Using the Revision Control System

Last revised: 09-24-96

The Revision Control System, or RCS, is a system for managing revisions of files. According to the reference page:

> RCS automates the storing, retrieval, logging, identification, and merging of revisions. RCS is useful for text that is revised frequently, for example programs, documentation, graphics, papers, and form letters.

For SSCC users, RCS is an excellent system for managing changes in SAS, SPSS, Limdep, and Stata code. RCS can:

- Store and retrieve multiple revisions of text.
- Maintain a complete history of changes.
- Automatically identify each revision with name, revision number,creation time, author, etc.
- Minimize secondary storage.

How RCS Works

The RCS system stores your latest revision of a file in an archive with the extension ",v". For instance, if you have a file called "srccode.sas", the RCS archive will be called "srccode.sas,v". This archive contains a copy of the latest version of your file, contents of the previous versions, and information about the various versions.

Preparing to use RCS

No preliminary work is needed to use RCS. However, if you want to keep your RCS archives in a different directory than your other work, you may create a subdirectory called RCS:

```
mkdir RCS
```

Note that the name RCS is capitalized. If this directory exists, all RCS archives will be stored in that directory, instead of the current directory. Note that you need one RCS directory for every directory in which you have source code.

The Initial Version of a File

When you first create a file that you want to maintain with RCS, you need to check it in to an archive. The command to check in a version, whether an initial version or a later one is **`ci`**, which stands for "check in". The general syntax is:

```
ci filename
```

For example, if you have a file called "srccode.sas", you would execute:

```
ci srccode.sas
```

The **ci** command will prompt you for a description of the file. Enter information and then type a single period (".") on a line and press Return. An RCS archive will be created and your original file will be removed.

Viewing a File

To view a file in an RCS archive without modifying it, use the command **co**, which stands for "check out". The general syntax is:

```
co filename
```

In our example, you would type:

```
co srccode.sas
```

This will check out the latest version of the file. You can only look at the file; you cannot edit it. When you are done viewing the file, you may remove it.

Modifying a File

To modify a file, you must check it out, lock the archive, edit the file, and then check it back in. To check out a file locked, use the **-l** option to **co**. For example:

```
co -l srccode.sas
```


This will create a file that you can edit. When you have completed your edits, check the file back in:

```
ci srccode.sas
```

Now, **ci** will prompt you for a log message. Enter an explanation of your changes. This is for your benefit, and so you may be as detailed or vague as you desire. Then, you may check out the file for viewing or editing as many times are you wish.

Listing Revisions

To list the revisions, use the **rlog** command. The general syntax is:

```
rlog filename
```

or in our example:

```
rlog srccode.sas
```

The **rlog** command will list the revisions, with revision number, date of revision, author, and log messages.

Retrieving a Previous Revision

Use the **co** command to retrieve a previous revision. The general syntax is:

```
co -r<rev> filename
```

For example, if you wanted to retrieve revision 1.2 of srccode.sas, execute:

```
co -r1.2 srccode.sas
```

This revision will be checked out.

Comparing Revisions

The rcsdiff command can be used to compare revisions. To campare a revision with a currently checked out version of a file, use the following syntax:

```
rcsdiff  -r<rev> filename
```

In our example, this would be:

```
rcsdiff  -r1.2 srccode.sas
```

This will display the differences between the two files. The rcsdiff command works on a line by line basis and shows which lines are in one file, and which in another. For instance, in the example below, line 2 is different between the file versions. In one version, the file being used is called “gss94.Z”, and in the other, it is called “gss95.Z”:

```
norman.ssc.wisc.edu> rcsdiff  -r1.2 srccode.sas
2c2
< FILENAME GSS PIPE 'zcat gss94.Z' ;
---
> FILENAME GSS PIPE 'zcat gss95.Z' ;
norman.ssc.wisc.edu>
```


You can also compare differences between revisions that are already checked in. The general syntax is:

```
rcsdiff -r<rev1> -r<rev2>
```

In our example, a command would be:

```
rcsdiff -r1.2 -r1.3 srccode.sas
```

This would display the differences between the 1.2 and 1.3 versions of the file. Any version numbers can be used, of course.

Using the RCS administrative Command

Occasionally, it may be necessary to lock and RCS archive so that you may add a revision to an archive, or unlock an archive, because you no longer intend to make a revision. In this case, you can use the **`rcs`** command lock or unlock a file. The general syntax for locking a file is:

```
rcs -l filename
```

The syntax for unlocking a file is:

```
rcs -u filename
```

With the **rcs** command, you can do many other administrative tasks. See the **rcs** reference page for additional information.

Learning More

The reference page for rcsintro(1) gives more details on the use of RCS commands. View this file by executing the command:

```
man rcsintro
```

For details on particular commands, look at reference pages for:

ci(1)
co(1)
rcs(1)

Other reference pages of interest are:

ident(1)
rcsdiff(1)
rcsintro(1)
rcsmerge(1)