Astronomical Data Analysis Software and Systems VI
ASP Conference Series, Vol. 125, 1997
Gareth Hunt and H. E. Payne, eds.

An ExpectTk/Perl Graphical User Interface to the Revision Control System (RCS)

R. L. Williamson II

Space Telescope Science Institute, Baltimore, MD 21218,
E-mail: ramon@stsci.edu

Abstract. In order to track changes in the TABLES/STSDAS code, as well as for Quality Assurance and to simplify the code release process, the GNU Revision Control System (RCS) was chosen as the method of revision control. Due to special needs of our group and the desire to have an easy-to-use interface to the RCS system that could be used not only for TABLES/STSDAS but for personal code as well, a suite of Perl scripts and an Expect/Tk graphical user interface was written. This user interface will be described as well as the functionality of the Perl scripts.

1. Introduction

Due to the size and complexity of the TABLES/STSDAS system, it was decided to place the packages under Revision Control. A survey of the Revision Control software available led to the choice of the GNU Revision Control System (RCS) as the method of revision control. This choice has many advantages:

- Free,
- Ease of Use,
- Secure, but not rigidly so,
- Tracks versions of code by version number or any number of logical version names, and
- Version numbers can be placed in comments in the code and executables, either manually or automatically by RCS so that the pedigree of executables can be determined.

Of course, as with many choices there are disadvantages:

- Only works at most with one directory at a time—doesn't do trees,
- Wild cards will do all files—including files like libraries and executables that are not under revision control,
- No mechanism for removing files from a distribution, yet leave the file in the system for checking out older versions of the packages, and
- Can be difficult to use if the source directory and the working directory are different.

Solution: A set of Perl scripts that enable directory tree check in and out, excluding some files and placing and keeping track of links, as well as retiring files. For ease of use, a User Interface is also needed.

2. Perl Scripts

Three Perl scripts control most of the actual checking in, checking out, and retiring of files.

2.1. checkin

This script is used to initiate the revision control and update changes to the code in the revision control system. Object files, libraries, executables, and directories are excluded from the checkin, and unless told otherwise will descend trees, checking in all files found there.

2.2. checkout

This script is used to check files out of the RCS and places them in a working directory to be examined and edited. Unless told otherwise, checkout will also descend trees, checking out all files found there.

2.3. doretire

This script takes the path to a file to be retired, and removes it from the distribution, while retaining it in a Crypt directory for access by checkin and checkout. Checkin and checkout automatically look for this Crypt directory and check the versions of the files there to determine whether they should be checked out as well.

3. Expect/Tk Graphical User Interface

The part of the system that the user sees is the Graphical User Interface. This code was written in Expect/Tk in order to take advantage of the ability to have an interaction window where the output from the RCS commands are shown, but also the user can reply to queries by the code.

This GUI connects to the checkin, checkout, and doretire commands through the action buttons in the middle of the GUI. Here each widget is described with respect to the number labels in Figure 1.

1. Package Radiobuttons

   These radiobuttons are used by Check Out and Do a mkpkg to determine the name of the package to act on. Only one option can be chosen at a time. These buttons are turned off when Check In is chosen, since they are not used by this task.

2. System Directory Radiobuttons

   These radiobuttons toggle which major branch of the chosen package is checked out or mkpkg done on. For STSDAS or TABLES for example, you can choose top, lib, or pkg. “Top” refers to the top-level directory.

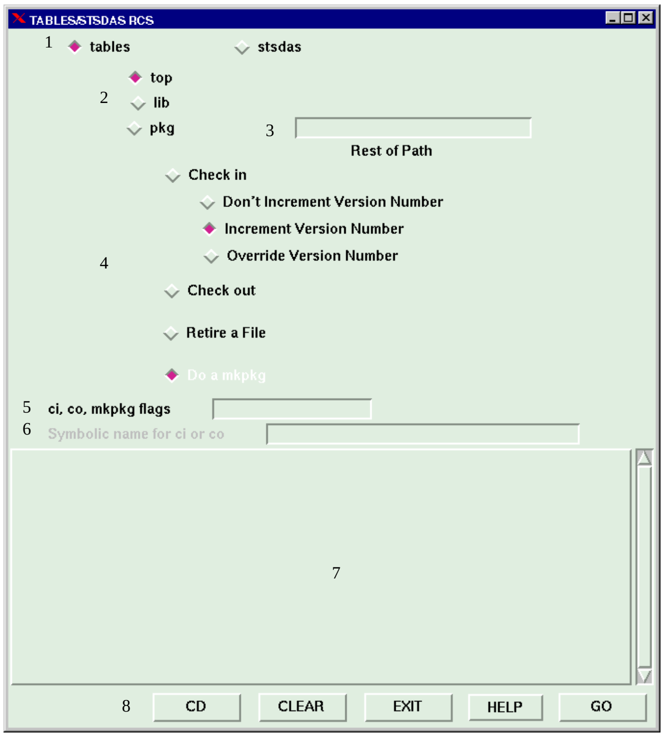

Figure 1. RCS Graphical User Interface.

Only one can be chosen. These buttons are turned off when Check In is chosen, since they are not used by the task.

3. Rest of Path Entry Widget

   This entry widget is for typing in the path to the package desired past the lib, pkg, etc. For example, if you would like to checkout the Synphot package, found in stsdas$pkg/hst_calib/synphot, you would type "hst_calib/synphot" in this widget after setting the Package and System Directory radiobuttons for stsdas and pkg. This widget is deactivated when Check In is chosen, since it is not used by this task.

4. Action Radiobuttons

   These radiobuttons toggle which action to take on the files. You can check filesinto the RCS system, Check files out of the RCS system, Retire a file or do a mkpkg within the configured directories for the packages. Only one can be chosen.

5. ci, co, mkpkg flags Entry Widget

   This entry widget is for typing any flags or arguments that might be recognized by RCS's ci or co, or IRAF's mkpkg. This widget parses the entry. Say, for example, you want to use mkpkg and the flag you want to use is -p tables and an argument is update. Then in the widget type `-p tables update`. The widget parses this and sends the parameters to mkpkg correctly. One flag added to this GUI is the only flag. Used in coordination with checkin or checkout, this flag tells the task to only check in or out the directory chosen, no subdirectories. This is especially useful when dealing with files in the top-level directory.

6. Symbolic name for ci or co Entry Widget

   This entry widget is for typing in the symbolic name for the code to be checked in or out. Multiple values can be placed in the widget space delimited for assigning multiple symbolic names to checked in code. This widget parses the Symbolic names and adds "-n" to the name for use by ci or co. Information in this widget are not used by mkpkg, so this widget is deactivated when the "Do a mkpkg" radiobutton is chosen.

7. Interaction Widget

   This widget is similar to a console for the tasks spawned from this GUI. All output from the tasks are sent to this widget, and any prompts for input will go here as well. To reply to the task, simple place the mouse cursor in this widget and type your reply.

8. Control Buttons

   These buttons control many of the aspects of the GUI.
   CD—Changes the working directory.
   CLEAR—Clears all entry widgets, and resets all buttons to the default values.
   EXIT—Kills all processes spawned from the GUI and exits the GUI.
   HELP—On-line help for the widgets.
   GO—Initiates the chosen action.

4. Personal Packages

The STSDAS RCS system can be customized using two configuration files which control the colors of the GUI and the packages under revision control. Almost any number of packages can be under configuration by adding the packages to these configuration files.

5. Availability

The STSDAS RCS system may be obtained by contacting the author at ramon@stsci.edu. Included will be instructions for installation as well as a user's guide.