Chapter 1
Software Licenses, Open Source Components, and Open Architectures

Thomas A. Alspaugh
University of California Irvine, USA

Hazeline U. Asuncion
University of Washington Bothell, USA

Walt Scacchi
University of California Irvine, USA

ABSTRACT

A substantial number of enterprises and independent software vendors are adopting a strategy in which software-intensive systems are developed with an open architecture (OA) that may contain open source software (OSS) components or components with open APIs. The emerging challenge is to realize the benefits of openness when components are subject to different copyright or property licenses. In this chapter, the authors identify key properties of OSS licenses, present a license analysis scheme to identify license conflicts arising from composed software elements, and apply it to provide guidance for software architectural design choices whose goal is to enable specific licensed component configurations. The scheme has been implemented in an operational environment and demonstrates a practical, automated solution to the problem of determining overall rights and obligations for alternative OAs as a technique for aligning such architectures with enterprise strategies supporting open systems.

INTRODUCTION

A substantial number of enterprises and independent software vendors are adopting a strategy in which software-intensive systems are developed with open source software (OSS) components or components with open APIs. It has been common for both independent or corporate-sponsored OSS projects to require that developers contribute their work under conditions that ensure the project can license its products under a specific OSS license. For example, the Apache Contributor License Agreement grants enough rights to the Apache Software Foundation for the foundation to license

DOI: 10.4018/978-1-4666-7230-7.ch001

the resulting systems under the Apache License. This sort of license configuration, in which the rights to a system's components are homogeneously granted and the system has a well-defined OSS license, was the norm and continues to this day.

However, we more and more commonly see a different enterprise software configuration, in which the components of an enterprise system do not have the same license. The resulting system may not have any recognized OSS license at all–in fact, our research indicates this is the most likely outcome–but instead, if all goes well in its design, there will be enough rights available in the system so that it can be used and distributed, and perhaps modified by others and sub-licensed, if the corresponding obligations are met (Alspaugh, Asuncion, & Scacchi, 2009). These obligations are likely to differ for components with different licenses; a BSD (Berkeley Software Distribution) licensed component must preserve its copyright notices when made part of the system, for example, while the source code for a modified component covered by MPL (the Mozilla Public License) must be made public, and a component with a reciprocal license such as the Free Software Foundation's GPL (General Public License) might carry the obligation to distribute the source code of that component but also of other components that constitute "a whole which is a work based on" the GPL'd component. The obligations may conflict, as when a GPL'd component's reciprocal obligation to publish source code of other components is combined with a proprietary license's prohibition of publishing source code, in which case there may be no rights available for the system as a whole, not even the right of use, because the obligations of the licenses that would permit use of its components cannot simultaneously be met.

The central problem we examine and explain in this chapter is to identify principles of software architecture and software licenses that facilitate or inhibit success of the OA strategy when OSS and other software components with open APIs are employed. This is the knowledge we seek to develop and deliver. Without such knowledge, it is unlikely that an OA that is clean, robust, transparent, and extensible can be readily produced. On a broader scale, this chapter seeks to explore and answer the following kinds of research questions:

- What license applies to an OA enterprise system composed of software components that are subject to different licenses?
- How do alternative OSS licenses facilitate or inhibit the development of OA systems for an enterprise?
- How should software license constraints be specified so it is possible for an enterprise to automatically determine the overall set of rights and obligations associated with a configured enterprise software system architecture?

This chapter may help establish a foundation for how to analyze and evaluate dependencies that might arise when seeking to develop software systems that embody an OA when different types of software components or software licenses are being considered for integration into an overall enterprise system configuration.

In the remainder of this chapter, we examine software licensing constraints. This is followed by an analysis of how these constraints can interact in order to determine the overall license constraints applicable to the configured system architecture. Next, we describe an operational environment that demonstrates automatic determination of license constraints associated with a configured system architecture, and thus offers a solution to the problem we face. We close with a discussion of some issues raised by our work.

BACKGROUND

There is little explicit guidance or reliance on systematic empirical studies for how best to develop, deploy, and sustain complex software