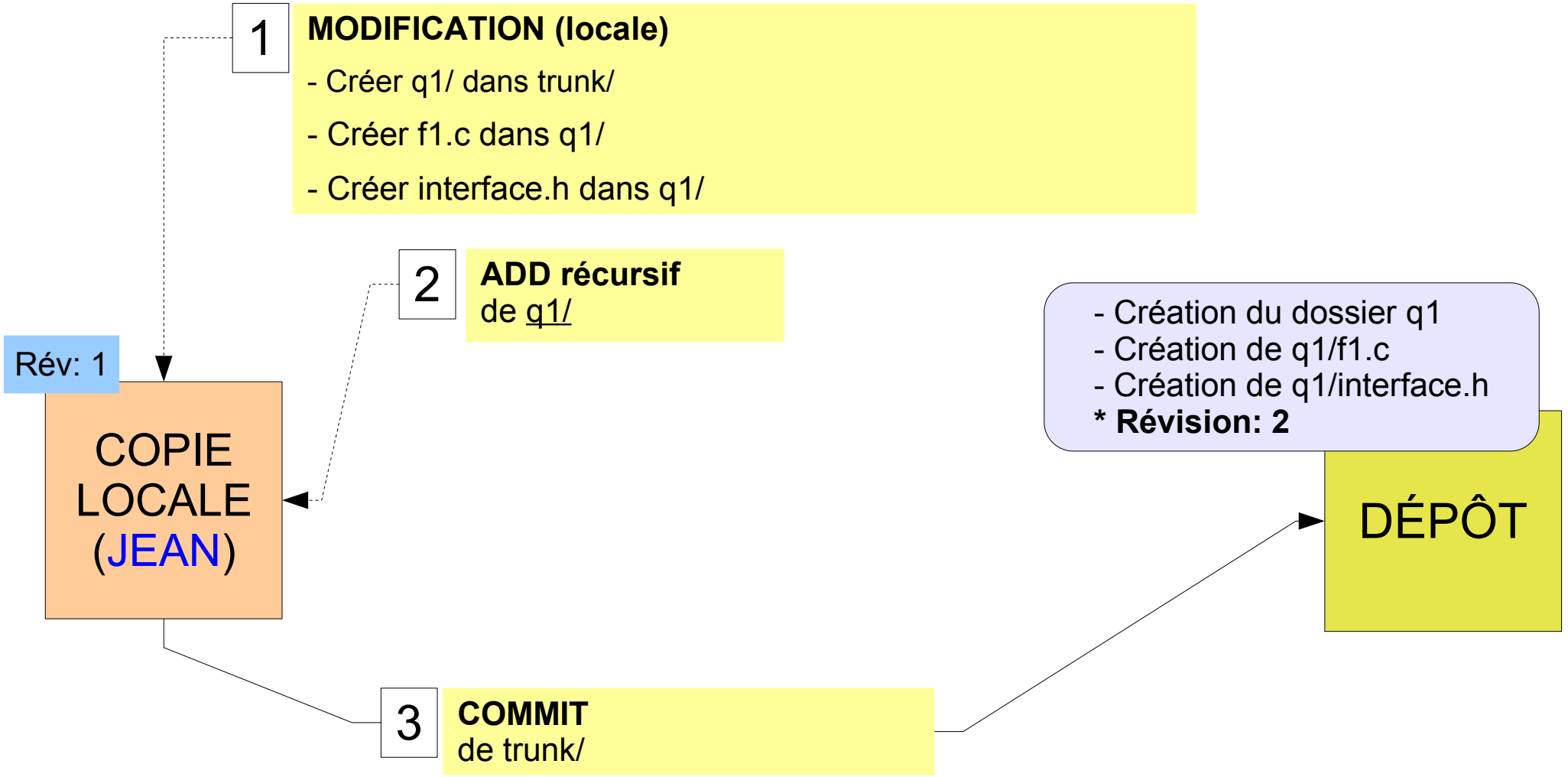
Étude de cas

Jean va créer l'arborescence pour le TP1.



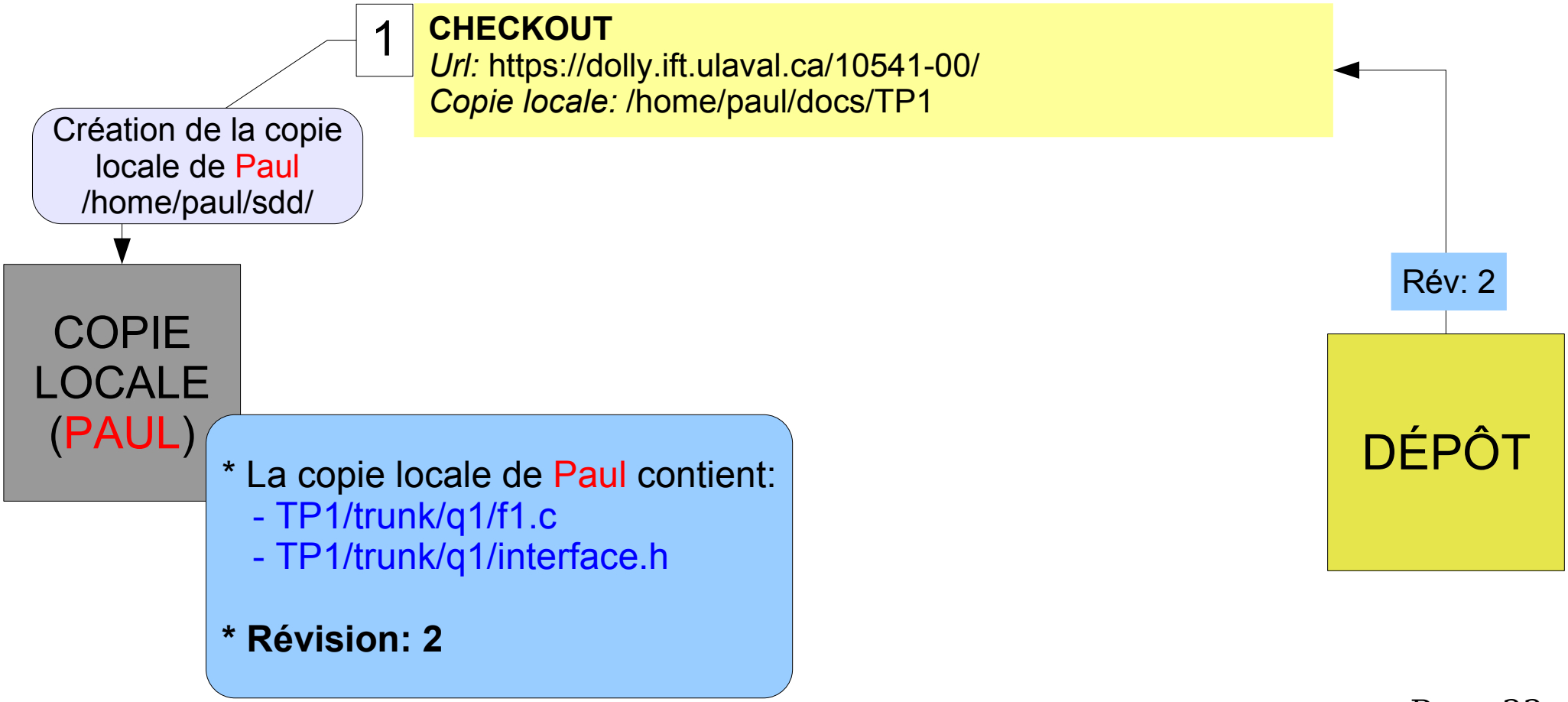
Étude de cas

Jean va créer de nouveaux fichiers et répertoires



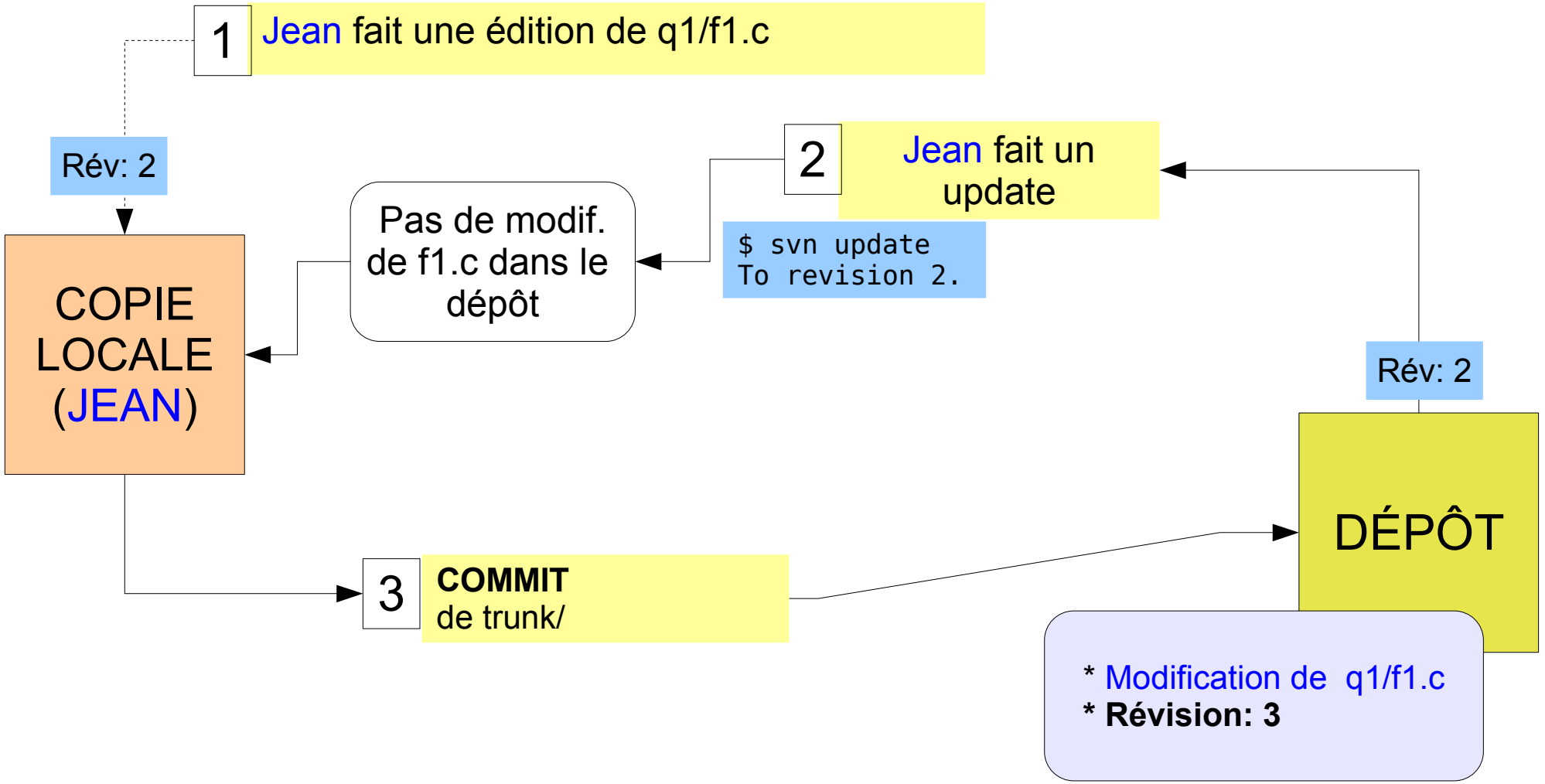
Étude de cas

Paul veut télécharger pour la première fois TP1



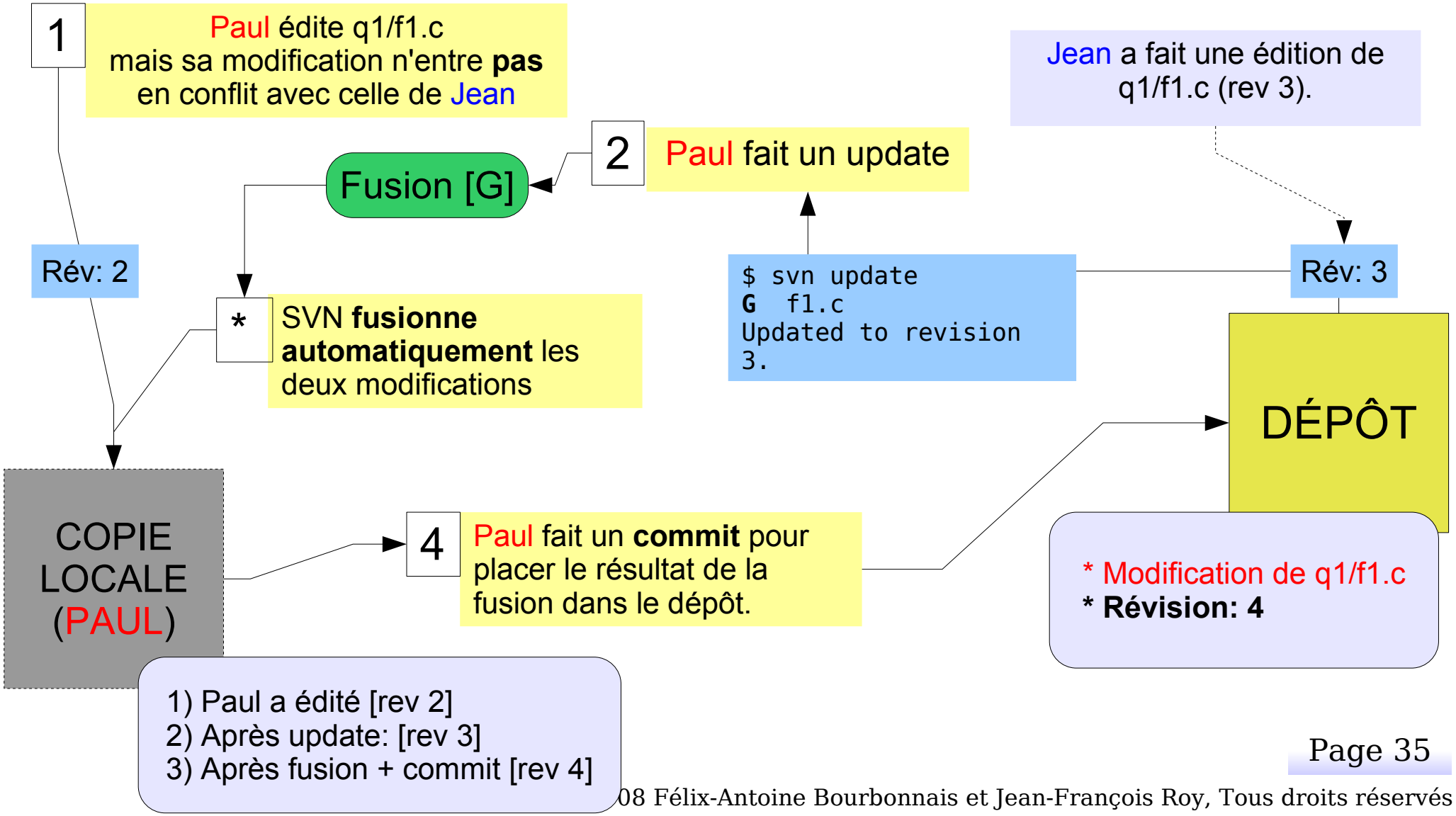
Étude de cas

Jean veut modifier le fichier f1.c



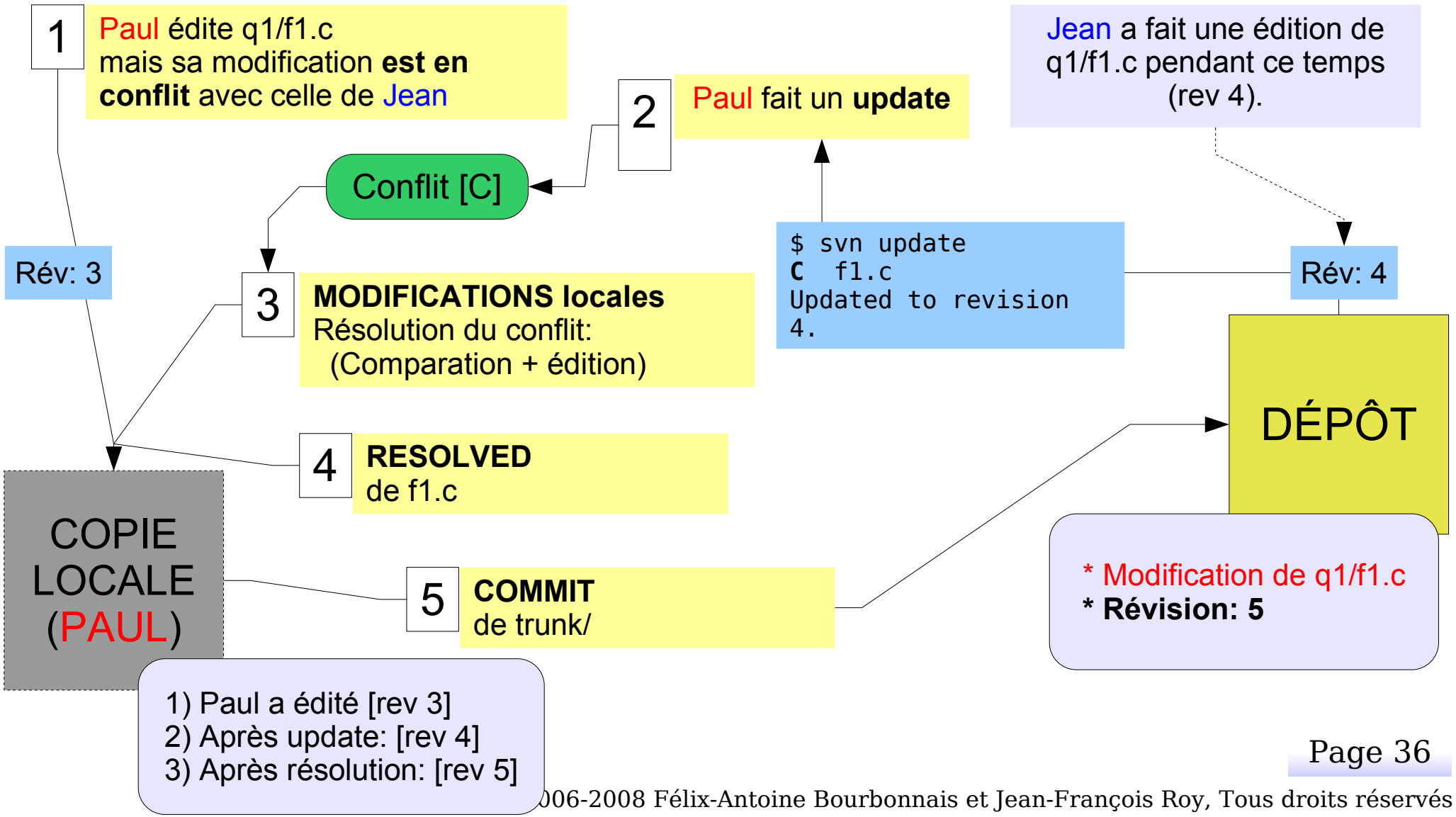
Étude de cas

Paul veut mettre sa copie locale à jour après avoir édité, lui aussi, q1/f1.c



Étude de cas

Paul veut mettre sa copie locale à jour après avoir édité, lui aussi, q1/f1.c



Partie VI

Intégration

« Hook scripts »

- Le serveur Subversion permet d'exécuter un script avant et après un certain nombre d'opérations
  - Commit, update, etc.
- On peut donc déclencher des actions lors d'événements dans un dépôt
- Exemples:
  - Envoi d'un courriel à une « mailing list » contenant le messages de commit et les fichiers modifiés
  - Compilation et exécution des tests automatiquement
  - Copie de sauvegarde du dépôt

API Subversion

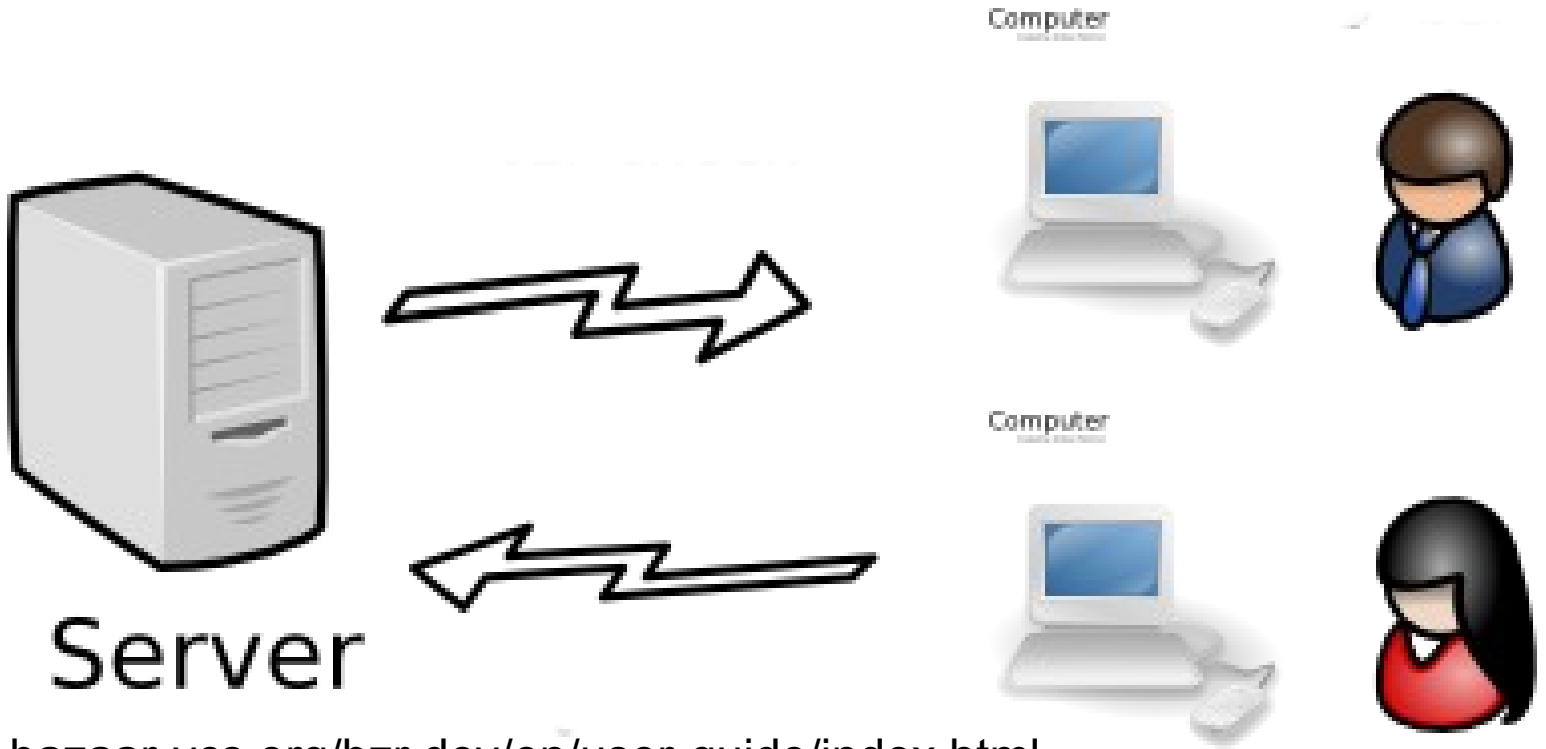
- Subversion offre une API riche pour permettre à divers clients d'opérer sur un dépôt
- Présentement, cette API est principalement utilisée pour naviguer un dépôt Subversion à partir d'une application web

Partie VII

Après Subversion

Le modèle centralisé

- Subversion utilise le modèle centralisé
  - Un dépôt maître sur un serveur connu
  - Tous les clients travaillent sur les mêmes branches
  - Le serveur doit être disponible pour effectuer toutes opérations

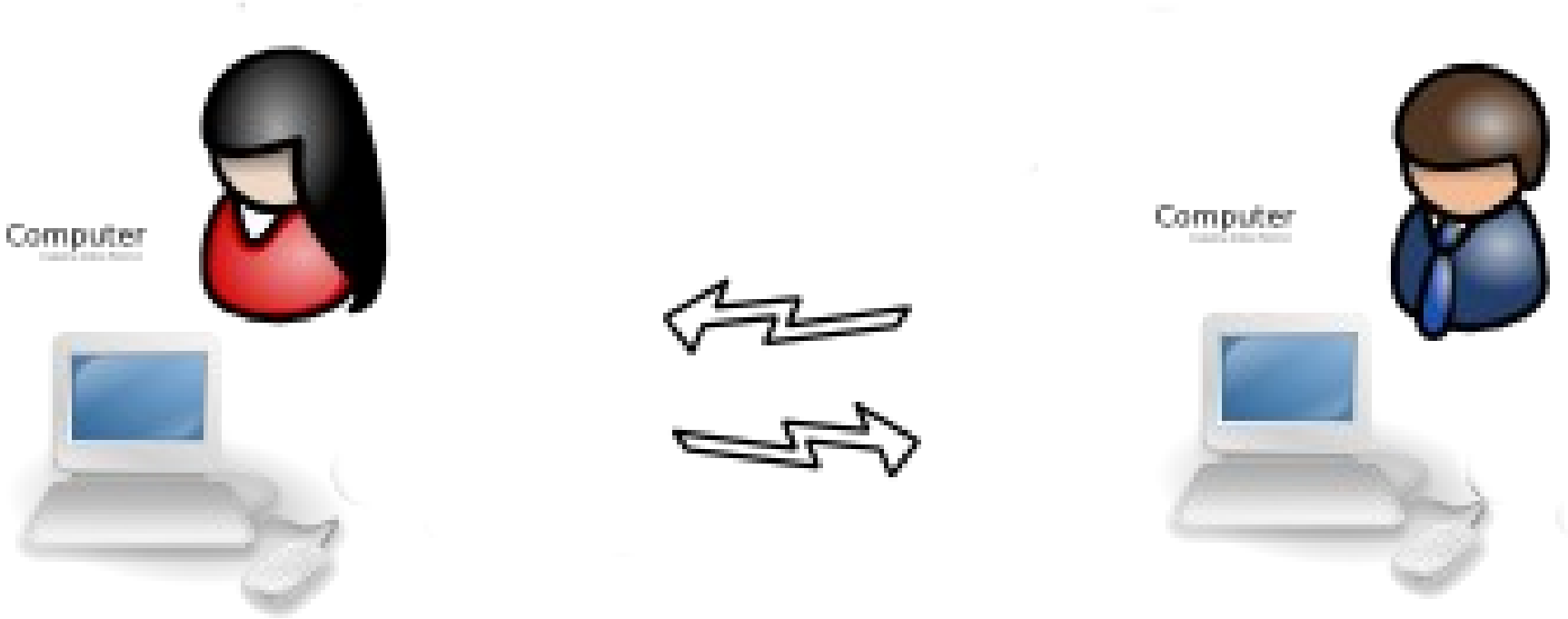


http://doc.bazaar-vcs.org/bzr.dev/en/user-guide/index.html

Le modèle distribué

- Le modèle distribué est plutôt récent et encore en cours de développement
- Il s'inspire du « peer-to-peer » (BitTorrent, etc.)
  - Chaque client possède son propre dépôt et ses proches branches
  - Un client peut aller chercher une branche dans un dépôt distant
  - Un client peut fusionner une branche distante avec les siennes
  - Un client peut pousser une branche locale vers un dépôt distant
  - Un client peut fusionner une branche locale dans une branche distante
- On peut facilement émuler le modèle centralisé avec ce modèle
- Algorithmes de fusion avancés

Le modèle distribué



http://doc.bazaar-vcs.org/bzr.dev/en/user-guide/index.html

Systèmes distribués

Bazaar

http://bazaar-vcs.org/

git

http://git.or.cz/

Mercurial

http://www.selenic.com/mercurial/wiki/

Partie VIII

Références

Références

Subversion

Version Control with Subversion

COLLINS-SUSSMAN, Ben
W. FITZPATRICK, Brian
PILATO, C. Michael.

http://svnbook.red-bean.com/

Subversion

Introduction à Subversion:
principes et conseils

LEGENDRE, Georges-Étienne
BOURBONNAIS, Félix-Antoine

http://legege.com/fr/documents/

Références

- ***Subversion (Site officiel)***
  http://subversion.tigris.org/

- **Using Subversion with KDE** (Tutoriel pour les développeurs)
  http://developer.kde.org/documentation/tutorials/subversion/

- **How to organize a Subversion Repository**
  http://docs.codehaus.org/display/HAUS/How+to+Organize+a+Subversion+Repository

- ***Subversion at the haus***
  KAZMIER Pete et MELAMED Zohar.
  http://docs.codehaus.org/display/HAUS/Subversion+at+the+haus

Références

- http://subversion.tigris.org/
- http://www.gentoo.org/doc/fr/cvs-tutorial.xml
- http://subclipse.tigris.org/
- http://rapidsvn.tigris.org/
- http://subcommander.tigris.org/
- http://tortoisesvn.tigris.org/
- http://www.alwins-world.de/programs/kdesvn/
- http://www.cvshome.org/
- http://www.tortoisecvs.org/
- http://www.lincvs.org/
- http://www.kde.org/apps/cervisia/

Présentation en ligne

Cette présentation est disponible en ligne:

http://www.rubico.info/docs/prog-subversion/