Chapter I

Do Not Check in on Red: Control Meets Anarchy in Two Open Source Projects

Jesper Holck, Copenhagen Business School, Denmark
Niels Jørgensen, Roskilde University, Denmark

ABSTRACT

For two Free/Open Source Software projects, Mozilla and FreeBSD, we describe the central elements in the software development processes: the technological infrastructure, the work organization, and the software process models. For each of these elements we discuss how the projects try to find an optimal balance between control (supposedly necessary for producing high-quality software) and anarchy (supposedly necessary for attracting and keeping voluntary developers). Several important considerations are identified: most importantly, control of access to bug-tracking systems and source code repositories, quality control of both individual contributions and production releases, the importance of the development branch, and control of developers’ prioritization of work tasks and availability. The results show that the two projects, even though they produce very different kinds of software (a web-browser suite and an operating system), are similar in many respects. However, they also show how difficult the balance between anarchy and control may be and that it is likely to shift over time.
INTRODUCTION

The statement “Do Not Check in on Red” can be found in red, capital letters on the website of Mozilla, a well-known F/OSS (Free/Open Source Software) web browser. Mozilla, like many F/OSS and some commercial projects, employs the principle of continuous integration, where a number of developers individually and in parallel add source code to a central repository. The statement is a request for developers not to check in new changes to the repository if severe errors have been found in the present source code, a situation announced with unmistakable red boxes on the website. Taken literally, the statement would have the unfortunate consequence of also barring possible fixes to these errors, effectively blocking all further progress; rather, it should be interpreted as a strong request for developers to only check in changes necessary to fix the detected errors while the tree is red. In this way, the statement “Do Not Check in on Red” conveys a fundamental tension in the project: allowing a continuous flow of changes and improvements to the software on one hand, and assuring quality by minimizing the problems resulting from deficient contributions on the other.

As discussed by O’Mahony (2003), it is generally assumed that F/OSS developers (“hackers”) don’t “embrace centralized modes of governance” and are “less likely to welcome formal organizing mechanisms,” instead believing “in the value of challenging work, technical autonomy, self-management, and freedom from a positional basis of power.” These assumptions indicate that F/OSS developers would prefer rather loosely controlled projects with a flat hierarchy, relying on individual autonomy, tacit norms, and self-organization rather than commands, control, and explicit rules. This is important for a F/OSS project facing the challenge of attracting and retaining voluntary, competent developers; even though some developers are paid for their work, typically this salary does not come from the project as such, but from companies or organizations volunteering for the project.

It is, however, also generally assumed that software project success is dependent on diligent project management, “planning, organizing, directing, and controlling … company resources … to complete specific goals and objectives” (Kerzner, 1989, as cited in Jurison, 1999), and tight control of the software development process is a key element in Software Process Improvement (Paulk, Curtis, Chrissis, & Weber, 1993).

The theme of this chapter will be the tension between these two, apparently conflicting demands on a F/OSS project: how are successful F/OSS projects able to find a reasonable balance between anarchy (supposedly necessary for attracting and keeping voluntary developers) and control (supposedly necessary for the effective production of high-quality software). In this context we will construe anarchism positively as “holding all forms of government authority to be unnecessary and undesirable and advocating a society based on voluntary cooperation and free association of individuals and groups” rather than “a state of lawlessness or political disorder due to the absence of governmental authority” (Anarchism, 2003).