SourceForge, Hamlib, and Rigserve: Free Beer, Free Speech, and Rig Control

Martin Ewing, AA6E

Abstract

Hamlib is a software library to support ham radio rig control begun in 2000, supporting many radio types and computer platforms. Rigserve is a new network server approach to rig control, inspired by Hamlib and based on Python classes. Both projects are managed at SourceForge.net. There are many advantages to developing Open Source Software in this open project environment.

KeyWords

rig control, open source, software development

Introduction

Amateur Radio software development and products have followed various development paths, from "amateur" and informal to fully commercial and professional. In this paper we describe two projects for rig control, and we also show how an Open Source Software (OSS) development approach has worked for them.

Ham culture (at its best) is about mentoring, helping, and sharing. As amateurs, we are happy to use commercial products when available, but many of us would prefer our software to be as widely used as possible rather than to seek a financial return. Even if a particular project does not take off, sharing ideas will support new advances elsewhere.

About Open Source Development

You have a great idea for a new Amateur Radio software application and you want to do this in the Free and Open Source Software style – making your code available at no charge and freely adaptable by other developers. You want to attract other programmers to help develop your code, and you want to attract users once the project is ready for beta testing or production. You need a revision control system to keep track of program versions and to combine the work of multiple developers.

Any Linux computer has most of the tools you need in principle, but in practice many projects choose to use an open source project hosting service. There are several such services, but SourceForge.net [1] is probably the largest and most popular. At no charge, a project on SourceForge.net receives Content Versioning System (CVS) or Subversion (SVN) source repository accounts, user and developer support forums and mailing lists, and a file repository for formally released project versions. In addition, there are facilities for bug reporting and tracking along with project-specific web and wiki hosting. SourceForge.net claims over 100,000 projects and 1,000,000 registered users. There are 219 projects identified "ham radio".

The Hamlib Project

Hamlib [2] is an example of a mid-scale OSS project.

The Hamlib project is an open source programming library for Amateur Radio rig control. Begun in 2000 by Frank Singleton (VK3FCS/KM5WS) and Stéphane Fillod (F8CFE), Hamlib now provides at least alpha support for 140 types of receivers, transceivers, and even some antenna rotators. The library is coded in standard C, but bindings are provided for C++, Perl, Python, and TCL. The source distribution now has some 188,000 lines of code in 767 files. There are 31 current developers (with CVS privileges) and a number of testers who contribute patches.

As a programming library, Hamlib targets developers, not end users. Having a convienient way to interface your new ham application to hundreds of rigs saves a lot of time and expands the potential user base for your programming efforts. Hamlib has been adopted by Dream, fldigi, gMFSK, Gpredict, grig, Xlog, PSKmail, and other projects. Hamlib is developed under Linux, but can operate under Windows and other operating systems.

Figure 1 shows how Hamlib works. It is a library linked with your application program. A "backend" handles the interface to your particular rig. Hamlib's main role is to be a subroutine library that will be linked in with a developer's program, but it also has a server personality. It is able to respond to Sun RPC calls provided across a network or on the local machine.

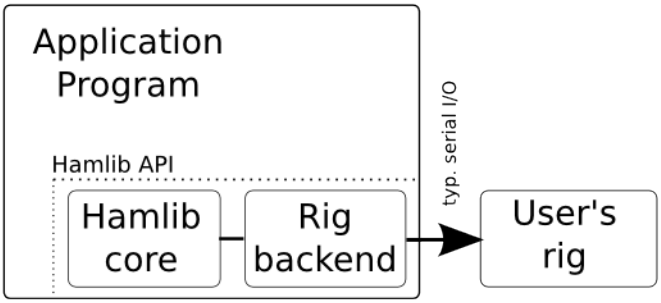

Figure 1. Hamlib Scheme

Hamlib is provided under the GNU "Lesser General Public License"[3] (LGPL), which makes it possible for commercial developers to use the library with proprietary programs. Hamlib's current distribution is version 1.2.6.

As a 7 year old effort, Hamlib faces problems that are common in mature OSS projects. Developers come and go as interests change, and early leaders have moved on. The code, after passing through the hands of dozens of programmers, has suffered feature creep and other technical issues that are stimulating design ideas for a potential version 2 release.

There are a few issues to highlight. First is the ongoing question of whether a general rig control solution should present the programmer with a simple, lowest common denominator interface that provides the basic functions used in the majority of control applications: read/set frequency, operating mode, filter settings, etc. Or, should we try to express the complicated options of the latest rigs?

Another question is how to fulfill the promise of wide interoperability among languages and platforms. This is a difficult problem for a library that must be linked with user software. Hamlib relies on SWIG [4], which wraps the C library calls for the Perl, Python, and TCL languages, with mixed success.

Finally, any control software that aspires to cover a wide range of rigs has a problem: how to code, test, and maintain rig drivers when no single programmer has access to very many live units to test? The project relies on back-end developers to provide code for the specific rigs they own themselves. This puts a premium on providing a simple and easy to program interface for these developers, who may be amateurs in software development – as well as radio.

The Rigserve Project

Hamlib's classic C language approach is efficient, elegant, and familiar – but rather baroque. In our opinion, C's low-level syntax works against clarity and ease of development. In addition, Hamlib has some internal architectural limitations – a 32-wide bit field that used to be generous, but now we are running out of bits for internal flags. Header files, preprocessor macros, elaborate typing, the need to compile in many environments have led to a structure that is not easy for new developers to master. (Fortunately, these problems are not very apparent to Hamlib *users*, who are the developers of new applications.)

Rigserve [5] is a new project begun as a demonstration of an approach to a second version of Hamlib, but there is no compatibility with Hamlib v1. Rigserve avoids the problems of library compilation and binding by running as a self-contained TCP/IP server using a straightforward human-readable protocol. Any programmer can communicate from any platform – anywhere (within security limits). See Figure 2.

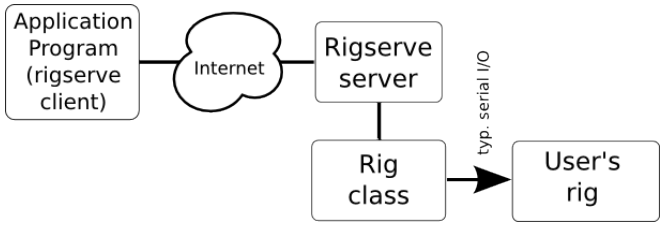

Figure 2. Rigserve Scheme

An equally important feature of Rigserve is that it is implemented in a high-level object-oriented programming language, Python. We prefer Python because of its rapid debugging, cross platform availability, its rich language and run-time environment, and its expressive, but straightforward class syntax. We value simplicity and clarity well above the somewhat slower execution of a "scripting language". Clarity is particularly important for hams who do not do software development every day.

Finally, the object oriented approach is a natural way of expressing the capabilities of amateur rigs, which come in families. For example, here is the current Rigserve class hierarchy for the Ten-Tec Orion:

object → Backend → Tentec → TT_orion →
{ TT_orion_v1 | TT_orion_v2 }

Backend is the fundamental rig object. It contains some status reporting and defines the range of standard methods.

Class *Tentec*, based on *Backend*, provides methods to work with the serial port. *TT_orion* describes the bulk of the Orion's features, while *TT_orion_v1* and *TT_orion_v2* express the small differences between the version 1 and version 2 firmware (e.g., S-meter calibrations).

A similar hierarchy will support Icom's CI-V interfaced rigs. Currently, we support the Icom R8500 and R75.

Rigserve's object approach easily allows for an "unlimited" number of rigs of the same or different types to be made available over a single IP port. Limits are imposed by the available I/O ports and memory, but not by the software architecture.

Rigserve command	*Comment*
open RIG1 TT_orion_v1	Create instance of a v1 Orion
put RIG1.CONTROL.init /dev/ttyS0 57600	Associate with serial port & baud rate
get RIG1.CONTROL.init	Initiate rig and return ID info
get RIG1.VFOA.freq	Get VFO A frequency
get RIG1.MAIN.rx_mode	Get Main receiver mode

Table 1. Sample Rigserve Commands

To allow for the widely different capabilities of rigs, Rigserve is developing a capability discovery mechanism that client applications can use to see which commands are supported. A more expressive (but less efficient) protocol like xmlrpc may be an alternative approach for Rigserve.

Rigserve now comprises about 3,500 lines of Python and is distributed under the GNU General Public License [6]. As a new project with only a few developers, there is a lot of room to grow, and we invite more participation.

Conclusion

Hamlib is an established software project that offers rig contol interfacing to many types of ham rigs for software application developers as a subroutine library. Rigserve is a new approach that implements rig control via a TCP/IP interface. More details are available on their project web sites.

These projects are examples of how open source software can be developed with a collaborative approach using a comprehensive project hosting service like SourceForge.net, which offers many services that greatly facilitate your project, including version control servers, forums, web services, and archives.

References

[1] SourceForge, Inc., *SourceForge.net Home Page*, http://www.sourceforge.net.

[2] Hamlib Project, *Ham Radio Control Libraries*, http://www.hamlib.org.

[3] Free Software Foundation, *GNU Lesser General Public License*, http://www.gnu.org/licenses/lgpl.html.

[4] SWIG Project, *Simplified Wrapper and Interface Generator*, http://www.swig.org.

[5] Rigserve Project, *Rigserve Website*, http://rigserve.sourceforge.net.

[6] Free Software Foundation, *GNU General Public License*, http://www.gnu.org/licenses/gpl.html.