# COTS, FLOSS, and Market Freedom in Safety-Centric Industries

Franco Gasperoni

gasperoni@adacore.com

AdaCore, 8 rue de Milan, 75009 Paris, France

## Executive Summary

This article discusses the relationship between commercial off-the-shelf software (COTS) and freely licensed open source software (FLOSS) from a purely business perspective. The emphasis is on safety-centric industries such as aerospace.

This works shows that, contrary to popular belief, COTS and FLOSS are two orthogonal and non-conflicting attributes of a software product since the first deals with commercial off-the-shelf availability while the second deals with licensing terms. COTS vendors have a continuum of licensing choices among:

- Restrictively licensed COTS (RL-COTS);
- FLOSS licensed COTS (FL-COTS);
- A mixture of RL-COTS and FL-COTS.

This work shows that the availability of source code in FL-COTS is not its key advantage when it comes to safety-focused industries. Rather, it is the set of freedoms guaranteed by FLOSS that makes FL-COTS an interesting alternative over RL-COTS.

More specifically, copyright grants a temporary monopoly on a software product. Up until recently this monopoly has been used to distort market freedom by creating vendor lock-in for software changes, support, and certification material. Our work shows that this is not and should not, in a free market economy, be a given. In particular, COTS sold with a FLOSS license (FL-COTS) has all the advantages of restrictively licensed COTS (RL-COTS) without the vendor lock-in disadvantages.

This paper does not claim that FL-COTS is systematically superior to RL-COTS since the choice between two COTS products involves commercial and technical aspects in addition to legal ones. Our objective is to show that other things being equal FL-COTS is always better than RL-COTS because the absence of lock-in creates a market free of monopolistic entanglements: FL-COTS is the ultimate in market freedom.

This article draws on over 10 years of working experience at AdaCore the commercial company behind GNAT Pro, the Free Software Ada 83, Ada 95, and now Ada 2005 technology used by the primary actors in safety focused industries such as aeronautics and air traffic management.

# 1. Freedom and Monopoly

### 1.1 Free Market

A free market is a market where economic exchanges are unconstrained and voluntary. In many countries, for example, people can trade goods such as cars or houses freely. Among other things a person in need of such a good has the freedom to:

- Contract someone to build it (e.g. custom-made house);
- Buy a pre-built, commercially available version (e.g. car in a dealership).

The prices and varieties of custom-made or commercially available goods are set by the laws of supply and demand.

### 1.2 Monopoly

A free market may exist for the exchange of a set of goods or may exist for an entire economy. No economy is 100% free market since in every country, even capitalist ones, restrictions are placed on the ownership and exchange of certain types of goods such as alcohol or tobacco.

A monopoly is a market situation where there is only one provider of a good. Monopolies are characterized by a lack of competition for the good and a lack of viable substitutes. In several countries, for instance, telephony was a monopoly up until recently.

A free market does not require the existence of systematic competition. A free market does not, however, prevent such competition from coming into existence.

# 2. COTS

People in need of software can turn to the market for purchase of custom-made or commercial-off-the shelf software (COTS). Technically speaking COTS is the term for software products that are ready-made and sold to the general public at list or market prices.

Several factors distinguish COTS used in safety-related or safety-critical systems from general COTS sold for consumer or office automation purposes, namely:

- Stringent customization, repair, and evolution requirements;
- Need for rapid and high-quality support;
- Evidence for quality assurance;
- High switching costs.

### 2.1 COTS Customization, Repair, and Evolution

COTS in safety-related or safety-critical systems often need to be tailored for specific use. In embedded devices, for instance, COTS operating systems or kernels require a great degree of customization: from creating or adapting the BSP (board support package) for the target hardware, to configuring certain OS modules such as the scheduler.

Moreover, because COTS is designed to meet the needs of a majority of disparate users there is a possible compromise on features, performance, and usability. This means that in a number of cases, and this is particularly true in safety-centric industries, COTS need to evolve and be adapted to match a specific set of customer feature or performance requirements.

In addition, COTS errors need to be repaired in earnest as they may slow down or even block a safety-related software project.

### 2.2 COTS and High-Quality Support

Because of the needs for COTS customization, repair, and evolution, and the level of sophistication of the COTS technology used in safety-related projects (e.g. in an airplane), rapid and high-quality support is important. Project costs, deadlines, and ultimately success are affected by the quality and turnaround times of the support offered by the COTS vendor to the customer.

### 2.3 COTS and Quality Assurance Evidence

Before deployment, developers of safety-related applications need to provide evidence for their quality. From IEC 61508 [IEC] to RTCA/EUROCAE DO-178B/ED-12B [DO1] quality evidence requirements vary depending on the type of industry and level of safety.

In the avionics industry, for instance, the DO-178B/ED-12B certification protocol [DO1] stipulates that the contribution of software to potential failure conditions in an airborne system is used to define 5 software levels. Each software level imposes specific objectives on software planning, development, verification, configuration management and quality assurance. The stringency of these requirements increases with higher safety levels.

Such needs for evidence ripple down all software development levels including independent COTS vendors who must themselves provide evidence for the quality of their products. In the case of DO-178B/ED-12B this type of evidence is often referred to as a certification package.

### 2.4 COTS and High Switching Costs

The elevated levels of financial, infrastructure, and training investments in key COTS components such as kernels, operating systems, databases, and software development tools used in safety-related projects means that switching costs between competing COTS may be significant.

## 3. Copyright and Software License

### 3.1 Software is protected by Copyright

Software, like literature, music, and cinematography is protected by copyright and governed by the Berne convention [BRN]. Copyright confers copyright holders a temporary monopoly on their work. The duration of this monopoly varies from life plus 70 years for individuals to 70 years after the first publication in the case of corporations (95 years in the USA). After this duration the monopoly ceases and the software work falls in the public domain.

The monopoly conferred by copyright to its holder concerns the right to use, copy, distribute, modify, and make derived works of the software (whether in source or binary form). To undertake anyone of these activities one must obtain a license from the copyright holder.

### 3.2 Software License

Broadly speaking a software license is the legal document listing the things the software recipient is and is not allowed to do with the software.

Copyright holders can, at their discretion, charge a fee in exchange for a copy of the software work and its attached license. In this case what the recipient is buying is not the software work per se but a copy of it along with its license.

The software license can be extremely restrictive limiting the rights of the users to a bare minimum and, as we will see later on, locking in the user after the initial purchase of a COTS. It can also be more permissive (freer) thereby reducing or even eliminating the effects of the monopoly conferred by copyright.

In this paper we do not claim that software copyright should be abolished, but rather that customers have the ability to influence the market and obtain more liberal licensing terms than what has been typical so far, thus removing potential vendor lock-in while allowing competition throughout the life-cycle of a COTS product.

## 4. FLOSS

FLOSS (Freely Licensed Open Source Software) is a recent term used to denote Free Software [FSF] and Open Source Software [OSS]. For many people FLOSS means:

- Access to the sources;
- Existence of a community dedicated to the evolution of the software;
- Free-of-charge software availability.

While these attributes are usually true, this is not what FLOSS is all about. As we will explain, FLOSS is all about the terms and conditions of the software license.

### 4.1 Free Software and Open Source Software

At the root of FLOSS is the Free Software movement started by Richard Stallman in the mid 80s [FSF]. In 1998, with the emergence of the open and community-based GNU Linux project, which is licensed as Free Software, some members in this community started using the term Open Source Software instead because of the openness of development, the existence of an open community of developers that Free Software licenses enables [OSS]. Today Free Software and Open Source Software are separate movements.

The fundamental difference between the two movements is philosophical: Open Source Software is a development methodology (the bazaar) [RAY] while Free Software is all about freedom including, and this is the key point from a business perspective, **freedom to market** [FSS, INF].

### 4.2 FLOSS Licenses

Even though Free Software and Open Source Software are associated with different criteria for acceptable software licenses [FSD, OSD], in practice nearly all software meeting one definition also meets the other. More specifically, all Free Software (FS) licenses qualify as Open Source Software (OSS) licenses and most OSS licenses are also FS licenses (the one notable exception being the Reciprocal Public License [RPL]). The term FLOSS has come to denote this common set of licenses.

The key freedoms embodied in FLOSS licenses are:

- The freedom to run the software for any purpose;
- The freedom to redistribute the software;
- The freedom to create and distribute derived works;
- The freedom to market the original or derived versions of the software.

Access to the source code and build procedures is a consequence of the freedom to create derived works since without these FLOSS recipients are not free to create and market derived works.

The ability to create and redistribute derived works allows the creation of an open community of developers collaborating in the evolution of the software. This is a possible side-effect of FLOSS licenses, not a requirement and not a guarantee.

FLOSS licenses put no restrictions on distribution or redistribution fees. The recipient of FLOSS can sell/resell copies of the software at any price.

### 4.3 Microsoft Windows and GNU Linux

As mentioned before, unless explicitly placed in the public domain, the use, copy, distribution, modification, etc. of all software whether FLOSS or otherwise, is governed by the laws of copyright. From this standpoint there is absolutely no difference between Microsoft Windows and GNU Linux. These are both copyrighted works. To be able to use them one must obtain a license from its copyright holders. This software license grants users certain rights and may impose certain restrictions in the use, modification, and/or redistribution of the software. The key difference between Microsoft Windows and GNU Linux lies in the terms of their license. This is quite a difference since the licensing terms of Microsoft Windows are very restrictive, while those of GNU Linux are far more permissive.

## 5. COTS and Software Licenses

### 5.1 COTS and FLOSS

Up until recently COTS was marketed under very restrictive licenses which severely limited the user's ability to use and copy COTS let alone make modifications or redistribute the software. These restrictive licenses created vendor lock-in after the COTS purchase, making the monopoly on the COTS very apparent.

Because waiting 70 or 95 years for the software work to fall in the public domain was not a viable alternative and in any case the potential lack of source code would have made this event irrelevant, Richard Stallman launched the notion of Free Software in the mid 80s. This started as an ideological movement with the creation in 1985 of the Free Software Foundation. Thus, initially, COTS was opposed to Free Software in people's minds since the first had to be purchased and had a restrictive license while the latter was available free-of-charge and came with a freer license.

This changed with the creation of corporations such as Red Hat in 1993 [RED] and AdaCore in 1994 [ADA]. Whether it's GNU Linux or the GNAT Pro Ada development environment these companies market COTS licensed under a FLOSS license (FL-COTS). Thus, when looking for software two orthogonal aspects must be distinguished:

(a) Whether the software is COTS, i.e. commercially available off-the-shelf;

(b) The terms and conditions of the software license (restrictive or FLOSS).

### 5.2 COTS Licenses: A Variety of Possibilities

It is important to understand that a COTS vendor owning the copyright of its software is under no obligation to:

- Give the same software license to all its customers: higher paying customers may obtain more favorable terms (e.g. a FLOSS license) while lower paying ones may receive a more restrictive license.

- A COTS vendor can provide certain of its products under a restrictive license, while marketing other products under a FLOSS license.
- The COTS vendor is under no obligation to make the sources/binaries available to all (e.g. by internet download) when the software is licensed as FLOSS. The source code must be provided only to recipients of FL-COTS.

The point of the above remarks is to show that COTS vendors have a continuum of choices between providing only restrictively licensed COTS (RL-COTS), a mixture of RL-COTS and FL-COTS, or entirely FL-COTS.

## 6. Software Quality and Software Licenses

What are the quality guarantees of FL-COTS and RL-COTS? Answer: absolutely none (unless explicit evidence, such as certification material, is produced to the contrary).

With RL-COTS customers may not even have access to the sources. Even when customers have access to source code and build procedures no license (FLOSS or otherwise) requires the availability or redistribution of design documents, development plans, quality assurance procedures, test suites, or other software development artifacts.

The quality and quality-oriented procedures vary greatly from one COTS product to another and the software license gives no guarantees in this matter. Some FLOSS proponents claim that because FLOSS is developed by a community of dedicated developers this yields better software. There is no systematic evidence of this. This really depends on the project.

There is nothing that prevents FL-COTS from achieving high-quality and being used in safety-critical systems. For the GNAT Pro Ada compiler alone, for instance, AdaCore has accumulated more than 10,000 tests in 12 million lines of Ada code. Furthermore, GNAT Pro is used in several safety-critical projects in the military, avionics, space, and railway industries (Canadian space arm, Airbus A380, 767 Tanker, C130AMP, Boeing 787 …). Finally, GNAT Pro comes with an Ada runtime certifiable to DO-178B/ED-12B Level A.

## 7. RL-COTS and Monopoly

RL-COTS is a good example of a product where market freedom exists at the initial purchasing stage when choosing among competing COTS to perform a given task.

After the initial purchasing stage RL-COTS has a number of drawbacks in its day-to-day use in safety-related projects. Up until recently, because the industry’s only choices were between do-it-yourself and RL-COTS, RL-COTS downsides were considered as inevitable.

These downsides stemmed from the fact that the customer had to abide by the restrictive rules set by the RL-COTS owner. Once an RL-COTS was chosen and deployed the monopoly conferred by copyright to its owner was used to create:

- Vendor lock-in for software customization, changes, and evolution;
- Vendor lock in for support;
- Vendor lock-in for certification material (e.g. for DO-178B certification).

Because of the high investments and high switching costs, customers cannot easily workaround this vendor lock-in by using two competing COTS in safety-related projects and having the COTS vendor compete for better service throughout the lifetime of the project.

Note that the previous drawbacks do not include absence of source code. In fact, thanks to the pressure exerted by the appearance of FLOSS, it is now possible to obtain the sources of certain RL-COTS against a modest fee. The difference of this source code with that of FL-COTS resides in the restrictions attached to the former. In particular, and unlike FL-COTS, customers of RL-COTS cannot use its sources to free themselves of vendor lock-in.

## 8. FL-COTS and Free Market

### 8.1 FL-COTS: No Vendor Lock-in for Software Evolution

RL-COTS forces customers to go to their original COTS vendor for software evolution. Because the COTS vendor has a monopoly over changes to its software this can be arbitrarily expensive.

With FLOSS the customer is not hostage of its COTS vendor. Because FLOSS licenses guarantee a **free market** when it comes to software changes, FL-COTS customers can solicit competitive bids when software modifications, additional tools, or extra libraries are needed. This is attractive for business and particularly so in safety-focused industries where the vendor lock-in is significantly more pronounced because of the heavy engineering investments required.

Taking GNU Linux as an example there are today several reliable COTS vendors for it, some making available specialized versions for the embedded market. Because of the lack of vendor lock-in, a customer that has committed significant resources to the use of the GNU Linux kernel in its software infrastructure can switch COTS vendors or can occasionally use additional vendors to add industry-specific features.

In the case of GNAT Pro, one of AdaCore customers (EUROCONTROL) needed an additional tool. To be economical this tool had to leverage on the guts of the GNAT Pro compiler to avoid redoing a parser and a semantic analyzer for Ada. To demonstrate that FL-COTS has no vendor lock-in EUROCONTROL did an open bid for the additional tool which was won by an independent French firm.

As another example, there was a segment in our industry that we (AdaCore) were not serving. An independent company used the GNAT Ada technology to build and market a COTS product for that market segment. We have since entered that market segment and are now competing with our own technology.

### 8.2 FL-COTS: No Vendor Lock-in for Support

The source code, scripts, and build procedures of RL-COTS cannot be given by a customer to third-party service providers to obtain higher levels of support whether in the form of training, consulting, or professional services.

Conventional wisdom tells us the COTS vendor knows best because it has developed the technology. In large software firms, however, the development team of the COTS vendor is segregated from its services organization and is often shielded from the final customer.

The possibility to have support organizations separate from the original COTS vendor allows the creation of a cottage industry which may be in a better position to provide localized services.

Note that for strategic reasons a number of vendors selling RL-COTS have decided to unbundled or outsource part of the service business inherently linked with their products. Some (e.g. Microsoft) have done this to create and foster an ecosystem of partners around their products.

Again, the difference with FL-COTS is **market freedom**. With RL-COTS it is the COTS vendor that solely decides its support strategy. With FL-COTS it is the market.

They key point here is not that production of software and its support should be systematically unbundled, rather it is the customer that should have the freedom to decide so when a business case exists.

### 8.3 FL-COTS: No Vendor Lock-in for Certification Material

COTS is designed to meet the needs of a majority of disparate users. In safety-related industries, certification material may be available only for certain certification protocols such as DO-178B/ED-12B or not available at all.

In the case of RL-COTS it is again the vendor that has a monopoly in what to do. With FL-COTS the customer can contract an independent provider for certification materials and audit trails. Incidentally, in this case the customer may decide to own the copyright for the certification material which he may provide to its partners, subcontractors or perhaps resell. Alternatively, an industry segment may decide to pool its resources to create certification evidence for kernels such as GNU Linux.

The key point here is that quality, when present, can be reversed engineered. Once more the difference between RL-COTS and FL-COTS is **market freedom**. With the former the COTS vendor is the only one deciding who can produce certification evidence for its software. With FL-COTS it is the market.

## 9. Conclusion

In the previous sections we have shown how FL-COTS is unaffected by the downsides (for customers) of RL-COTS. COTS vendors have a continuum of choices between providing only RL-COTS, a mixture of RL-COTS and FL-COTS, or entirely FL-COTS.

This paper shows that all other things being equal an FL-COTS is always better, for the customer, than an RL-COTS because the lack of vendor lock-in for software changes, support, and certification material aligns the COTS vendor interests with the customer's around high-quality products and services at competitive prices. Put it another way, to the question:

> Should the industry lobby with its vendor(s) so that, **when it is of value**, the vendor(s) make their COTS available **to them** with a FLOSS-like license?

The answer is a resounding yes.

In summary, other things being equal, FL-COTS is always better than RL-COTS because the absence of lock-in creates a market free of monopolistic entanglements: FL-COTS is the ultimate in market freedom. It is somewhat ironic that some RL-COTS vendors who are strong proponents of a free market just to fall short of this for their own products.

## 10. References

[ADA] http://www.adacore.com/

[BRN] Berne Convention for the Protection of Literary and Artistic Works. Available at http://www.wipo.int/treaties/en/ip/berne/trtdocs_wo001.html.

[DO1] Software Considerations in Airborne Systems and Equipment Certification. RTCA/EUROCAE DO-178B/ED-12B.

[FSD] http://www.fsf.org/licensing/essays/free-sw.html

[FSF] http://www.fsf.org/

[FSS] http://www.fsf.org/licensing/essays/selling.html

[IEC] http://www.iec.ch/zone/fsafety/fsafety_entry.htm

[INF] Panelists consider the 'business case' for open source, Infoworld, Oct. 1, 2002. http://www.infoworld.com/articles/hn/xml/02/10/01/021001hnbizopen.html?s=IDGNS

[OSD] http://www.opensource.org/docs/definition.php

[OSS] http://www.opensource.org/

[RAY] The Cathedral & the Bazaar, by Eric S. Raymond. O'Reilly, 1999.

[RED] http://www.redhat.com/

[RPL] http://www.opensource.org/licenses/rpl.php