Chapter 7.2
Software Quality and the Open Source Process

Sameer Verma
San Francisco State University, USA

ABSTRACT

This chapter introduces the open source software development process from a software quality perspective. It uses the attributes of software quality in a formal model and attempts to map them onto the principles of the open source process. Many stages of the open source process appear to have an ad-hoc approach. Although open source is not considered to be a formal methodology for software development, it has resulted in the development of very high quality software, both in the consumer and in the enterprise space. In this chapter, we hope to understand the open source process itself, and apply it to other methodologies in order to achieve better software quality. Additionally, this chapter will help in understanding the "Wild West" nature of open source and what it may hold for us in the future.

INTRODUCTION

The concept of quality is an abstract one. While quantity can be specified in standard terms such as a gram or a gallon, quality is usually a relative term. It is relative to the assessment of the product or process, as perceived by an individual or an organization (Barbacci et al., 1995). Quality is sometimes defined as compliance *to* a standard (Perry, 1991). It seems that the implications of quality vary from one task to another, and so does its assessment. However, generally speaking, we can agree on the direction it takes when the quality of an entity improves or degrades. In the case of software, the quality of software can be assessed by its characteristics. Several models exist that either measure software quality using a quantitative surrogate, or in terms of attribute sets. In this chapter, we use a model (Bass et al., 2000), where the quality of software is assessed by its attributes. These attributes include per-

formance, security, modifiability, reliability, and usability of a particular system. We will explore these attributes, the challenges they pose to the open source development process and how the open source community measures up to these challenges.

BACKGROUND

The importance of information systems and technology is well-established in our society, both in personal and professional space. It is difficult to imagine a society without access to information. Software forms an important cog of that system. While hardware such as desktops, laptops, and PDAs provide a platform for implementing computing power, software is the flexible component that is responsible for expression of innovation, creativity, and productivity (Lessig, 2005). Ranging from word-processing to number-crunching, the quality of software influences and impacts our lives in many different ways. It is no surprise that software quality has become the subject of many studies (Halloran & Scherlis, 2002).

Software Quality

While the quantitative side of software can be measured and optimized in terms of its capability to solve mathematical formulations and render graphics (Wong & Gokhale, 2005), it is the qualitative perception that matters to the end user and the organization (Bass et al., 2000). Quality is also harder to assess, so it is often ignored or replaced by quantitative measurements such as lines of code. Obviously, more lines of code do not always imply better quality (Samoladas et al., 2004).

In this chapter, we will use a model that uses five attributes, namely, performance, security, modifiability, reliability, and usability, to assess the quality of software. Performance involves adjusting and allocating resources to meet system timing requirements (Bass et al., 2000). Security can be described as freedom from danger, that is, safety. It may also be viewed as protection of system data against unauthorized disclosure, modification, or destruction (Barbacci et al., 1995). Modifiability is the ability of a system to be changed after it has been deployed. Requests can reflect changes in functions, platform, or operating environment (Bass et al., 2000). Reliability of a system is the measure of the system's ability to keep operating over time (Barbacci et al., 1995). Usability is a basic separation of command from data. It relies on explicit models for task, user, and system (Bass et al., 2001). These attributes put together can assist in assessing the quality of a software project. Of course, these attributes apply to all kinds of software, whether open source or proprietary.

Open Source Process

Next, let us examine the open source process. The purpose of looking at open source as a process, as opposed to a product, is that the open source ideology permeates well beyond software production (DiBona et al., 1999). The proponents of open source believe in a philosophy of open source more so than simply the software. They look upon software as an enabler of change in a society that increasingly relies on information technology for its survival and growth (Lessig, 2005). The following sections are by no means exhaustive, but should provide a substantial background to understanding the tenets of quality in the open source context.

Open source software (OSS) is largely developed by freelance programmers, who create freely distributed source code by collaborating and communicating over the Internet (Moody, 2001; Raymond, 1999; Sharma et al., 2002). A small, but significant proportion of the software is also developed in software companies such as IBM, Sun Microsystems, and Intel. Open source software is generally fashioned in what is often