
Alan W. Brown and Grady Booch

Rational Software,
8383 158th Avenue NE, Redmond WA 98052
abrown@rational.com, egb@rational.com

Abstract. One of the most intriguing ways that commercial developers of software can become more efficient is to reuse not only software but also best practices from the open-source movement. The open-source movement encompasses a wide collection of ideas, knowledge, techniques, and solutions. Commercial software vendors have an opportunity to both learn from the open-source community, as well as leverage that knowledge for the benefit of its commercial clients. This paper looks at a number of the characteristics of the open-source movement, offers a categorization of open-source dimensions, and provides an analysis of the opportunities available to commercial software vendors when applying the lessons from the open-source movement.

Introduction

Open-source represents one of the most interesting and influential trends in the software industry over the past decade. Today, many organizations are looking toward open-source as a way to provide greater flexibility in their development practices, jump-start their development efforts by reusing existing code, and provide access to a much broader market of users [1].

However, what is widely referred to as the “open-source movement” is in reality a plethora of relative independent initiatives representing a variety of technology innovations and approaches. Elements of the movement’s best practices include broadening the notion of a project team, frequent release of new software builds, greater collaboration across geographically dispersed teams enabled by the Internet, and creation of publicly available source code for adaptation and reuse. While a few common threads and community voices have emerged (e.g., Richard Stallman, Eric Raymond, Linus Torvalds, and Tim O’Reilly), the “community” remains essentially a collection of different activities galvanized by a recognition of the benefits of broader, more open access to the software development process, and to the results of that process.

A number of these innovations and approaches have a direct impact on commercial software vendors. There are some aspects on which these vendors can build, others that are in direct competition, and some that offer partnering opportunities in the spirit of “cooperation”.

© Springer-Verlag Berlin Heidelberg 2002
In this paper we examine the relationship between commercial software vendors and open-source. While there is a great deal that could be said, here we concentrate on a few of the major threads with an eye toward the business and economic aspects of this relationship. Many more aspects of the technical and legal relationships require further exploration. Additionally, this paper assumes that the reader does not require too much background or motivation for commercial software vendors’ interest in open-source.

Many commercial software perspectives could be adopted in this analysis of open-source, each with its own peculiarities and concerns. However, in this paper the authors assume the perspective of one particular class of commercial software vendor – vendors of software development methods and tools. This is the community with which the authors are most familiar, and in many respects this view is representative of the broader set of perspectives. Hence, the term “commercial software vendor” can be interpreted to refer to this narrow definition, although with obvious application in most cases to the broader commercial software vendor community.

Reuse and Open-Source

It can be argued that the open-source approach is the software industry’s most successful form of large-scale software reuse. Open-source software offers the most astounding range of reusable assets for any software project. Open-source software is available for virtually all activities, runs on every platform, and can be used in almost every business domain for which software is written. The open-source movement is predicated on reuse, in many cases with licensing that insists that derived products are themselves released into the public domain.

Two kinds of open-source software reuse can be observed. In the first case, the reuse of open-source software is a planned strategy to augment a popular free product, taking advantage of the wide informal network of open-source developers and users and thus minimizing internal development resources. Often these efforts focus around highly popular open-source products such as Linux, Apache, or StarOffice. Similarly, efforts may be specifically designed in this way through industry consortia. An interesting example of this is the group of commercial software development organizations who have committed to using the Eclipse platform as the basis for software development tools.\footnote{The Eclipse consortium includes IBM, WebGain, Borland, TogetherSoft, Merant and Rational Software. See \url{www.eclipse.org} for details.} Eclipse provides an open-source tools framework with an extensible plug-in architecture. As a result, commercial tool vendors do not have to re-implement common functions for managing menus and toolbars, file explorers, and drawing surface manipulation. Rather, they can concentrate on extending this framework with their own value-added domain-specific behavior.

In the second case, reuse of open source is more ad hoc. Particularly in terms of the infrastructure of software development, there are many open-source software products that are so pervasive that they have become part of the fabric of most development organizations. Examples of such products include EMACS, the GNU compilers, build tools such as Ant, and scripting languages such as Perl, PHP, and TCL.

\footnote{The Eclipse consortium includes IBM, WebGain, Borland, TogetherSoft, Merant and Rational Software. See \url{www.eclipse.org} for details.}