

A Three-Layer View Model of OSS: Toward Understanding of Diversity of OSS

Toshihiko Yamakami
CTO Office, ACCESS
Nakase 1-10-2, Mihama-ku, Chiba-shi
Chiba, Japan 261-0023
Email: Toshihiko.Yamakami@access-company.com

Abstract—Open source software (OSS) is becoming increasingly visible in the software industry. The entire social and commercial fabric supporting the creation of software is changing. OSS is a global distributed community-oriented model of software development with OSS-oriented licenses. This characteristic of OSS has created a wide range of diversity and heterogeneity in OSS. In order to achieve management, evaluation, measure readiness, and leveraging of ecosystems, it is important to decompose the complexity, diversity, and heterogeneity of OSS. The author analyzes seven aspects of OSS in order to help construct a new model to achieve this goal. Then, the author proposes a three-layer view model of OSS in order to position the seven aspects for in-depth analysis of OSS.

Keywords — open source software; view model; three-layer structure; analysis of OSS; diversity and heterogeneity

I. INTRODUCTION

OSS has penetrated into the software industry. The quality of OSS according to commercial standards is high enough for enterprise deployment. The adoption of OSS by the enterprise is widely visible. It is widely accepted that many OSS projects are first rate citizens in the software industry. At the same time, however, there is a wide range of diversity in OSS.

There are hobby-based voluntary communities, social reform communities in developing countries, full-time OSS-based business model-oriented startup company employee-driven communities, and academic non-profit communities. The OSS concept covers software artifacts including code, licenses, development best practices, promotion, diffusion, innovation, social justice, ethics, philosophy, social movement, community, culture, governance, and organizational engagement.

Starting from the simple idea of publishing source code to the general public, OSS has come to include a wide range of different aspects of computing after more than a decade, thanks to advances in computer and communication technologies that foster mature software artifacts that support OSS.

The OSS stack is now reasonably complete enough to provide reasonable IT solutions using OSS. There is an increasing number of large-scale OSS software. Some platform software code, such as Symbian, has reached a level of 40 million lines of code.

It is increasingly recognized that the term OSS has been interpreted by a wide range of different meanings in different

communities and projects. The norms, goals, and values of OSS show a high degree of diversity. The author has come to recognize the need to provide a general framework of OSS, for both research and practical purposes. Updegrove joked that when there are two people talking about OSS, there are three different meanings [1].

In order to identify the position of an OSS project within the wide spectrum of OSS, the author discusses seven aspects of OSS. Then, the author proposes a three-layer view model of OSS, to help locate the positions of these seven aspects.

II. PURPOSE AND RELATED WORKS

A. Purpose of Research

To cope with the increasing diversity and heterogeneity of OSS, it is important to provide a framework that properly decomposes the complexity, diversity, and heterogeneity of OSS.

The aim of this paper is to identify a layered view model of OSS in order to gain an in-depth understanding of OSS.

B. Related Works

OSS was separated from free software in the late 1990's in order to revisit the commercial issues of using OSS.

Raymond discussed open source from the business model perspective in this famous open source work series [2].

OSS has continued to evolve. Watson presented the second generation of OSS, or professional OSS [3] in contrast to three types of first generation OSS, such as community OSS, sponsored OSS, and corporate distribution.

Letellier discussed the third generation of OSS [4]. From his description, the first generation occurred in the early days, when it was mainly a group of loosely connected individuals (roughly before 1990), the second generation appeared with the worldwide web at a time with large-scale communities federated by associations of individuals (e.g. the Apache Software Foundation), and the third generation is the gathering of legal entities. The third generation goes beyond the level of single projects in terms of cross-project collaboration. In the third generation, the parties involved are legal entities, not individuals, and requires a new model of governance.

Open innovation around OSS was discussed by Alexy [5]. OSS represents a typical example of open and distributed innovation [6].

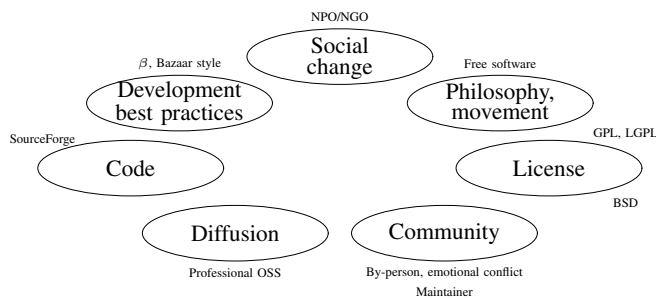


Fig. 1. Seven views of OSS.

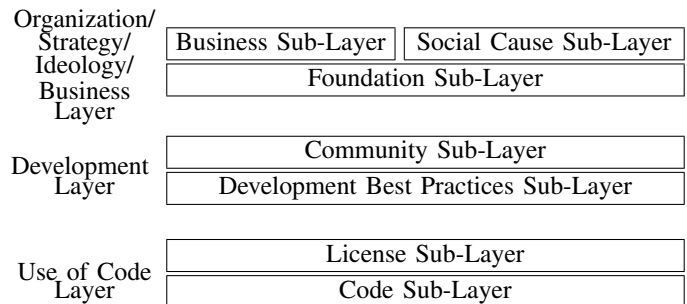


Fig. 2. Three-Layer View Model of OSS

There are several examples of emerging foundation engineering utilizing open source software in mobile platform software [7] [8] [9]. This rise of foundation engineering has caught the attention of industry worldwide.

As OSS has evolved, the diversity of OSS has also increased. Chopra discussed OSS from the viewpoint of information justice [10]. Berry discussed OSS from the viewpoint of democracy and public participation. This is only a part of the diversity found in OSS currently.

The originality of this paper lies in its classification of a wide range of diversity within a view model that captures the complexity of OSS management.

III. 7-ASPECT VIEW MODEL OF OSS

As OSS has penetrated the software industry, it has become widely accepted as a software development practice. The maturity of tools, infrastructure for communication, and project management has increased to a level that allows for global distributed collaborative development of large-scale software.

The skills and experience in the development of OSS are proliferating and leveraging the creation of new OSS projects for different roles, goals, and communities.

The globalization and advances in computer and communication technologies facilitate even more complicated OSS projects. The attitudes towards OSS and social acceptance of OSS have improved during the evolution of OSS.

This increased acceptance further exposes the different aspects of OSS.

The seven views of OSS are illustrated in Fig. 1. These seven aspects represent the different functions that OSS performs. As OSS has matured, each aspect has independently evolved to leverage complicated relationships among these aspects. The autonomous distributed nature of OSS has harnessed the diversity and heterogeneity of OSS.

IV. THREE-LAYER VIEW MODEL OF OSS

A. Three-Layer View

It is visible that some of the seven aspects are heterogeneous. It is also recognizable that some of them are tightly coupled.

For example, the promotional aspect used in a professional OSS company using a dual license to generate revenue streams is very different from public participation in social-welfare and governmental software with grass roots efforts attempting

to improve social welfare. Even within each aspect, there is a wide spectrum. In the license domain, free software and dual licenses including OSS licenses and commercial license are different. In the community domain, a community with full-time employees who have corporate sponsorship and a volunteer-centered community are different. Grass-roots communities and well-organized industrial consortia are different. It is also true that multiple aspects are interrelated. The GPL license is the heart of the GNU Foundation as well as the philosophy "freedom of software". These characteristics are closely interrelated.

In order to decompose this complexity, it is necessary to:

- clearly identify the seven different aspects, and
- identify the general relationships among these seven aspects.

In order to satisfy these requirements, the author classifies these aspects into 3 groups that describe a code-related perspective, a community-related perspective, and a high-level governance perspective. According to this classification, the author proposes a three-layer view model of OSS, depicted in Fig. 2.

Each sub layer is described in Table I. This covers the seven different aspects of OSS mentioned above.

B. Use of Code Layer

The use of the code layer represents the basic artifact of OSS, code and licenses. It is transparent to development best practices and communities above this layer, because code can be utilized by different communities and different best practices, barring any licensing restrictions.

This is the layer that deals with deliverables and the legal terms associated with them.

C. Development Layer

The development layer covers communities and development best practices. These two components also cover tools, and the development process, as well as roadmaps. This represents the development process, orthogonal to the layer above of this layer.

This is the layer that deals with day-to-day operations.

D. Organization, Strategy, Ideology, and Business Layer

This layer covers the higher social factors that drive software development. It is separate from the underlying development

TABLE I
SUB LAYER DESCRIPTION

Layer	Sub-layer	Description
Organization/ Strategy/ Ideology/ Business	Social Cause	Social change/reform by NGO/NPO in the developing countries. Activists. Social change by distributing OSS or open standards, enabling social reforms. Democracy pursuit by publicly available source code.
	Business	Professional OSS. Fulltime employed, 100 % company-owned software, commonly provided by dual license (OSS license and commercial license).
	Foundation	Non-business organizational structures that foster OSS development. Industrial foundations such as Eclipse Foundation and Symbian Foundation. Social movement such as Free Software Foundation.
Development	Community	Community fostering OSS projects. Social ties, participation, encouragement, expectation, engagement, trust, and hierarchy in OSS projects Leadership, control and conflict resolution. Community Cultures, norms, guidelines, and policies.
	Development Best Practices	OSS-based software development. Bazaar style development. Beta-released user-engaged software development with publicly available source code. Use of specific technical infrastructure chosen by the community.
Use of Code	Technical Information	Technical information for source codes. Software artifacts surrounding code.
	License	OSS licenses.
	Code	Code shared by OSS licenses. Software artifacts.

process and code layers because this represents the higher layer governance and motivation that drives each development project, which is important in persistent large-scale projects. This layer may be missing from OSS projects in some cases, especially at an early stage. It should be noted that this layer is the driving layer in some cases, such as ideological movements, social reforms or business development based on OSS. This is the layer that covers the motivation and management strategy of OSS. It is gaining increasing importance in the current landscape of OSS. In addition, the growth in the amount of OSS code increases the weight of this layer. When the amount grows to a certain limit, it requires coordination at a higher layer for large projects, whether they are OSS-based or not.

Corporate involvement is an important factor that covers foundations in many well established OSS projects. Foundations are not just for corporate involvement, but also for philosophical, ideological, or grass-rooted efforts for social changes such as NGO/NPO.

The business sub-layer covers a wide range of business model engineering for creating revenue streams in either directly related or indirectly related OSS projects. This business sub-layer represents the commitment of full-time employees when there is corporate involvement.

Social changes are driven in developing countries, utilizing OSS projects. This is another important aspect to be captured in the view model of OSS. It also covers volunteer-based

humanitarian projects based on OSS.

This is the layer that deals with decision making regarding:

- operations, and
- values and goals of deliverables.

V. DISCUSSION

A. Advantages of the Three-layer View Model

The proposed three-layer view model provides a framework to decompose the complicated aspects of OSS, in order to improve the management of OSS. Also, this decomposition enables further analysis of inter-layer and inter-sub-layer relationship of each OSS project.

For example, it is increasingly common for proprietary software vendors to turn their assets into OSS. It is important to provide an assessment of OSS-readiness that covers long-term transition management. The proposed model provides a basic framework for generating a checklist for software vendors to assess OSS-readiness.

Adoption of OSS is a basic question for companies, for academics, and for the general public. In order to analyze the adoption of OSS and to design a plan for adoption, the proposed model helps facilitate adoption management by, identifying the core part of how to adopt OSS.

As the complexity of OSS projects grows, it is important to provide a methodological framework for the management of OSS. The proposed model provides a framework for systematically managing OSS projects.

In order to evaluate OSS, it is necessary to cover the multifaceted nature of OSS management. The proposed model provides a methodological framework to cover exclusive measuring of different factors in the management of OSS.

In order to assess OSS ecosystems, this model provides a framework for evaluating by making layer to layer, or sub-layer to sub-layer comparisons.

B. Implications from the Model

The proposed model facilitates increased awareness of three different management plans for each layer. It represents the different management plans as follows:

- governance and strategy management,
- community and process management, and
- software artifact management.

Not all OSS projects have all of the sub-layers. Some OSS projects have a business sub-layer, others do not. Some OSS projects have a social cause sub-layer, others do not. And, some OSS projects have a foundation sub-layer, others do not.

At the same time, some sub-layers are quite stable over time, and others are not. The license sub-layer is the most stable sub-layer in this view model. It is possible to change or update licenses in some cases, usually, an OSS project sticks to one license term for most of its life-span. Or at minimum, most projects reach a stage where the license terms are fixed. The foundation sub-layer is also stable. Usually, most OSS projects do not have foundations. The number of OSS foundations is far smaller than the number of OSS projects.

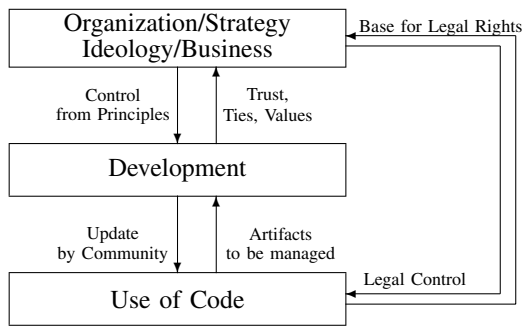


Fig. 3. Relationships among layers

On the contrary, the code sub-layer is changing. In successful projects, it continues to change day-by-day.

Some sub-layers are explicit, and others are implicit. An example of explicit sub-layer is the license sub-layer. Implicit examples include the community sub-layer. The name of the community may persist, but the internal community context often changes and evolves in many cases.

C. Inter-layer Relationships

There are several different relationships between layers. The types of relationships include:

- control,
- influence,
- constraint, and
- feedback.

The organization/strategy/ideology/business layer drives the license sub-layer's use of the code layer. Therefore, there is a strong bond between each sub-layer if it exists with the license sub-layer.

If there is a foundation sub-layer, it controls the community sub-layer in the development process layer.

The relationships among layers are depicted in Fig. 3.

From these relationships, the evolution of OSS can be measured in terms of both vertical (intra-layer) and horizontal (inter-layer) evolution.

The major relationships between the layers are shown in Table II.

D. Limitations

This is a descriptive study that explores a new framework for deconstructing complicated aspects of OSS. The quantitative analysis of each layer is beyond the scope of this paper.

There are some missing components of OSS. One example is the issue of patent protection.

There is some vague positioning also. For example, the technical infrastructure can exist in the best practices sub-layer or the foundation sub-layer.

The proposed model is a general model. The models for business-oriented, social-justice-oriented, and non-profit-oriented OSS can be considered separately. From this perspective, the current model lacks components to address each specific OSS.

TABLE II
MAJOR RELATIONSHIPS BETWEEN OSS SUB-LAYERS

type	layer-layer	description
control	foundation-community	Foundation sub-layer controls community sub-layer.
	community-code	Community sub-layer controls code sub-layer.
	community-tech info	Community sub-layer controls technical info sub-layer.
influence	technical info-community	Technical info sub-layer influences community sub-layer.
	business-foundation	Business sub-layer influences foundation sub-layer.
	social cause-foundation	Social cause sub-layer influences foundation sub-layer.
	code-techinfo	Code sub-layer and technical info sub-layer mutually influence each other.
dependency	foundation-community	Foundation sub-layer depends on trust, values, social ties, norms on community sub layer.
	foundation-code	Foundation sub-layer needs code as a source of authority for the license provider.
feedback	foundation-community	Community sub-layer provides feedback on control to foundation sub-layer.

This is an exploratory work, therefore, the in-depth analysis of the relationships among different layers and sub-layers remains for further research.

VI. CONCLUSION

OSS has become mainstream in the software industry after a more than a decade of evolution. As OSS has become widely accepted as a common software development process utilizing the advances in computer and communication technologies, the diversity and heterogeneity of OSS has increased. Its role, governance, goals, community structure and mechanisms, and business models have continued to diverge.

As the number of participants and the level of engagement of companies have increased, cultures, management, motivations and engagement have continued to diverge.

It has come to a point where there is little in common between different OSS projects in some cases. In order to maintain healthy growth of divergent OSS projects, it is necessary to provide a basic framework to describe this diversity and heterogeneity.

OSS is a complicated and dynamic phenomenon involving socio-technical interactions. The complexity, diversity, and heterogeneity of OSS projects are difficult to overestimate. In order to sort out the seven different aspects of OSS, the author proposes a three-layer view model of OSS, which leverages awareness of the complexity of OSS projects. The success of OSS projects depends on the positioning within the proposed layered view model, because a different focus requires a different strategy for fostering and maintaining OSS projects. In addition, the layered-view fosters relationships among the different layers, which is another complicating factor that impacts the management of OSS projects.

As the penetration of OSS continues, the interactions and interdependencies of OSS components are increasing. The proposed view model provides a basic framework to increase

awareness of different standpoints from different OSS projects in order to gain mutual understanding.

REFERENCES

- [1] A. Updegrove, "The future of F/OSS: Questions, challenges and opportunities," Plenary Address of OSS'10 (Notre Dame, IN, USA), June 2010.
- [2] E. S. Raymond, "The magic cauldron," <http://www.catb.org/essr/writings/cathedral-bazaar/magic-cauldron/>, August 2000.
- [3] R. T. Watson, M.-C. Boudreau, P. T. York, E. Greiner, and J. Donald Wynn, "The business of open source," *CACM*, vol. 51, no. 4, pp. 41–46, April 2008.
- [4] F. Letellier, "Open source software: the role of nonprofits in federating business and innovation ecosystems," <http://flet.netcipia.net/xwiki/bin/download/Main/publications-fr/GEM2008-FLetellier-SubmittedPaper.pdf> (a submission for AFME 2008), January 2008.
- [5] O. Alexy and J. Henkel, "Promoting the penguin: Intraorganizational implications of open innovation," Social Science Research Network, available at http://papers.ssrn.com/sol3/papers.cfm?abstract_id=988363, February 2009.
- [6] O. Alexy and G. George, "Placing a value on openness: A legitimacy perspective on open and distributed innovation," Social Science Research Network, available at http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1019527, October 2009.
- [7] LiMo Foundation, "LiMo Foundation Home page," <http://www.limofoundation.org/>, January 2007.
- [8] Symbian Foundation, "Symbian Foundation web page," available at: <http://www.symbianfoundation.org/>, 2008.
- [9] Google, "Android - an open handset alliance project," <http://code.google.com/android/>, November 2007.
- [10] S. Chopra and S. Dexter, "Free software, economic 'realities', and information justice," *SIGCAS Comput. Soc.*, vol. 39, no. 3, pp. 12–26, December 2009.