第 1 回 UNIX ゼミ　Subversion マニュアル

ゼミ担当者　：　木田 清香，千田 智治，村上 耕平
指導院生　：　梶原 広輝，柴田 優，山川 望
開催日　：　2005 年 4 月 17 日

1　はじめに

大規模なシステムの開発を行うプロジェクトは，複数の開発者で進められることが多く，開発中のシステムが大規模であるほど，バージョン管理が重要となる．例えば，オープンソースプロジェクトや企業でのソフトウェア開発などでは，開発中のプログラムなどの改良やバグの修正をバージョンにより管理し，プロジェクトが進められている．もちろん，個人で開発するソフトウェアなどに関しても，改良の履歴をバージョンによって管理することは非常に有用である．そのため，情報系の学生にとってバージョン管理は必須のスキルである．そこで，本ゼミではバージョン管理に関する基礎知識，バージョン管理システム”Subversion”を利用したリポジトリの管理方法，及びワーキングコピーでの操作などの基本事項について学ぶ．

2　バージョン管理システム

2.1　バージョン管理とは

ソフトウェアやシステムの開発は，おもにソースコードの修正を繰返し，改良をしていくことが活動の基本となる．ただ，開発現場ではソースコードやドキュメントに修正を加えるのが一人だけとは限らず，複数の開発者が並行して開発を進める場合も存在する．このため，ソースコードへの修正をやみくもに加えていくだけでは,「いつ」,「どこを」,「どう」修正したのかわからなくなってしまう．そこで，ファイルを修正する際には，修正前の状態を記録し，修正後に差分を取ることで，ファイルの更新を記録する方法が考え出された．この方法を用いることで，プロジェクトの進行を管理することが出来る．バージョン管理システムは基本的にこの方針に基づいて作られている．

2.2　Subversion とは

Subversion とは，ファイルのバージョン管理をするアプリケーションソフトである．Subversion は主にプログラムの開発現場などで使用されるが，Subversion 自体はプログラムをはじめ，どんなファイルでも管理できる．複数人が同時に同じファイルを編集することができ，編集した内容が競合 (コンフリクト) していなければ両方の変更を自動的に統合できる．

3　基本用語

ここではバージョン管理システムで用いられる基本用語について簡単に説明する．

- バージョン (version)

  ファイルに対して変更を加え，それをバージョン管理システムに登録するごとにそのファイルのバージョンと呼ばれる番号が上がっていく．また，バージョンのことをリビジョン (revision) ともいう．

- リポジトリ (repository)

  バージョン管理システムでは，ファイル群は一箇所にまとめられて管理される．このファイルをまとめたものをリポジトリという．この中にファイルの変更履歴 (version history) も記録される．

- ワーキングコピー (working copy)

  リポジトリにあるファイル群を編集するためコピーしたもので，ファイル群のワーキングコピー内での変更履歴も，同じディレクトリ内で保存されている．なお，同じディレクトリを複数のリポジトリのワーキングコピーとすることはできない．

- チェックアウト (checkout)

  リポジトリからワーキングコピーを取り出すことをチェックアウトという．

- コミット (commit)

  ワーキングコピーでの更新結果を確定し，リポジトリに変更を反映させることをコミットという．

- アップデート (update)

  あるワーキングコピーでは編集してなくても，他の開発者のワーキングコピーからのコミットにより，リポジトリが更新されている可能性がある．リポジトリの更新状況をワーキングコピーに反映させることをアップデートという．

- コンフリクト (conflict)

  ファイルの変更をコミットする際に，手元で行った修正と他からコミットされた修正が，同じ箇所にお

いて異なった修正である場合，コンフリクトしているという．コンフリクトが検出された際，ユーザはワーキングコピーを編集しコンフリクトを解消する必要がある．

- インポート (import)

  最初にリポジトリにファイル群を登録することをインポートという．

- エクスポート (export)

  編集するつもりはなく，管理ファイルを除いて対象ファイルだけを取り出す操作のことをエクスポートという．ソフトウェアのリリースなどはこの一例である．

4 Subversion コマンド

Subversion を用いてバージョン管理する際に用いるコマンドを紹介する．

- svnadmin create

  カレントディレクトリにリポジトリを作成して初期化する．

  ```
  svnadmin create [リポジトリ名]
  ```

  Subversion を利用する際，このコマンドで，あらかじめリポジトリを作成しておく必要がある．

- import

  バージョン管理するファイル群をリポジトリに登録する．

  ```
  svn import -m "コメント" URL[インポート先]
  ```

  カレントディレクトリにあるファイルが全て，リポジトリに登録される．

- checkout

  ワーキングコピーを取り出す．

  ```
  svn checkout URL[チェックアウト元] ディレクトリ名
  ```

  リポジトリから指定した名前のディレクトリの中に，編集用にファイル群を取り出す．ディレクトリ名を指定しなければ，リポジトリと同じ名前のディレクトリが作成される．なお,「checkout」は「co」と略されたコマンドでも同様の動作となる．

- commit

  ワーキングコピーでの編集をリポジトリへ反映させる．

  ```
  svn commit -m "コメント"
  ```

- add

  Subversion の管理下にファイルを追加する．

  ```
  svn add [ファイル名]
  ```

  コミットするまでリポジトリには反映されない．

- delete

  ファイルをリポジトリから削除する．

  ```
  svn delete ファイル (ディレクトリ)
  ```

  コミットするまでリポジトリには反映されない．

- move

  「move」には 2 通り使い方があり，リポジトリに対して UNIX のコマンドの「mv」と同じ操作が可能である．

  ファイル (ディレクトリ) 名の変更を行う．

  ```
  svn move 古いファイル名　新しいファイル名
  ```

  ファイル (ディレクトリ) の移動

  ```
  svn move ファイル名　移動先
  ```

  コミットするまでリポジトリには反映されない．

- status

  ワーキングコピーにあるファイルの更新状態を表示させる．

  ```
  svn status
  ```

  行のはじめに表示される大文字英字がファイルの状態を示す (Table 1)．ワーキングコピーが変更されていない場合，何も表示されない．

Table 1　記号の表す状態

記号	ファイルの状態
?	Subversion の管理下にない
A	add してまだコミットされていない
M	修正部分がマージ出来た
D	ファイルが削除されている
C	コンフリクトしている

「-v」オプションをつけることでより詳細を表示することができる．

```
svn status -v
```

左から順に，ファイルの状態，ワーキングコピーのバージョン名，最後にコミットしたバージョン，コミットしたユーザ名，ファイル名が表示される．

- export
  ファイルを編集用ではなく取り出す．

```
svn export URL[エクスポート先]
```

チェックアウトと違いエクスポートしたデータは，編集用ではないのでバージョン管理用のファイルを含まない．

- diff
  ワーキングコピー内の変更された点を参照する．

```
svn diff
```

ファイルに編集前後の削除および追加部分を知ることが出来る．

- log
  指定したファイルの作業履歴を調べる．

```
svn log ファイル名
```

変更のあった際のリビジョン番号，変更したアカウント名，変更時刻，変更操作，が表示される．

- revert
  ワーキングコピーでの修正を取り消す．

```
svn revert ファイル名
```

コミットする前の段階で，ワーキングコピーのファイルを修正前の状態に戻すことが出来る．

- mkdir
  リポジトリ内にディレクトリを作成する．

```
svn mkdir ディレクトリ名
```

コミットするまでリポジトリには反映されない．

5　チュートリアル

Subversion の利用方法を，kouhei ユーザの場合を例に順に示す．

まず，mikilab にログインし，各自のリポジトリ作成用のディレクトリ内 (/home/svn/personal/［アカウント名］) でリポジトリを作成する．

```
local:~$ ssh kouhei@mikilab.doshisha.ac.jp
kouhei@mikilab:~$ cd /home/svn/personal/kouhei/
kouhei@mikilab:/home/svn/personal/kouhei$ svnadmin create repos
kouhei@mikilab:/home/svn/personal/kouhei$ ls
repos
kouhei@mikilab:/home/svn/personal/kouhei/repos$ logout
```

次に，リポジトリにインポートするファイルを用意する (今回は echo コマンドで簡単なテキストファイルを作成する)．ファイル作成後，mikilab に作成したリポジトリにファイルをインポートする．

```
local:~$ echo "ahaha" > ahaha.txt
local:~$ svn import svn+ssh://kouhei@mikilab.doshisha.ac.jp/home/svn/personal/kouhei/repos/
¥
> -m "initial import"
Enter passphrase for key '/home/kouhei/.ssh/id_dsa':
Adding ahaha.txt

Committed revision 1.
```

これでリポジトリにファイルが登録される．

リポジトリにインポートしたファイルをチェックアウトして，ワーキングコピーを取り出す．ここでは，work というディレクトリを作成し，それをワーキングコピーとする．

```
local:~$ mkdir work
local:~$ svn checkout ¥
> svn+ssh://kouhei@mikilab.doshisha.ac.jp/home/svn/personal/kouhei/repos work
Enter passphrase for key '/home/kouhei/.ssh/id_dsa':
A ahaha.txt
Checked out revision 1.
local:~$ ls
ahaha.txt work
local:~$ cd work
work $ ls
ahaha.txt
```

ワーキングコピー内にインポートされたファイルがあるか確認する．

ワーキングコピー内に新たにファイルを作成する．このとき，新しいファイルは Subversion の管理下にないため，add コマンドにより Subversion の管理下に追加する．追加後，コミットしてその追加を確定する．

```
work $ echo "pupupu" > pupupu.txt
work $ svn status
?  pupupu.txt
work $ svn add pupupu.txt
A pupupu.txt
work $ svn commit -m 'initial add'
Enter passphrase for key '/home/kouhei/.ssh/id_dsa':
Adding pupupu.txt
Transmitting file data .
Committed revision 2.
```

ワーキングコピー内のファイルに修正を加え，ワーキングコピーの状態を確認する．ファイルの状態が「M」と表示され，ファイルに修正が加えられているのが確認できる．
その後，コミットして，リポジトリへ変更を確定する．

```
work $ svn status
work $ echo "pepepe" >> ahaha.txt
work $ svn status
M ahaha.txt
work $ svn commit -m 'initial ahaha.txt'
Enter passphrase for key '/home/kouhei/.ssh/id_dsa':
Sending ahaha.txt
Transmitting file data .
Committed revision 3.
```

これで，ファイルへの変更が確定され，リポジトリに反映される．

ワーキングコピー内でファイルを修正しその差分を確認する．

```
work $ echo "popopo" >> ahaha.txt
work $ svn diff
Index:  ahaha.txt
===================================
--- ahaha.txt (revision 3)
+++ ahaha.txt (working copy)
 -1,2 +1,3
ahaha
pepepe
+popopo
```

ahaha.txt のファイルの中に,「popopo」が追加されたことが分かる．

ファイルの修正を取り消す.

```
work $ svn status
M ahaha.txt
work $ svn revert ahaha.txt
Reverted 'ahaha.txt'
work $ svn status
work $
```

これでファイルが修正前の状態に戻る.

ファイル名の変更を行う.

```
work $ svn move ahaha.txt hogehoge.txt
A hogehoge.txt
D ahaha.txt
work $ svn commit -m 'rename ahaha hogehoge'
Enter passphrase for key '/home/kouhei/.ssh/id_dsa':
Deleting ahaha.txt
Adding hogehoge.txt

Committed revision 4.
```

リポジトリのファイル名が変更される.

ファイルの削除を行う．delete でワーキングコピー内から削除した後，その削除操作をコミットして確定する.

```
work $ svn delete pupupu.txt
D pupupu.txt
work $ svn commit -m 'delete pupupu'
Enter passphrase for key '/home/kouhei/.ssh/id_dsa':
Deleting pupupu.txt

Committed revision 5.
```

最後に，特定のファイルのこれまでの作業履歴を表示する.

```
work $ svn log hogehoge.txt
Enter passphrase for key '/home/kouhei/.ssh/id_dsa':
------------------------------------------
r4 | kouhei | 2006-04-16 15:46:08 +0900 (Sun, 16 Apr 2006) | 1 line

rename ahaha hogehoge
------------------------------------------
r3 | kouhei | 2006-04-16 15:34:57 +0900 (Sun, 16 Apr 2006) | 1 line

initial ahaha.txt
------------------------------------------
r1 | kouhei | 2006-04-16 15:12:05 +0900 (Sun, 16 Apr 2006) | 1 line

initial import
------------------------------------------
```