Chapter 1.5
Open Source Software:
Strengths and Weaknesses

Zippy Erlich
The Open University of Israel, Israel

Reuven Aviv
The Open University of Israel, Israel

ABSTRACT

The philosophy underlying open source software (OSS) is enabling programmers to freely access the software source by distributing the software source code, thus allowing them to use the software for any purpose, to adapt and modify it, and redistribute the original or the modified source for further use, modification, and redistribution. The modifications, which include fixing bugs and improving the source, evolve the software. This evolutionary process can produce better software than the traditional proprietary software, in which the source is open only to a very few programmers and is closed to everybody else who blindly use it but cannot change or modify it. The idea of open source software arose about 20 years ago and in recent years is breaking out into the educational, commercial, and governmental world. It offers many opportunities when implemented appropriately. The chapter will present a detailed definition of open source software, its philosophy, its operating principles and rules, and its strengths and weaknesses in comparison to proprietary software. A better understanding of the philosophy underlying open source software will motivate programmers to utilize the opportunities it offers and implement it appropriately.

INTRODUCTION

Open source software (OSS) has attracted substantial attention in recent years and continues to grow and evolve. The philosophy underlying OSS is to allow users free access to, and use of, software source code, which can then be adapted, modified, and redistributed in its original or modified form for further use, modification, and redistribution. OSS is a revolutionary software development methodology (Eunice, 1998) that involves developers in many locations throughout

the world who share code in order to develop and refine programs. They fix bugs, adapt and improve the program, and then redistribute the software, which thus evolves. Advocates of OSS are quick to point to the superiority of this approach to software development. Some well-established software development companies, however, view OSS as a threat (AlMarzouq, Zheng, Rong, & Grover, 2005).

Both the quality and scope of OSS are growing at an increasing rate. There are already free alternatives to many of the basic software tools, utilities, and applications, for example, the free Linux operating system (Linux Online, 2006), the Apache Web server (Apache Software Foundation, 2006; Mockus, Fielding, & Herbsleb, 2000), and the Sendmail mail server (Sendmail Consortium, 2006). With the constant improvement of OSS packages, there are research projects, even complex ones, that entirely rely on OSS (Zaritski, 2003). This opens new research and educational opportunities for installations and organizations with low software budgets.

Incremental development and the continuity of projects over long periods of time are distinctive features of OSS development. The software development processes of large OSS projects are diverse in their form and practice. Some OSS begins with releasing a minimal functional code that is distributed for further additions, modification, and improvement by other developers, as well as by its original authors, based on feedback from other developers and users. However, open source projects do not usually start from scratch (Lerner & Tirole, 2001). The most successful OSS projects, like Linux and Apache, are largely based on software provided by academic and research institutions. In recent years, more and more OSS has been derived from original software provided by for-profit companies.

A large potential-user community is not enough to make an OSS project successful. It requires dedicated developers. In Raymond's (1998) words, "The best OSS projects are those that scratch the itch of those who know how to code." For example, the very successful Linux project attracted developers who had a direct interest in improving an operating system for their own use. Similarly, webmaster developers contributed to the development of the Apache Web server project.

Despite the characterization of the OSS approach as ad hoc and chaotic, OSS projects appear, in many cases, to be highly organized, with tool support that focuses on enhancing human collaboration, creativity, skill, and learning (Lawrie & Gacek, 2002). The good initial structural design of an OSS project is the key to its success. A well-modularized design allows contributors to carve off chunks on which they can work. In addition, the adoption of utility tools and the use of already existing OSS components are necessary if an OSS project is to succeed.

The growing interest of commercial organizations in developing and exploiting OSS has led to an increased research focus on the business-model aspects of the OSS phenomenon. There are a number of business models for OSS, all of which assume the absence of traditional software licensing fees (Hecker, 2000). The economics of OSS projects is different from that of proprietary projects (Lerner & Tirole, 2002). Models of effort and cost estimation in the development of projects involving OSS are needed (Asundi, 2005).

In the past, most OSS applications were not sufficiently user friendly and intuitive, and only very knowledgeable users could adapt the software to their needs. Although the use of OSS is growing, OSS is still mainly used by technically sophisticated users, and the majority of average computer users use standard commercial proprietary software (Lerner & Tirole, 2002). The characteristics of open source development influence OSS usability (Behlendorf, 1999; Nichols, Thomson, & Yeates, 2001; Raymond, 1999), which is often regarded as one of the reasons for its limited use. In recent years, the open source community has shown increased awareness of