Governance of Open Source Software

By
Andrew J. Murren
Sila Solutions Group
Arlington, United States
amurren@ieee.org

The Open Source Software Institute's mission is to promote the development and implementation of Open Source Software (OSS) solutions within U.S. Federal, state, and local government agencies. To achieve this mission, OSSI serves as an ***Open Source Advocate****, an* ***R&D Facilitator****, and a* ***Strategic Integrator*** *for all sectors.*

9 March 2016

Table of Contents

***Abstract* – The use of Open Source Software (OSS) in enterprises is more pervasive than most organizations know. Many organizations do not understand OSS or have misconceptions about its nature. This lack of understanding results in many organizations not managing the OSS running on their enterprise infrastructure. In this document we introduce OSS's differences and similarities with commercial proprietary software, and how to include OSS in an organization's existing governance framework.**

Keywords: Open source software; oss; floss; foss; governance; open standards; free/libre software; free software; enterprise

I. Introduction

In today's enterprises the use of Open Source Software is ubiquitous. Despite its prevalence most organizations do not understand the full potential or true costs of OSS. Organizations need to include OSS in their existing IT governance processes to achieve the benefits OSS offers, and to avoid technical and legal issues that can arise from failing to manage OSS. By its very nature, OSS differs from commercial proprietary software in significant ways, and needs to be dealt with differently from traditional commercial proprietary software governance structures.

Organizations find OSS attractive because OSS is based on open standards; improving interoperability and avoiding vendor lock in. Software developers use OSS to make their jobs easier and allow them to develop applications faster. Enterprise infrastructure support organizations are attracted to OSS due to the flexibility it offers. In acquiring OSS, one does not have to go through standard purchasing channels, i.e., OSS can be downloaded and used without the organization having any means of tracking its use. It is not unusual for system administrators, network administrators, or developers to use OSS without any controls. Any Java developer can use one of the many OSS development frameworks with no process to evaluate the software for security vulnerabilities, review of the license potential legal liabilities to the organization, or a way to ensure the software is updated when updates are released. Therefore, OSS brings not only the benefit of ease in using pre-developed software OSS may increase security vulnerabilities, as well as, future legal ramifications. This can be mitigated by incorporating OSS into current governance structure for compliance.

There are two audiences for this paper. The first audience is those who want to understand more about what Open Source Software is. The second audience is those executives and senior managers who want to understand what OSS is, how it potentially impacts their organization, and an option on how to manage it. This paper will provide this audience with enough understanding of OSS to begin the process of developing realistic policies on OSS and incorporating it into the organization's governance processes in order to reduce their organization's exposure to legal, reputational, technical, and other risks.

II. Defining Open Source Software

A. Department of Defense Definition

One of the first and most significant hurdles to overcome in integrating OSS into an organization's governance structure is to understand the definition of OSS. In its 2009 Open Source Software memo the Department of Defense (DoD) defined OSS as:

> "software for which the human-readable source code is available for use, study, reuse, modification, enhancement, and redistribution by the users of that software." [1]

This definition can be used for any US government organization that does not have their own definition.

B. Open Source Community Definitions

The Open Source Initiative's Open Source Definition [2] and the Free Software Foundation definition of Free Software [3] provide the criteria to determine exactly what the two organizations consider meeting the definition of Open Source Software or Free (or Libre) Software respectively. The primary difference between Open Source and Free/Libre software is in what the software license allows.

C. The OSSI Definition

OSSI defines Open Source Software (OSS) as:

> "Source code that is publicly available and meets the criteria listed in the Open Source Definition (OSD) as defined by the Open Source Initiative (OSI)."

The OSSI definition provides organizations with a simple definition that can be used to identify OSS and integrate it into their governance program. The definition also enables organizations to understand the similarities and differences between Open Source and other software products, and how these differences require changes to existing governance processes and can impact other existing policies and procedures.

III. Types of Software

A. Closed Source

Closed source software is source code that does not meet the criteria of Open Source. The primary types of closed source software are listed below.

1) *Proprietary Software:* Proprietary software is source code not releasable to the general public without a license. The vendor or developer will provide no access to the source code, or provide access to selected individuals or organization under very strict conditions and limits per a licensing agreement. Proprietary software may implement proprietary solutions that preclude them from interoperating with software from other vendors or solutions based on industry or open standards.

Frequently proprietary software is also called commercial software, but that is not strictly accurate. While the vast majority of software sold is proprietary software, some proprietary is never sold. Also some OSS is sold, which makes it commercial. For this reason it is both inaccurate and confusing to use the terms proprietary and commercial interchangeably.

2) Shareware: Shareware is an executable program releasable to the public that can be used for a limited time at no cost to the consumer. Once that time has expired or a threshold has been reached, the vendor or developer expects to be paid for use. The source code is never released for evaluation of quality, security, or the inclusion of malicious software (malware). Due to the inability to evaluate the shareware's source code, this type of software should never be used on sensitive networks or systems.

3) Freeware: Freeware is similar to shareware in that the compiled executable is release to the public for free use, and unlike shareware, no payment is expected. Like shareware the source code is never released, and hence, there is no way to evaluate the source code for quality, security, or the inclusion of malware. Like shareware, freeware should never be used on sensitive networks or systems.

4) Private: Private software is any software than the developer does not share. Most software built by organizations for their exclusive internal use is private software.

B. Open Source

Open Source Software is software for which the source code is available for use and modification. It is also based upon open or industry standards which allow for interoperability with any other software that implements the same standards.

There is a philosophical division within the community that develops and advocates for OSS. The division is between advocates for Free/Libre software and Open Source Software. This division has led to some of the confusion of what OSS is. The philosophical division has no practical effect on organizations using OSS except in the terms of the software licenses.

Software licenses for OSS should be reviewed by the organization's legal team before OSS is used in any manner. For the sake of consistency with the OSSI definition we will use the term OSS when referring to non-closed source software. This does not reflect any philosophical position; it is only to maintain consistency.

1) Free/Libre Software: The Free Software Foundation (FSF) is the original organization dedicated to providing freely distributable software source code. The FSF was founded on the concept that all software source code should be available for others to examine, use, and modify. The FSF considers it perfectly acceptable to charge money for software, provided the source code is given to the buyer and the buyers freedom to examine, use, and modify the source code is not prevented. The term 'free' has caused a lot of confusion over time so the inclusion of the term 'libre' has been added to make the freedom aspect of their project clearer. The FSF has advocated the use of the term 'Libre Software' to make the concept of freedom central and to avoid confusion over no-cost. The FSF has advocated the concept of 'copyleft', which is a form of copyright that protects the user's rights over the rights of a works originator. The FSF considers approximately 40 licenses as meeting the Free Software definition [3]. However, not all licenses are considered fully 'copyleft'.

2) Free/Libre and Open Source Software (FLOSS): Free/Libre and Open Source Software (FLOSS) and Free and Open Source Software (FOSS) are terms advocated by the FSF as neutral terms to encompass all software that meets the definition of both the FSF and the OSI. FSF prefers the use of FLOSS. There are no practical differences in which term is used since the most important differences between these different terms are the software license terms.

3) Open Source Software (OSS): The Open Source Initiative has developed the Open Source Definition (OSD) [2] which provides a much larger umbrella for software to be considered OSS. OSSI has combined their definition with other OSS definitions to provide a more global and comprehensive definition. Not all of the software licenses reviewed for compliance with the OSD by the OSI meet the more narrow FSF definition of Free/Libre Software. The OSI recognized over 70 licenses [4], including all of the FSF approved licenses, as meeting the OSD.

C. Additional Software Terms

1) Orphan Works: The US Patent and Trademark Office (USP&TO) defines an orphan work as:

> "copyrighted works whose owners cannot be identified or located, making it impossible to negotiate terms for their use." [5]

Determining if a work is an orphan work can be time consuming and imprecise and the current position of the USP&TO make them difficult to use. There is an effort to get the USP&TO to change its stance on orphan works in order to allow their use. A major consideration about projects that are or may be orphaned is the lack of support and updates from the developer. Orphaned projects do not receive security patches, feature updates, or any form of support from the developer and copyright owner; however, there may be some support from other users of the software. Any security patches or feature updates contributed by anyone other than the developer and copyright owner should be disallowed. If for some reason there is a need to use a patch from someone other than the copyright holder it must be given very careful legal, security, and technical review before being used.

Depending on the OSS license the orphaned work is released under, it may be possible to use the available source code as a baseline to begin independent development of the project, a practice known as

creating a 'fork'. Forking is explicitly permitted in licenses that meet both the OSD and the Free Software definition. However, the decision to use source code from an orphaned project should be done after a full legal review to avoid any future liabilities. The organization must also conduct an analysis of the costs to develop and maintain the forked project.

2) Commercial Off-The-Shelf (COTS): While this is an official US government definition it is equally applicable to non-governmental organizations. This definition works well for any organization because it covers commercial proprietary, commercial OSS, and non-commercial OSS equally. This definition is:

> "A software and/or hardware product that is commercially ready-made and available for sale, lease, or license to the general public." [6]

It is important to realize that frequently commercial proprietary COTS products contain or are built upon OSS software.

3) Government Off-The-Shelf (GOTS): The definition of GOTS is similar to COTS with the difference being that the product is paid for by the government for government use.

> "A software and/or hardware product that is developed by the technical staff of a Government organization for use by the U.S. Government. GOTS software and hardware may be developed by an external entity, with specification from the Government organization to meet a specific Government purpose, and can normally be shared among Federal agencies without additional cost. GOTS products and systems are not commercially available to the general public. Sales and distribution of GOTS products and systems are controlled by the Government." [6]

4) Hybrid: This term refers to software that is composed of software components from different sources, such as OSS, GOTS, COTS, and private. It most often is used to refer to software developed exclusively for an organization's internal use that uses OSS components (such as libraries) and private code developed internally or under contract by a third party.

IV. Software Support

A. Types of Software Support

When an organization acquires commercial software there is an expectation that the vendor will provide some level of support via a Service Level Agreement (SLA). Depending on the software there may be some consulting service available to install, configure, conduct training, or to customize the software. As a minimum vendors generally provide support via a call center, email, and documentation for users and administrators for customers. Almost all vendors now provide support on-line forums where other users of the software and employees of the vendor can answer questions. Some vendors provide support via chat or Instant Messaging (IM). All developers, both closed source and open source, provide security patches, bug fixes, and feature updates for supported software. When considering the use of OSS organizations should include what level and type of support they will need.

B. Commercial Support

Commercial support is when technical support for the software is available for a fee or the support is included in the license cost. Most frequently the software vendor provides the technical support, but support may also be purchased from integrators or other third party firms. Commercial proprietary vendors and commercial open source vendors often provide extensive documentation, call center support, and consulting services in addition to software patches and updates.

C. Community Support

Community support is when technical support is non-commercial and comes exclusively from the developers and the user community. Generally community support is available through the project web site in the form of documentation and user forums. Many vendors provide community support in addition to their commercial support.

In his paper "Community and Commercial Strategies in Open Source Software" [7] Anthony Wasserman identifies two types of community supported OSS projects, independent projects and foundation-based projects. Independent projects are frequently developed by a small core of volunteers with support from the developers and the user community. Foundation-based OSS projects are supported by a non-profit organization with financial support from corporations. The foundation-based projects have formal governance structures in place that result in high quality and very stable products. Some foundation-based projects offer limited commercial support in addition to their community support.

The practical impact of using community supported OSS is that some level of in-house technical support capability may need to be developed and supported in order to provide the expertise that would be traditionally available from commercial support. Each OSS product will have different levels of support requirements. In some cases it may be sufficient to have an internal user forum, for others it may require a small dedicated team.

Table 1. Software Support

	Commercial Support	**Community Support**	
		Foundation Based	**Independent**
Bug Fixes	Yes	Yes	Yes
Call Center Support	Yes	No	No
Chat or IM	Some	Some	Some
Consulting Services	Some	Some	No
Documentation	Yes	Yes	Yes
Email Support	Yes	Yes	Yes
Feature Updates	Yes	Yes	Yes
On-line Forums	Yes	Yes	Yes
Security Patches	Yes	Yes	Yes
SLA	Some	No	No

V. Classes of Software

There are many different types and classes of software for many uses. For the purposes of this topic we can reduce this to two primary classes, applications and components. An organization may determine more classes are required for their particular situation.

A. Applications

Applications are essentially off-the-shelf products that are complete and that can be run without any additional development. Some examples are the Apache web server, the PostgreSQL database, or the LibreOffice office suite. For inventory, configuration management, change control, and other purposes OSS applications can be managed and tracked much the same way as their commercial closed-sourced equivalents.

B. Components

A component is any software that cannot be run without additional development work. Software libraries are an excellent example of components. A major challenge to components is that once used they are often difficult to track. Because of this management and tracking is critical for keeping components up to date with security patches and feature updates, and to maintain license compliance.

VI. Differences Between Open and Closed Source Software

There are several key differences between OSS and closed source software. The most obvious is the cost which is tied directly to a second important difference. Because OSS licenses do not restrict the use of the software by the user, the user can make and run as many copies as they desire. The user can use the software for whatever purpose they want. This lack of restriction on the use of the software eliminates the primary ways of charging for the software and means to track its usage. Because there is no usage license, an automated license tracking solution may not provide accurate tracking and management of OSS deployments. Other tools such as configuration management systems or dedicated OSS management tools may provide a better solution. It is important to note that some commercial support for OSS may have a per-seat or annual cost for the support or proprietary support tools.

Table 2. Comparison of Closed and Open Source

	Closed Source				Open Source	
	Proprietary	**Shareware**	**Freeware**	**Private**	**Commercial**	**Community**
Source Code Available	No or only with restrictions	No	No	No or only with restrictions	Yes, except for Proprietary Features	Yes
Binary Executables Available	Yes	Yes	Yes	Yes	Yes	Yes
No Restrictions on How Used	No	No	No	No	Yes	Yes
Unlimited Number of Users	No	No	No	No or only with restrictions	Yes, except for Proprietary Features	Yes
Right to Study How Software Works	No	No	No	No or only with restrictions	Yes, except for Proprietary Features	Yes
Right to Redistribute	No	No	Yes	No	Yes, except for Proprietary Features	Yes
Right to Modify	No	No	No	No or only with restrictions	Yes, except for Proprietary Features	Yes
Right to Fork	No	No	No	No	Yes, except for Proprietary Features	Yes

Another key difference is that all OSS is built using open standards. While most commercial proprietary software does support open standards, some vendors will include proprietary extensions, use

proprietary data formats, or use proprietary interfaces that create vendor lock-in. The compliance with open standards has several benefits including the avoidance of vendor lock-in and greater interoperability. Adherence to open standards allows an organization to select software based on criteria such as available support, ease of administration, and user features instead of price and compatibility with a vendor's proprietary solution.

In many situations the difference between the functionality offered by an OSS product and a proprietary product may not be significant. In other situations there can be significant differences in the capabilities and features between proprietary and OSS products. For some products that require highly skilled analysis and frequent updates, such as virus scanners, the commercial proprietary products may be a better solution. Other products that do not require frequent changes, such as web servers, the OSS products may be a better solution. The decision to select a proprietary or an OSS product should be conducted based upon the merits of the product, not just the up-front price.

VII. The cost of Open Source Software

The confusion around the word 'free' can be resolved with a simple phrase, TANSTAAFL, which means "There Ain't No Such Thing As A Free Lunch" [8]. Although OSS may be acquired at little or no cost, there are other costs the enterprise must be prepared to incur. Due to the very nature of OSS the lack of commercial support and the low up-front cost to acquire presents management challenges. Understanding and managing the indirect costs of OSS is one of the primary challenges.

Commercial support may be purchased for most OSS, for some OSS an organization must provide internal support or rely on community support. An example of this is the difference between the commercial distribution of Linux from Red Hat and its equivalent non-commercial distribution. Red Hat Enterprise Linux (RHEL) is a commercial Linux distribution built upon OSS with commercial support provided by Red Hat. CentOS is a community supported, non-commercial Linux distribution that is intended to be a free, non-commercial equivalent version of RHEL. An organization that decides to run CentOS instead of RHEL will have to provide almost all support internally instead of having the commercial support offered by Red Hat.

VIII. Governance

Any OSS an organization uses is an asset that must be managed the same as any other asset. Since OSS has some subtle and critical differences from other assets, even other software, it needs to be dealt with differently. Generally commercial OSS can be managed under an organization's existing Software Asset Management (SAM) program. This is because commercial OSS has a number of features similar to commercial proprietary software such as support or usage tracking requirements. Community supported OSS does not have support or usage tracking requirements so it does not fit a traditional SAM program.

A. The OSS Portfolio

Organizations should establish an OSS Portfolio and formally appoint an OSS Portfolio Manager (PfM) and an OSS Champion. The OSS Portfolio would be responsible for approving, acquiring, evaluating, and managing OSS throughout the entire organization. The OSS Portfolio will be responsible for reporting up the governance chain the same as other equivalent departments or offices within the organization. It would also be the internal advocate for the use of OSS and educating organization personnel about OSS. The OSS Portfolio can use a modified version of the organization's existing SAM program. Any public release of source code, either enhancements to existing OSS projects or of an organization established and sponsored OSS project, is managed through the OSS Portfolio.

The OSS PfM is the single person in charge of and responsible for all aspects of OSS within the organization. The PfM will be part of the organization's technical department and be equivalent to other

portfolio or service department managers. In addition the OSS PfM will provide reports, information, and support to the organization's OSS Champion. The primary responsibility of the OSS PfM is to develop and enforce policies and procedures for the use of OSS that meets the organization goals and legal requirements. The OSS PfM would also be the primary point of contact between the organization and the OSS community and will be responsible for compliance with OSS licenses and contributions of funds and resources to the OSS community.

The OSS Champion should be a Senior Executive at the 'C' level, or a member of the Board of Directors. The OSS Champion will provide strategic guidance and oversight to the organization's use and support of OSS. This is different than the OSS PfM who is responsible for day-to-day management of OSS. The PfM and Champion will need to coordinate closely to define and refine the organization's internal OSS policies and community engagement.

B. Acquiring

Unlike closed source software most OSS can be downloaded directly from the Internet with little or no oversight by the organization. The OSS portfolio should establish a central repository of approved OSS that users can download from instead of directly from the Internet. This will allow the OSS PfM to track usage; update the software as needed; and to thoroughly evaluate the software for quality, security, and license acceptability prior to internal use. The OSS PfM will need to develop and enforce policies and procedures for acquiring, evaluating, and approving OSS for internal use. The use of commercial OSS can reduce or eliminate the need for extensive testing at the time of acquisition.

The OSS Portfolio should acquire any OSS directly from a known trusted source and follow the best practices for confirming the integrity of the downloaded software. The source code should be scanned using Static Code Analysis (SCA) tools to find security issues. Once the SCA is completed the code should be compiled into a binary form and subjected to the organization's standard security testing process (such as vulnerability scanning, Dynamic Code Assessment (DCA), fuzz testing, and penetration testing).

An organization may determine that it is best to only distribute OSS it has compiled internally instead of the installation packages available from the developer. If the organization decides to only use internally compiled packages the personnel and resources to sustain this practice will need to be established and maintained. If the organization decides to use the installation packages provided by the OSS project the installation packages should be subjected to the organization's standard security testing process.

C. Managing

The first step in managing OSS within the organization is getting a comprehensive inventory of all OSS being used in the organization. This ranges from a complete Operating Systems (OS) such as Linux, to applications such as the Apache web server or the Eclipse Integrated Development Environment (IDE), to programming language interpreters such as Perl or Python, to programming frameworks such as Java Struts, and to software libraries such as OpenSSL.

Commercial and non-commercial OSS management products are available for tracking and managing OSS software, including software frameworks and libraries. Organizations should consider integrating these products into their OSS Portfolio governance process. If possible, existing tools for monitoring and managing software on servers, end user desktops, laptop, and mobile devices should be used to monitor for users downloading OSS directly from the Internet and being used without authorization.

Policies, guidelines, processes, and procedures for the use, tracking, updating, managing, and internal distribution of approved OSS need to be developed and enforced. The OSS Portfolio team will need to coordinate with the organization's groups that manage end user software, servers and other system operations, network infrastructure, and security to develop and implement the OSS policies. All of the

organization's employees and contractors working on site will need to be educated on the OSS policies and practices to ensure compliance with the least disruption and confusion.

D. Legal Liabilities and License Compliance

The following discussion is provided to give organizations an understanding of some of the legal issues concerning the use of OSS. This is not legal advice and should not be considered as legal guidance. Organizations should consult with legal counsel to understand how legal and licensing issues will impact their organization.

Legal liability is often raised as a concern that deters organizations from using OSS. There is the possibility of legal repercussions of using any software. The greatest risks most organizations are exposed to when running proprietary closed source software is failing to comply with license restrictions on the number of copies being used and the sharing of the software internally in violation of the vendor's license. These basic and potentially very expensive violations are not an issue with OSS. With OSS there are concerns about the risks of liability for patent violations, export law violations, and other legal liabilities. Using commercially supported OSS, and to a certain degree using foundation based projects, can reduce these risks. Commercially supported OSS can allow the organization to transfer these risks to the vendor. Foundation based projects are generally supported by large global corporations who wish to use the software globally. These foundations often have reviewed the legal issues concerning their software and may have taken steps to remediate many of these concerns. Generally independent OSS projects do not have the ability to conduct legal reviews or have done limited reviews. While there have been some instances of OSS projects infringing on patents or technology licenses it is a rare occurrence.

The organization will need to understand and develop legal guidance on the use of OSS internally and by third parties developing software for the organization under contract. This will require the OSS PfM to work closely with the organization's legal team to determine which OSS projects and licenses are acceptable and not acceptable for internal use. For example in the case of selecting licenses it may be determined that any OSS licensed under the GNU GPL version 2 is acceptable for internal use and development, but that software licensed under GPL version 3 requires a case-by-case review. A review by the OSS PfM and the legal team may determine that commercial and foundation-based OSS are acceptable without additional legal review but independent OSS projects need to be reviewed on a case-by-case basis. Conducting a legal review and developing clear legal guidelines for using OSS early in establishing the OSS governance program will significantly reduce the risk of exposure to legal liability.

In addition to developing internal policies and procedures for using OSS, organizations should develop policies and contract language about the use of OSS by third parties developing products for the organization. This should address the indemnification of the organizations from any OSS license infringements by the third party. This is similar in concept to indemnification of the organization from patent or trademark disputes that the third party may become involved in.

E. Reporting

As an organization establishes its OSS Portfolio it will need to develop measurements and metrics to gauge its progress and status. Some of the measurements and metrics used in the existing SAM program can be used without change, while others may need to be adapted for use with OSS. Reporting should follow the same policies and processes equivalent portfolios/departments/groups follow. In addition to reporting on the internal use of OSS the OSS PfM should track and report on community involvement.

F. To Give or Not to Give, That is the Question

In order to maintain compliance with some of the OSS licenses some changes to the source code need to be contributed back to the project and released under the same OSS license. Additionally as an organization uses OSS it may find that it needs or wants to fix a bug, make a feature enhancement, or

provide new functionality to a particular OSS project. The organization will need to determine its policy on contributing to OSS projects and the OSS PfM must be empowered to enact the policy. The organization should also have a clear policy on individual employees contributing their personal time and expertise to OSS projects.

The organization can also contribute to the OSS community by joining and supporting OSS organizations such as foundation-based OSS projects. Financial support of foundation-based OSS projects gives the organization a way to support the continued development of the project and potentially to participate in the governance of the project while sharing the costs with other organizations. The OSS PfM would be responsible for determining what projects and organizations to support, and what form that support should take.

IX. Summary and Conclusions

Open Source Software can offer organizations significant cost saving and flexibility. The unmanaged use of OSS can expose organizations to security risks, legal liabilities, and unforeseen costs. In order to achieve the full potential of OSS while minimizing the risks organizations need to develop and implement a program to integrate OSS into their governance process. Organizations need to understand and account for the differences between managing traditional commercial proprietary software and OSS. Establishing an OSS Portfolio that has the responsibility and authority to manage all OSS throughout the organization, to engage the Open Source community, and to develop and enforce the organization's OSS related policies and procedures is one method for integrating OSS into the organization's governance process. By fully integrating OSS into the governance process organizations will gain control over a portion of their enterprise computing base they currently have little or no visibility or control over.

REFERENCES

[1] Department of Defense Chief Information Officer. (2009, Oct 16). Clarifying Guidance Regarding Open Source Software (OS). Washington, DC, USA: The Department of Defense.

[2] The Open Source Definition, http://opensource.org/osd

[3] The Free Software Foundation, http://www.gnu.org/philosophy/free-sw.html

[4] Open Source Initative Open Source Definition compliant licenses, http://opensource.org/licenses

[5] U.S. Copyright Office, Report on Orphan Works at 2 (2006), http://www.copyright.gov/orphan/orphan-report.pdf ("Orphan Works Report").

[6] Committee on National Security Systems Instruction (CNSSI) No. 4009 Committee on National Security Systems Glossary, https://www.cnss.gov/

[7] Wasserman, A.I., Community and Commercial Strategies in Open Source Software, http://repository.cmu.edu/silicon_valley/172/

[8] Heinlein, R. A. (1966). The Moon is a Harsh Mistriss. New York City, NY: GP Putnum.