JIM (JLC detector Simulator) の開発と CVS を用いた program 管理

細田直康、東京都立大学

1994 年 8 月 4 日

概要

これは jlc cluster (jlcux1,jlcux2,jlcux3) で JIM program の開発を行なう人のための manual です。開発は simu group でのみ行なえます。質問、意見等は hosoda@jlcux1.kek.jp に送って下さい。

目次

1 Coding rule **2**
- 1.1 使用言語 . . . 2
- 1.2 Include 文 . . . 2
- 1.3 SAVE 文 . . . 2

2 CVS とは **2**
- 2.1 概念 . . . 2
- 2.2 基礎用語 . . . 2

3 CVS command **3**
- 3.1 環境設定 . . . 3
- 3.2 Source を取ってくる . . . 3
- 3.3 File を新たに作成し増やす . . . 3
- 3.4 File を削除する . . . 4
- 3.5 Work space での変更を CVS に登録する . . . 4
- 3.6 Repository の file で自分の file を更新する . . . 4
- 3.7 Version tag を付ける . . . 5
- 3.8 File の情報を得る . . . 5
- 3.9 全ての変更を終え local copy を消す . . . 6

4 CVS Repository **6**
- 4.1 Repository の設定 (CVS 管理者のみ) . . . 6
- 4.2 Directory 構造 . . . 7

1 Coding rule

1.1 使用言語

ANSI FORTRAN 77 を用いる。

1.2 Include 文

唯一 ANSI 規格でないものとして、 INCLUDE 文を用いる。 HP では以下のように記述する。

```
INCLUDE 'gcflag.h'
```

1.3 SAVE 文

COMMON 文を作成した時はその後に SAVE 文を加えておいて下さい。以下は例です。

```
COMMON /SAMPLE/ xxx,yyy,zzz
SAVE   /SAMPLE/
```

2 CVS とは

2.1 概念

CVS (Concurrent Versions System) は複数の人が同時に同じ program source を開発出来るようにする system である。

CVS は file を"repository" と呼ばれる場所に持っている。 User はいつでも自分の directory に repository から file を取ってきてこれを自由に変更できる。作業が終った後は自分の変更を repository に戻しておく。 File を取ってきた後でもし他の人が先に repository を変更した場合は、自分の変更を戻すときに CVS が教えてくれるので、 user は変更点を merge してから repository に戻す。 CVS ではそれぞれの開発段階に tag (version) をつけることができ、最新の version だけでなく、古い version を取ってくることもできる。

2.2 基礎用語

Repository: File の master copy のある directory。 CVS 管理者が作っておく。

Module: Main repository 中の directory。

RCS: Revision Control System。 CVS が従う下位の program。

Check out: Repository から file を取ってくること。

Revision: File の version を同定するための英数字による tag。

3 CVS command

以下は CVS repository がすでに作られていて、 program を開発しようとする人が主に用いる command です。ほとんどの command は user が check out した (する) directory 中で使われます。

CVS の online help は

```
% cvs -H
% cvs [command] -H
```

です。

3.1 環境設定

CVS を使う人は repository の位置を環境変数で指定しなければなりません。 csh/tcsh user は

```
setenv CVSROOT /proj/phys/master
```

と /.profile か /.login file に書いて下さい。また、

```
setenv EDITOR mule
```

で cvs commit で用いる editor を指定できます。この default は vi です。

3.2 Source を取ってくる

登録された module を取り出すには、 current directory を work space にしてから、

```
% cvs checkout jim
```

とします。これにより current directory に jim directory が作成され、その中の directory や file も取り出されます。この状態で、自由に内容が変更できます。ただし、 jim directory 中に CVS directory が、さらに他の directory 中にもそれぞれ CVS directory が作られますが、これは CVS の管理する directory なので変更しないでください。

3.3 File を新たに作成し増やす

新たに file を増やすには、その file を作成した後で、

```
% cvs add file-name
```

としてください。

3.4 File を削除する

すでにある file を削除するには、その file を rm した後で、

```
% cvs remove file-name
```

としてください。

File 名の変更は cvs remove/cvs add の組合せで行ないます。ただし削除や名前変更は他に check out している人との間で混乱を起こすかも知れないので注意して行なって下さい。

3.5 Work space での変更を CVS に登録する

File の編集などが終って自分の work space で変更した内容を CVS に伝えるには、jim directory の上 (check out をした directory) に行き、

```
% cvs commit
```

とします。これにより current directory から下の部分を CVS に伝えます。ふつうは editor が起動し、変更の log を入力すれば、 CVS 管理下の file も更新されます。

なるべく repository に入れたくない file(*.o,jim,jim++ etc.) は rm しておいた方がいいでしょう。 Default では以下の file は無視されます。

```
RCSLOG  RCS  SCCS  CVS*  clslog.*  tags  TAGS
.make.state  .nse_depinfo  *~  #*  .#*  ,*
*.old  *.bak  *.BAK  *.orig  *.rej  .del-*
*.a  *.o  *.so  *.Z  *.elc  *.ln
```

通常は commit を行なえば変更が登録されますが、 check out してから commit するまでの間に他の user が同じ file を commit している場合には、その旨を示して登録を行ないません。 CVS では file を自由に変更できるためこのような事態が起こり得ます。このときは次の update command を用います。

3.6 Repository の file で自分の file を更新する

他の人によって先に repository が更新されたときはその変更を組み合わせねばなりません。これは、

```
% cvs update
```

で半自動的に行なわれます。 File がどのように組み合わされたかは、 file 名の前に記号で示されます。

U: repository に変更は無かった。

M: 変更はあったが問題なく merge できた。

C: 変更が重なっていた。

結果が M のときは user が再度その file を check する必要があります。 C のときは <<<<< や >>>>> で異なる行が示された file ができるので、これを再編集しなければなりません。いずれも間違いがないようにしてください。

3.7 Version tag を付ける

Check out して変更した program に例えば jim1_5 という version tag をつけることができます。これは jim directory の上 (check out をした directory) に行き、以下を行ないます。

```
% cvs tag jim1_5 jim
```

これは CVS の特徴の 1 つです。

のちに古い version tag の module を取り出すには、例えば以下のようにして下さい。

```
% cvs checkout -r jim1_5 jim
```

File の持っている全ての tag を見るには、

```
% cvs status -v
```

としてください。

3.8 File の情報を得る

File の commit 時の message や誰が変更したかを見るには、

```
% cvs log [file-name]
```

File の revision 間での違いや version tag 間での違いは、

```
% cvs diff -r 1.1 -r 1.2 [file-name]
% cvs diff -r jim0_5 -r jim1_5
```

などで見ることが出来ます。

CVS repository の情報は、

```
% cvs history -a -e
```

などで見られます。

3.9 全ての変更を終え local copy を消す

全ての変更を終えたら、 check out した directory,file を消すことが出来ます。まず、 jim directory の上 (check out をした directory) に行き、

```
% cvs release -d jim
```

で削除されます。もし変更されたけれど、 commit や add/remove などがされていない file があると、それが示されます。その時は"n" で release を中止して、上記を行なってから再度 release してください。示された file がいらない file であるなら"y" で終了してください。

4 CVS Repository

4.1 Repository の設定 (CVS 管理者のみ)

Repository のなかには CVS の基本的な管理 file が必要なので、誰もが自由に作成するのでなく、管理者を決めて取り扱うべきです。今回は細田が行ないました。

1. 環境変数で root を指定する。

   ```
   % setenv CVSROOT /proj/phys/master
   ```

2. CVS source directory (CVS を解凍してできたもの, /home/hosoda/cvs/cvs-1.3) に行き、

   ```
   % ./cvsinit
   ```

   とする。この directory 中の他の file が必要なのである。これで /proj/phys/master/CVSROOT directory が作られ、必要な file が用意される。

3. 自分の working directory (/home/hosoda/cvs/work) に行き、

   ```
   % cvs checkout modules
   ```

   とする。 /home/hosoda/cvs/work/modules directory が作られ、中に CVS directory, modules file が出来る。

4. Editor を起動し、これから加える module (directory) を中に書く。

   ```
   % cd modules
   % mule modules
   ```

   例えば、以下を書く。

```
# Add other modules here...
jim             jim
src             jim/src
doc             jim/doc
```

5. 変更した内容を CVS に伝える。

```
% cvs commit
```

6. modules directory を消す。

```
% cd ..
% cvs release -d modules
```

7. 自分の working directory に CVS に入れたい directory を用意する。

```
% mkdir jim
% cd jim
% mkdir src
% mkdir doc
```

さらにその中に file を用意する。

8. 新しく module を登録する。 jim directory(/home/hosoda/cvs/work/jim) に行き、

```
% cvs import -m '1st setting of jim' jim jim jim0_5
```

9. 以上で jim directory 以下は登録されたので、 working directory は普通に rm,rmdir で消してかまいません。

10. Group に属している人が使用できるように permission を変更しておきます。まず CVSROOT 以下の全ての directory permission を 775 にします。次に CVSROOT/CVSROOT directory に行き、 history,commitlog file を 664 にします。

4.2 Directory 構造

```
/proj/phys/master --- jim -+- src
                           +- doc
```