33rd Space Symposium, Technical Track, Colorado Springs, Colorado, United States of America
Presented on April 3, 2017

OPEN SOURCING THE BALL AEROSPACE COSMOS COMMAND AND CONTROL SYSTEM

Ryan Melton
Ball Aerospace, rmelton@ball.com

ABSTRACT

Ball Aerospace COSMOS is a free and readily available open source command and control system for operations and test that brings a set of functionality to the aerospace community that has previously only been available in proprietary and expensive COTS solutions. Open source software like COSMOS fills a very specific need for SmallSat/CubeSat missions where spending more money on a traditional ground system than the full cost of the satellite doesn't fit within the budget. However, nothing is ever truly free and there are a lot of misconceptions around open source software in the Aerospace industry. What are the actual costs of open source software? Is it safe to use open source software? Why would I open source my software? This presentation will answer those questions and more while highlighting the choices we made in open sourcing the Ball Aerospace COSMOS Command and Control System. Additionally this presentation will discuss how adoption of COSMOS has gone in the two years since it was open sourced, and where it is going in the future.

WHAT IS OPEN SOURCE SOFTWARE AND WHY SHOULD YOU USE IT?

A simple definition is that open source software is any software where the source code is available to the software's users. The opposite of open source software is closed source software, where the source code is not available and only the original creators of the software have the ability to add features or modify it in any way. Microsoft Windows is a classic example of closed source software. For open source software, in almost all cases the users are allowed to modify and redistribute the software to anyone and for any purpose as long as they do so under the terms of the software's license. Common licenses are discussed later in this document.

Why should you use open source software? The most obvious reason is cost. Open source software typically costs nothing up front to use. A less obvious reason is that sometimes open source products are functionally more capable and colloquially "better" than their proprietary competitors. Many open source projects have far more contributors and are updated far more frequently than any single company could hope to achieve without a gigantic budget. However, the most important reason for using open source software in 2017 is that starting any new software project from scratch without taking advantage of the benefits of the thousands of man years of development that has been put into open source software is almost an insane proposition. Can you still write software without making use of open source, yes. Can your competitors do the same thing at ten times less cost by using open source software, yes.

IS IT SAFE TO USE OPEN SOURCE SOFTWARE?

This is probably the most common question regarding open source software. Many people are afraid that hackers are going to introduce malicious code into open source repositories that will then infect their organizations. The answer to this question is of course "it depends", but let me explain. All reputable open source projects have a limited set of committers who can actually put code into the repository. Mature repositories will require peer code review before anything is allowed to be merged into the "master" branch where software lives before it is officially bundled into a formal release.

Modern code services such as GitHub use a development model where outside contributors can request changes to a repository by submitting what is called a "pull request", basically a limited set of requested changes to the code. The mere act of submitting a pull request does not introduce any new code into the repository. After receiving a pull request, the repository's owners will review the change, and usually request further modifications or improved testing of the new code. Only after thorough review will any outside code be introduced.

In summary, all repositories have owners and the safety of a given codebase (as with all software), comes down to do you trust the owners of the repository. The code is open for review, though with large code bases doing a thorough review for malicious code can be very time consuming and costly. In general, it is the safest to only use open source code from known authors, and with a good preexisting following of users.

WHY SHOULD YOU OPEN SOURCE YOUR OWN SOFTWARE?

There are numerous reasons why companies choose to open source their software. I've broken down the main arguments by category.

To Make Money

There are several business models that revolve around open source software, but the three most common are commercial licensing, support contracts, and upsells. Commercial licensing allows the software to be used under different terms than the open source license. The most common use case is to couple commercial licensing with a GPL licensed open source license. Companies that want to create software using the GPL licensed code, but would like to keep the code they write proprietary even when they distribute it to their own customers, will often be interested in commercial licensing options.

Support contracts provide email, phone, or even on-site support to open source software. Depending on the complexity of the software and the value it provides, purchasing a support contract is often a more cost effective solution than hiring an in-house expert and it provides piece of mind in case something goes wrong in future.

Upsells involve providing additional capabilities through closed source software, or more restricted licensed software, that work alongside the open source software. This is a great model if the open source software gains a big following, and there is a high value complementary feature that would add a lot of value to existing users.

Additionally open sourcing software has a lot of marketing value for your company as a whole. A quality piece of open source software applies quality to your whole organization.

To Save Money

Maintaining software costs money and it is money that is seldom budgeted for. One solution is to open source the software and attract a community or specific individual maintainers who are excited to work with open source software. This community can provide bug fixes and new functionality free of charge to the sponsoring organization. It should be noted that building communities takes a lot of time. If a company wants to completely shed the cost of maintaining an open source piece of software, a much more effective method is to find a new owner or owners for projects to maintain them going forward.

To Attract and Retain Talent

Top engineers can spend huge portions of their life working on closed source software within organizations. As tenures with employers decline in our changing world, the possibility of having that software open sourced and providing benefit to the entire world can be a great attractor for talent that wants to make a bigger impact and "change the world". Also, the esteem of leading an open source project can be a great aspect for retaining top employees.

Altruistic Reasons

The most common altruistic reason to open source software is to give back to the community. Most of the software we write within large organizations relies on open source software underneath to produce the amazing internal software applications we write. Sharing some of these creations with the open source community is a way of giving back or paying it forward.

For most companies, the true reason to open source software is **"all of the above"**.

WHAT ARE THE ACTUAL COSTS OF OPEN SOURCING SOFTWARE?

The cost of open sourcing software can be broken down into two phases. The first is the upfront cost to do the work of getting upper management by in and approval, working with your legal team to choose a license, getting ITAR approval, and getting the code into a state where it represents the quality of your company. The actual cost of these steps will vary but a company should allocate between 3 and 6 months to complete all the steps.

The second phase of the cost of open sourcing software is hosting/distributing the software and maintaining it. Thanks to systems like GitHub and Gitlab, the costs of hosting and distribution are either free or very low for open source projects. The cost of maintaining the software (outside of the cost of maintaining the software as closed source) will vary based on the amount of community interaction the software attracts. In general this will be very low for the first few years depending on how much effort you put into marketing the new open source software.

WHAT ARE THE DIFFERENCES BETWEEN COMMON OPEN SOURCE LICENSES?

The one common aspect of all open source licenses is that the users of the code will be allowed to look at the source code. Other than that, there are key fundamental differences between the commonly available open source licenses that users of open source software must be aware of. Note: This is not to be taken as legal advice. Please refer to a lawyer before making any decisions regarding choice of license or the implications of using software under any given license. The following is a list of the most common open source licenses.

GPL[1]

The GNU General Public License (or GPL) is one of the most common licenses found in open source software covering major products such as the Linux kernel. It is one of the longer licenses available and establishes a concept called "copyleft". Copyleft refers to the feature of the GPL license that any software that is derived from or uses GPL software for its functions must typically also be licensed under the GPL. For this reason, GPL licensed software is sometimes considered "infectious". Two versions of this license are commonly seen (v2 and v3), the differences between the two are primarily related to the possible assertion of patent rights from users. Large patent heavy organizations will want to look closely at these provisions.

One of the most common misconceptions of the GPL license is that is forces companies to distribute their code. This is not true. Companies are only required to distribute source code, if asked, and if they had previously distributed a binary only version of the software to someone. What the GPL license does require, is if you do choose to distribute the code to anyone for any reason, that the recipient must also gain all the rights to use the code for whatever purpose they want, or to redistribute the software however they want, as required by the GPL. If the code is never distributed, then no rights are passed to anyone.

LGPL[2]

The GNU Lesser General Public License (LGPL) is very similar to the regular GPL license, but has fewer restrictions. It is sometimes incorrectly referred to as the Library GPL because it is typically used with software libraries. The Qt GUI framework is an example of a popular open source library licensed under the LGPL. The key difference with the LGPL is that it protects the code licensed under it in the same way as the GPL, but is not infectious to the larger product built on top of it. For example, a company can create a derived product or a new product that makes use of LGPL code, but still keep the overall product proprietary or licensed under commercial terms. The company following this model is only required to provide access and rights to the LGPL licensed portion of the code to anyone the overall product is distributed to.

MIT / BSD Licenses[3,4]

The MIT and BSD license are both very simple (3 paragraph) and permissive licenses. They assert who is the copyright holder of the software, and explicitly declare that the software comes without warranty. Both were written before patents on software were common, and therefore do not address that subject. These are the best licenses to choose if you want the maximum adoption of software (which allows proprietary use cases).

Apache[5]

The Apache license is another common and fairly permissive license. However, it is much longer and more complicated than the MIT / BSD licenses and does address patents. Probably the best license choice for large organizations that don't mind using a permissive license.

Other

There is nothing magical about the licenses listed above, and there is nothing that prevents you from licensing code that you own the copyright to under different terms. However, using a license not listed above may significantly affect adoption of your software as the implications of a custom license will not be well understood.

HOW DO YOU PROTECT COPYRIGHT WITH OPEN SOURCE SOFTWARE?

Typically organizations maintain copyright to software they write unless a contract specifically declares otherwise. However, the game changes once a project starts accepting code from outside contributors. Typically any code contributed to a project remains the copyright of the contributor, with the code being licensed under the same terms as the open source software is being distributed under. However, this can cause issues for organizations who want to maintain the capability to use dual licensing models where software is offered under both an open source license and a commercial license. (As soon as someone contributes code, the original sponsoring organization no longer owns copyright to the complete work).

The solution to this problem is using something called a Contributor License Agreement (CLA). A CLA defines the rights that the contributing copyright owner gives to the organization that maintains the open source product. Typically these documents assure that the contributor must have the copyright to the code being contributed, it must not be encumbered by patents, and that the contributor gives the project the rights to use the software however they wish. Overall, copyright usually remains with the contributor.

WHAT CHOICES WERE MADE IN OPEN SOURCING BALL AEROSPACE COSMOS?

The decision to open source COSMOS was made in August of 2014, and it took around five months to complete the processes of choosing the license (GPLv3), getting ITAR approval, setting up a repository on GitHub, and actually releasing the software. Several more months before August were spent pitching the idea to management. COSMOS was officially released on December 29th, 2014.

COSMOS is available under both an open source license (GPLv3) and through commercial licenses. GPLv3 was chosen as it is a great license to encourage adoption and support within the community, and it also provides the right level of legal rights to discourage someone from making a proprietary solution on top of it. For those wishing to create proprietary solutions with COSMOS, the commercial licenses are very affordable and we have already sold several of them.

Contributors to COSMOS must sign a Contributor license agreement. This allows Ball to maintain full rights to the COSMOS software and to be able to license it including outside contributions, under commercial license terms. View our GitHub repository to read the terms of the agreement.

COSMOS is distributed and hosted on GitHub at https://github.com/BallAerospace/COSMOS.[6] To date since our open source release, we have completed 190 change requests to COSMOS with 28 tickets being open at the time of this writing. Of those 218 total tickets, approximately 30 have come from outside of Ball. Seven pull requests including code from authors outside of Ball have been submitted and committed to the project.

WHAT HAVE BEEN THE RESULTS OF OPEN SOURCING BALL AEROSPACE COSMOS?

According to rubygems.org, COSMOS has been downloaded and installed over 60,000 times. It is being used around the world from the U.S. to Brazil to South Korea to Australia by both universities and aerospace companies. Companies such as NASA, Raytheon, Lockheed, Blue Canyon, and BAE are using COSMOS for everything from CubeSat missions to internal testing projects. Universities such as CapTech, Utah State, and John Hopkins are using it for senior projects and experiments. COSMOS is a fully functional command and control system that can benefit every aerospace company in the industry, and likely many companies outside of aerospace. Find out more about COSMOS and install it yourself from: http://cosmosrb.com.[7]

---

[1] GNU General Public License - https://www.gnu.org/licenses/gpl-3.0.en.html
[2] GNU Lesser General Public License - https://www.gnu.org/copyleft/lesser.html
[3] MIT License - https://opensource.org/licenses/MIT
[4] BSD License - https://opensource.org/licenses/BSD-3-Clause
[5] Apache License - http://www.apache.org/licenses/LICENSE-2.0
[6] COSMOS Source Code: https://github.com/BallAerospace/COSMOS
[7] COSMOS Website: http://cosmosrb.com