Free and Open Source Software

Mohammad AlMarzouq
Clemson University, USA

Guang Rong
Clemson University, USA

Varun Grover
Clemson University, USA

INTRODUCTION

Free and open source software (F/OSS) is emerging as a promising alternative to proprietary software. The interest in F/OSS solutions is growing as firms realize that it could help reduce IT expenditures. Unfortunately, despite the heightened interest, F/OSS solutions remain misunderstood, and a number of myths regarding this approach still prevail.

F/OSS has been described as simply software, or even software specific for the Linux operating system, with hardly any reliable support available. Some have considered it a silver bullet solution that will always create superior quality software at lower or no cost (Wheatley, 2004).

The purpose of this article is to demystify these misconceptions surrounding F/OSS and provide an understanding of its basic concepts. Then, based on these concepts, we try to illustrate how companies can benefit from F/OSS. Our hope is that this article would assist interested observers to better understand F/OSS and help managers make more informed decisions regarding F/OSS solutions.

BACKGROUND

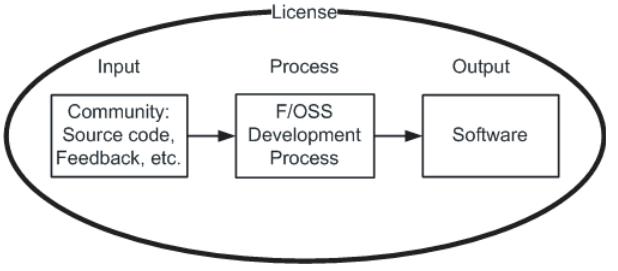
When computers first originated, the norm in most academic and corporate labs was to freely exchange programs and ideas between programmers. This was the early form of F/OSS. This norm changed when IBM unbundled its software and hardware in the 1970s. This move created a market and value for software. In order to preserve this newly found software value, software producers restricted user access to human readable source code in order to protect software secrets (Glass, 2004). This meant that users could not modify the software that they owned and more importantly, restricted the free flow of ideas. This did not fit well with many programmers, most notable was Richard Stallman from MIT's Artificial Intelligence Lab. Stallman believed strongly in the freedom of the user to use his software and created the Free Software Foundation (FSF) in 1985 to promote the development and use of free software. The following success of projects using the free software development model, such as Apache and Linux, inspired Eric Raymond to write his seminal piece *The Cathedral and the Bazaar* in 1997 that brought the attention of the corporate world to the free software development model (Raymond, 1999a). In a brain storming session on February 3, 1998 Todd Anderson, Christine Peterson (of the Foresight Institute), John "maddog" Hall and Larry Augustin (both of Linux International), Sam Ockman (of the Silicon Valley Linux User's Group), and Eric Raymond, agreed on the need for a marketing campaign to win the support of fortune 500 companies to ensure the long-term survival of the movement. The participants saw the need to use a term other than "free" which they figured would hurt the movement's chances of gaining support from the corporate world because of its ambiguous meaning. As a result of this session, the term *open-source* was coined by Christine Peterson and the Open Source Initiative Organization (OSI) was established ("History of the OSI," n.d.).

Richard Stallman of the FSF opposed this movement because in his opinion the term open-source was not pure enough. So the FSF and OSI remained separate movements promoting similar practical principles, but with different philosophies and beliefs. FSF decided to be vocal about its beliefs and maintained the *free* software label. They believed that the user's "freedom" is a priority, an end of itself, and they should be able to do whatever they want with their software. OSI on the other hand did not explicitly express the user's right for software freedom, but promoted it as a means of producing better software. The end is to get the corporate world to buy into this concept. Because of the coexistence of both philosophies, we refer to the group of software that adheres to the OSI or FSF principles as free and open source software (F/OSS). [1]

F/OSS IPO MODEL

The F/OSS development system can be conceptualized as an input-process-output (IPO) system, with the license as the boundary, the community as the input provider, the F/OSS

Figure 1. F/OSS IPO system



development methodologies as the process, and the software as the output (see Figure 1). So we would expect that all four components to have implications on the quality of the final product (software).

The License

The F/OSS license acts as the boundary that identifies the system as an F/OSS system. It specifies the terms by which the software is to be used and distributed. It serves as a governing mechanism that enforces the norms of the F/OSS community and provides motivation for programmers by protecting their efforts from appropriation (Bonaccorsi & Rossi, 2003).

There are numerous F/OSS licenses available that all maintain the openness and free redistribution of the source code.[2] The difference between the licenses reflects the philosophical differences between the FSF and OSI on how to advance the F/OSS projects. There are two principles that observers should be aware of (Lee, 1999):

- **The copyleft principle:** Software derived/revised from original F/OSS source code must remain F/OSS, and privatization of any part or whole of the program is prohibited.
- **The GPL compatibility principle:**[3] Licensed F/OSS cannot be mixed with proprietary source code.

Both the FSF and OSI have very similar criteria in qualifying licenses and there is a great deal of overlap between the approved licenses of both organizations. The fundamental difference between the two lies in the underlying philosophy. FSF recommends more open licenses that enforce both the GPL compatibility and copyleft principles, while the OSI is less stringent on this matter. Generally, an F/OSS license must allow programmers to access, modify, and redistribute the source code. F/OSS licenses do not prevent a software producer from demanding a distribution fee for his product. F/OSS licenses however prevent the software producer from placing restrictions on how the software should be used or redistributed after it is in the hands of the users ("*Selling Free Software,*" n.d.).

The Community

The community consists of all the developers and users of the F/OSS. The community and all their contributions are conceptualized as the input to the IPO system, which includes source code, documentation, and feedback (i.e., bug reports, support requests, and feature requests) (Raymond, 1999b).

The growth of the community ensures the ongoing survival of the F/OSS project and further improvement of the product. The advancement of the projects and community is dependent on the members who have the motivation and the ability to contribute. The most active of the community contributors are known as the *core*. The core is responsible for the majority of source code development. They also have the most control over the features and design of the software product. Occasional source code contributors are known as *co-developers*. They contribute by modifying or reviewing code or submitting bug fixes in addition to feedback. But the majority of the community members are the *users* who do not contribute with code submissions. Depending on the level of feedback, users can be active by providing some feedback, or passive, by providing none (Crowston & Howison, 2005).

THE DEVELOPMENT PROCESS

F/OSS development communities do not seem to adopt or practice traditional software development processes (e.g., the waterfall model). Many F/OSS projects begin with a prototype with predefined requirements developed from scratch or based on existent older product (Scacchi, 2002). Then, this early version incrementally evolves through rapid development iterations from the community, while concurrently managing as many designing, building, and testing activities as possible. Five main steps were identified for this approach: (1) code submission, (2) peer review, (3) precommit test, (4) development release and parallel debugging, and (5) production release (Jorgensen, 2001).

The F/OSS development process is an iterative process with feedback loops in every stage. The source code is constantly updated to meet the dynamic requirements that change along with the needs of the users (see Figure 2). These requirements are updated based on user feedback.

The Software

A continuous output of the F/OSS system is the software, which demonstrates some unique benefits when compared to proprietary software, such as: