Technology Innovation Management Review

January 2014

Image licensed under CC BY Geir Tønnessen

Open Source Business

Welcome to the January 2014 issue of the *Technology Innovation Management Review*. This month's editorial theme is Open Source Business. We welcome your comments on the articles in this issue as well as suggestions for future article topics and issue themes.

Carleton UNIVERSITY

www.timreview.ca

Publisher

The *Technology Innovation Management Review* is a monthly publication of the Talent First Network.

ISSN

1927-0321

www.timreview.ca

Overview

The *Technology Innovation Management Review* (TIM Review) provides insights about the issues and emerging trends relevant to launching and growing technology businesses. The TIM Review focuses on the theories, strategies, and tools that help small and large technology companies succeed.

Our readers are looking for practical ideas they can apply within their own organizations. The TIM Review brings together diverse viewpoints – from academics, entrepreneurs, companies of all sizes, the public sector, the community sector, and others – to bridge the gap between theory and practice. In particular, we focus on the topics of technology and global entrepreneurship in small and large companies.

We welcome input from readers into upcoming themes. Please visit timreview.ca to suggest themes and nominate authors and guest editors.

Contribute

Contribute to the TIM Review in the following ways:

- Read and comment on articles.
- Review the upcoming themes and tell us what topics you would like to see covered.
- Write an article for a future issue; see the author guidelines and editorial process for details.
- Recommend colleagues as authors or guest editors.
- Give feedback on the website or any other aspect of this publication.
- Sponsor or advertise in the TIM Review.
- Tell a friend or colleague about the TIM Review.

Please contact the Editor if you have any questions or comments: timreview.ca/contact

Editorial: Open Source Business

Chris McPhee, Editor-in-Chief

Welcome to the January 2014 issue of the *Technology Innovation Management Review* in which we follow our tradition of devoting the first issue of the year to the theme of **Open Source Business**.

In the first article, **Michael "Monty" Widenius**, the founder and original developer of MySQL and MariaDB, and **Linus Nyman**, a doctoral student at the Hanken School of Economics in Helsinki, Finland, present an overview of the business side of open source software. Although much is now known about the business of open source, there are surprisingly few resources that provide a starting point for further understanding. This introductory article focuses on the motivations for businesses; common open source licenses and how they relate to community and corporate interests; issues regarding the monetization of an open source program; and open source business models. A detailed list of recommended reading is provided for those readers who wish to explore these topics in greater depth.

Next, **Juho Lindman** and **Linus Nyman** from the Hanken School of Economics in Helsinki, Finland, compare open data and open source from a business perspective. After describing the key characteristics of open data and open source, they examine the differences and similarities between the two phenomenon with regards to licensing, commercial aspects, and relevant actors. The article concludes with practical insights for managers and entrepreneurs who are evaluating business models based on open data.

In the third article, **Derek Smith**, **Asrar Alshaikh**, **Rawan Bojan**, **Anish Kak**, and **Mohammad Mehdi Gharaei Manesh** from the Technology Innovation Management program at Carleton University in Ottawa, Canada, examine the collaboration barriers faced by different groups of actors in an open source ecosystem. Based on a review of relevant literature, the authors identify the barriers that are common to all actors and the barriers that are unique to governance actors, competitors, complementors, and the core community. The article concludes with six recommendations for entrepreneurs and managers to overcome both types of barriers to collaboration in an open source ecosystem.

Finally, this issue also includes a report on a recent TIM Lecture on "The Business of Open Source", which was presented by **Michael Weiss**, Associate Professor in the Technology Innovation Management (TIM; carleton.ca/tim) program at Carleton University.

In February, we will publish articles selected from the International Seeking Solutions Summit, which was held on November 5th, 2013 in Quebec City, Canada. See our March 2013 issue (timreview.ca/issue/2013/march) for details of the Seeking Solutions approach to local open innovation.

We hope you enjoy this issue of the TIM Review and will share your comments online. Please contact us (timreview.ca/contact) with article topics and submissions, suggestions for future themes, and any other feedback.

About the Editor

Chris McPhee is Editor-in-Chief of the *Technology Innovation Management Review*. Chris holds an MASc degree in Technology Innovation Management from Carleton University in Ottawa and BScH and MSc degrees in Biology from Queen's University in Kingston. He has over 15 years of management, design, and content-development experience in Canada and Scotland, primarily in the science, health, and education sectors. As an advisor and editor, he helps entrepreneurs, executives, and researchers develop and express their ideas.

Citation: McPhee, C. 2014. Editorial: Open Source Business. *Technology Innovation Management Review.* January 2014: 3.

Keywords: open source business, open source software, licensing, business models, open data, collaboration, ecosystems, patterns

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

> *"Ideology isn't what has sold the open source model. It started gaining attention when it was obvious that open source was the best method of developing and improving the highest quality technology.*
>
> Linus Torvalds
> Software Engineer and creator of the Linux kernel

This article is meant as a primer for those interested in gaining a basic understanding of the business of open source software. Thus, we cover four main areas: i) what motivates businesses to get involved in open source; ii) common open source licenses and how they relate to community and corporate interests; iii) issues regarding the monetization of an open source program; and iv) open source business models currently employed. This article is particularly suitable for people who want a general understanding of the business of open source software; people who want to understand the significant issues regarding an open source program's potential to generate income; and entrepreneurs who want to create a company around open source code.

Introduction

In a world built on openness, in which licensing dictates that the product is not only free of charge, but can be freely copied, modified, and redistributed by enthusiasts and competitors alike, how *can* anyone possibly make money on open source? The question of how one can monetize open source software is a significant one. The quest for, and dissemination of, its answer was the spark that started what was to become the *Technology Innovation Management Review* (Lavigne, 2007: timreview.ca/article/92; McPhee, 2011: timreview.ca/article/465).

Although much has been learned during the years since the emergence of open source and the business that grew to surround it, there are still few articles that attempt to summarize its dynamics. Perhaps the most well known of those efforts is Hecker's "Setting up Shop" (1998; tinyurl.com/28n7o3), which largely focused on what strategies could be employed utilizing open source. Now that open source is a much more mature field than it was back then, we can focus on documenting what entrepreneurs have done rather than could do.

The goal of this article is to concisely explain the nuts and bolts of how the business of open source works, including sufficient detail to serve as a useful primer on the topic – a springboard for further reading. Our focus is on approaches that generate income based on open source software and its development (e.g., not hardware manufacturers with an open source involvement).

The article is structured as follows. First, we offer a brief look at some of the main corporate motivations in open source. Second, we cover the most common types of open source licenses and the main aspects and concerns for businesses and programmers regarding licensing. Third, we outline the most significant points in a piece of software's earning potential. Finally, we briefly describe the more common business models in use today, and we examine their pros and cons from the standpoints of both the developers and entrepreneurs. Included at the end of the article is a list of recommendations for further reading.

Background: Corporate Motivations

The adoption of open source code allows businesses to harness the creativity and labour of both their employees and their customers in a way that is not available to firms employing only proprietary software licenses. Indeed, where developer motivations include many social motivations, firms have tended to emphasize economic and technological reasons for entering and contribut-

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

ing to open source (Bonaccorsi and Rossi, 2003; tinyurl.com/lfx847l). In addition to the possibility of a shortened development time (e.g., Dahlander, 2007; tinyurl.com/kg8wdd6), open source projects commonly report a wider adoption of their code (e.g., West, 2003; tinyurl.com/6s68jno) and receive more high-quality feedback and bug reports than closed source projects (see Schindler [2007; tinyurl.com/mv8eea9] for a comparison). Open source licensing also enables a faster average time from discovery to solution (Schindler, 2007; tinyurl.com/mv8eea9). Indeed, open source products have been often shown to be superior to their proprietary counterparts (e.g., Wheeler, 2007; tinyurl.com/r1yk). Furthermore, companies can see development of their product in directions they did not realize was significant to their users, as well as the development of features that are too far from the firm's core business to receive in-house funding for development. As an example, only two of the more than 20 language connectors for MySQL were programmed in house; the rest were developed and submitted by the community.

By joining an open source development effort, corporations can also influence the direction of its development. Furthermore, open source has been identified as a strategy for implementing long-term sustainable software systems (e.g., Lundell and Gamalielsson, 2011; tinyurl.com/n24dw4u). Open source can also be adopted as a competitive strategy, for example through making the functionality of a competitor's product freely available (Fitzgerald, 2006; tinyurl.com/al995aj). Open source can also be of value to companies that offer products other than software, for example by promoting open source in areas that facilitate the deployment of their hardware (Fitzgerald, 2006; tinyurl.com/al995aj).

Open Source Licenses

A basic understanding of licensing is important for entrepreneurs and programmers alike. License choice decides what can be done with a program and what other programs (or, rather, licenses) it can and cannot be combined with. All open source licenses guarantee users the rights to use the program, access the source code, modify the source code, and redistribute the program in its original or modified form. However, beyond these basic rights, licenses differ in significant ways. Based on these differences, open source licenses are commonly divided into three main categories: i) permissive licenses, ii) weak copyleft licenses, and iii) strong copyleft licenses. The licensing requirements of copyleft licenses are only triggered upon distribution. This means that, for personal use, one can do largely whatever one wants with open source code, but if and when one distributes a program the stipulations of the license are triggered and must then be complied with. Note, however, that the AGPL license has some minor restrictions, which will be discussed later.

One of the most important elements of, and differences between, open source license types relates to a concept called *license compatibility*. License compatibility is a term used to describe the issue of which licenses can be combined. Particularly, from a business perspective, license compatibility considers which licenses can be combined with proprietary software. A further issue, though one of lesser interest, is that of the right to change the license, in particular whether one is allowed to change an open source license to a proprietary one. For businesses, this may be of interest as a source of free code. The issue of changing to a proprietary license splits the developer community into two camps. Those who are for it generally want to ensure (or at the very least do not mind) that their code is as valuable to corporate interests as possible. Those who are against it generally want to ensure that the open source project remains a freely available community good in perpetuity. The issue of license combining (including embedding) and license change is summarized in Table 1.

Permissive licenses

Permissive licenses allow a high degree of freedom to use and reuse (or fork) the code. It is not an extreme oversimplification to distil the permissive licenses down to the message: "here's the code, do whatever you want with it". (Commonly, one needs to distribute a copy of the copyright with the code, but in practice,

Table 1. Post-distribution rights of open source license types

Rights	Permissive	LGPL	GPL
Can it be combined with proprietary programs?	Yes	Yes	No
Does it allow the license to be changed to a proprietary license?	Yes	No	No
Does it guarantee access to source code?	No	Yes	Yes

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

this need not be more complicated than including a readme file.) In other words, it is possible to fork a permissively licensed program and make it closed source. (As an example, both Apple's OS X and iOS operating systems contain code that was copied from permissively licensed open source projects, most notably BSD: tinyurl.com/kffrf.) An issue which sets the permissive licenses apart from the copyleft licenses is that, once the source code is compiled, one does not need to distribute the original source code with the compiled (i.e., binary) version of the program. Among the more common permissive licenses are the Apache (tinyurl.com/kmenxch), MIT (tinyurl.com/3vfsyal), and BSD (tinyurl.com/lejoxn7) licenses.

Weak copyleft licenses (LGPL)
Weak copyleft licenses, such as the GNU Lesser General Public License (LGPL; tinyurl.com/mp4w4lw), can be combined with proprietary code, but cannot be relicensed under a proprietary license. So, although a firm's proprietary program can remain proprietary, even when combined with the LGPL, the LGPL-licensed program cannot be made proprietary. Furthermore, any modifications to an LGPL program must also be licensed under the LGPL. The Mozilla Public License (MPL; mozilla.org/MPL/) is also a weak copyleft license.

Strong copyleft licenses (GPL)
Much like the LGPL is synonymous with *weak* copyleft, the GNU General Public License (GPL; tinyurl.com/2459b5) is synonymous with *strong* copyleft. Hence, we will focus our discussion of strong copyleft licenses on the GPL. Although use of the GPL is in decline (Aslett, 2011; tinyurl.com/7ujq7sj), as of the writing of this article, it is still the most common open source license overall (Black Duck Knowledgebase; tinyurl.com/nl4z94t). The GPL requires any modifications to the code to also be licensed under the GPL. From a business perspective, the key issue to be aware of is that combining or embedding a program with the GPL necessitates the (re)licensing of all connected software so that it is also under the GPL. In practice, this means open sourcing any proprietary programs connected to a GPL-licensed program, and is therefore something many firms seek to avoid. Importantly, programs licensed under a GPL license cannot be re-licensed under a more permissive license (i.e., neither as LGPL or permissive).

A general comment regarding license change is that one can commonly change a license to a more restrictive license type but not to a more permissive one. Furthermore, only the permissive licenses can be changed to proprietary.

With the rise of cloud computing, a variation of the GPL license worth special mention is the Affero General Public License (AGPL; tinyurl.com/lzmmq8n). The AGPL differs from the GPL in that online use of a program is considered distribution, thus triggering the requirement for license compliance (i.e., source code access is required) even though a physical copy of the program has not been distributed. In other words, using an AGPL-licensed program in the cloud necessitates distribution of source code.

Choosing a license
Open source licensing is a more complex topic than can be covered in detail here. Furthermore, because legal precedent is rather limited, there are issues regarding licensing that are still subject to interpretation and that are coloured, among other things, by pragmatic versus ideological concerns. Thus, what may and may not be done under certain conditions is to some extent a matter of opinion. We recommend a close study of licensing before any final licensing decisions are made. For further reading, please refer to the links at the end of this article.

On the Business of Open Source

Establishing a sufficient, steady income is a significant challenge in creating a company around open source software. Thus, although open source is a superior development model, there is no guarantee that one's program will make enough money to fund its continued development. Of particular significance to the business of open source are the questions of program ownership and location in the software stack, because these factors affect what business models one can choose from. In particular, the answers to these questions help decide whether one can employ what is arguably the most lucrative open source business model: dual licensing.

Ownership of code
A company or person that owns the rights to the code they develop can sell closed source copies of the code, which is a standard practice with proprietary programs. The dual licensing, business source, and (to a lesser extent) open core business models, which will be described in further detail later, require ownership of the code.

Location in the software stack (and "embedded" programs)
Most software relies on other software to run. This concept of software codependence is most apparent in the so-called software stack. On the top of the stack is the application: a word processing program, a photo ed-

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

itor, a game, etc. Digging deeper, one can find elements such as databases, middleware, and an operating system. It is not important for the purposes of this article to understand the layers or functions of a software stack; it is merely enough to know that such layers exist and that a program's location in the stack is significant to its overall importance to the stack. Programs higher up in the stack rely on programs lower down to function, but not the other way around. Whereas a word processor needs an operating system to be able to run, an operating system does not need a word processor for it to function. One way for an open source program to gain potential value is having other programs rely on it: by being embedded in the software stack and by being a required component for applications and other programs to function properly – or even run at all.

Business Models

Although a business model can usefully be seen as something much more complex than merely a revenue source (e.g., West, 2007: tinyurl.com/dxsemd; Bailetti, 2009: timreview.ca/article/226), at its essence is the question of how the firm can create value for the customer while simultaneously extracting some of that value for itself (West, 2005; tinyurl.com/ov69jb8). For the purposes of this article, we make use of very broad brush strokes in our interpretation, using the term "business model" to indicate the way in which a company delivers value to a set of customers at a profit (e.g., Johnson, 2010; tinyurl.com/m9uf6xe). Recommended reading for more in-depth analyses of questions related to business models are offered at the end of the article.

The business models of open source can be divided in two main categories: those that require complete (or at least partial) ownership of the code and those that do not. Table 2 outlines the criteria for selecting an open source business model; however, it should be noted that these business models need not be mutually exclusive.

Support contracts and services

Support and services are closely related approaches; in fact, companies that provide one commonly also provide the other. Thus, although they could be separated, we have chosen to group them under one heading. The services business model is one in which income is generated by offering services in the form of, for example, training, consulting, or extensions development around an open source product. Companies that offer services will commonly also offer long-term support contracts, thereby achieving a more stable income than by merely focusing on one-off services. Two of the main challenges with the support and services approach are the lack of scalability and that the typical profit margin of 20–30% is not enough to pay for full-time developers for the project.

The availability of support and services is an important factor for customers (e.g., Shanker, 2012; timreview.ca/article/635) and can be considered a necessary element for software to become truly successful. Bear in mind that, although support should be offered, it need not be provided by the same company that develops the software. Examples of a support and services providers are Red Hat (redhat.com) and SkySQL (skysql.com). For more information on Red Hat's approach, see Suehle (2012; timreview.ca/article/635).

Open core or commercial extensions

Open core is a business model in which the core of a program is open source, with additional closed source features provided for a fee. Open core has gained much momentum over the past few years. However, it is an approach primarily focused on appealing to the venture capitalist rather than the end user (Prentice, 2010; tinyurl.com/pqpmptk). The economic rationale is clear-cut, but the reaction of the community and customers may not be as easy to estimate. Although pragmatic firm motivations are accepted by the community provided they comply with the rules of the community (Bonaccorsi and Rossi, 2003; tinyurl.com/lfx847l), some developers see

Table 2. Criteria for business model selection

Criteria	Support and Services	Open Core	Business Source	Dual Licensing
Can I choose this business model even if I do **not** own the code?	Yes	No*	No	No
Can I choose this business model even if my program is **not** embedded?	Yes	Yes	Yes	No

*Ownership is required for the closed source extensions

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

the open core approach a breach of those rules. The proponents of free software criticize it on ideological grounds and proponents of open source software criticize it on technical grounds, due to the restrictions to the development model caused by limited access to the code. From the perspective of the end user, open core forces vendor lock-in and is furthermore faulted with not delivering and sustaining the cost savings and flexibility of open source software (e.g., Phipps, 2012; tinyurl.com/9tjv8c9). Potential outcomes of adopting this model may include problems in attracting and maintaining developers (see Dahlander and Magnusson, 2005: tinyurl.com/88djuec; 2008: tinyurl.com/6w6k95q), or even the emergence of a competing fork (Nyman, 2013; tinyurl.com/mahze3o).

However, it should be noted that there are successful open core projects, which show that the approach can work. If considering an open core approach, it is worth bearing in mind that the more useful the core product is, the greater the potential community interest will be. Thus, making non-critical parts of the program closed will lessen the potential negative effect on developer interest in the project. A time-limited hybrid licensing (Sprewell, 2010; tinyurl.com/n8zeoqr), in which the closed source components of open core become open source after a 1–5 year delay, has been proposed to help meet the demands of both users and developers. However, we posit that the business source approach explained below may be a more mutually beneficial means to the same end. Examples of open core are not as easy to come by as the frequent discussion of the topic over the past few years would imply. Perhaps the best-known example is MySQL (mysql.com), which offered dual licensing of an identical product (a closed source and a GPL version) under its previous owners, but has changed to an open core approach for its free version after it was purchased by Oracle (Young, 2011; tinyurl.com/3hyxttc).

Business source

Business source is a business model that employs two different licenses with a time delay. In this business model, the source code is openly distributed and freely editable. However, for a set amount of time, a predefined segment of users (0.1–1% is suggested) have to pay to be allowed to use it. After this initial time period (3 years is suggested), the license automatically changes to an open source license. Business source is a new entrant in the field of open source licensing, which we first detailed in the June 2013 issue of the *Technology Innovation Management Review* (Widenius and Nyman, 2013; timreview.ca/article/691). It was created to help simultaneously meet the needs of both the open source community and the open source entrepreneur; being too restrictive in one's licensing can harm community growth, whereas being too permissive can harm business growth. Though a newly introduced concept, there are already reports of companies switching to business source, with both developers and owners pleased with the results (Widenius, 2013; tinyurl.com/mkurs58). For a more in-depth presentation of the business source approach, with a sample license, see Widenius and Nyman (2013; timreview.ca/article/691).

Dual licensing

Dual licensing is a business model in which a program is offered under two separate licenses, commonly one version under a copyleft, GPL-style license and another under a commercial, closed source license allowing for proprietary use (and combining with other proprietary software). Traditionally, the source for both versions is identical, except for changes in the copyright. Dual licensing is the best option for programs that are embedded, and for which one owns the code. The primary customers are companies who want to include software in their own packages, but who do not want to release their code under open source, as is required by the GPL. Its excellent scalability makes dual licensing the most potentially lucrative of the business models presented herein. The first ever program to adopt a dual licensing approach was Ghostscript (tinyurl.com/2p6zmt); MySQL (msql.com) – (before and during its ownership by Sun – was the second program to utilize this approach, and the first to use GPL as the open source license.

Software as a service

Software as a service (SaaS) is a fairly new business model in which connectors and application programming interfaces are open source but the server code they connect to is not accessible to the end user. For instance, one may use an application that can access certain data on a server, but not be able to access the actual source code (of, for example, the database management system) on the server one accesses. Although SaaS is not directly related to open source, it is included here because it can incorporate open source components. Examples of SaaS businesses are Salesforce (sales force.com) and Web of Trust (mywot.com); in building their service, they may use open source software on their servers, but this software is not distributed to their users.

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

Managerial Implications

When deciding whether or not to start an open source project, the following managerial implications should be considered:

1. Before starting a new open source project, check if a similar project already exists. Participating in an active program is preferable to starting a new fork. If there are similar programs that have been abandoned, do some research to find out why they were abandoned. Repositories such as GitHub (github.com) and SourceForge (sourceforge.net) have a myriad of abandoned programs.

2. Find a company or a group of users that want to work with you to define the scope of the project. From the start, you will want to have users using the product while it is still in development.

3. Two of the most important decisions will be business model and license. If you are planning on relying on community participation, be mindful of their reactions to both business model and license choices. See the end of this article for further reading on community.

4. In choosing a business model, consider these questions: Do you want to concentrate on services or development? Do you plan to have a big community or work with a few big companies? Do you plan to take in investors? And, if so, what is your exit plan?

5. In choosing a license, consider these questions: What will your business model be? How much control do you want to have over the use (and potential forks) of your code? What kind of community do you want to attract around the product?

6. If you plan to rely on community participation, remember to use community-creating tools to reach and communicate with them: web pages, a forum or knowledge-base, email lists, bug system, build systems, source code repository, etc. You can start by hosting your project on GitHub, SourceForge, or another repository, but you will eventually want to host it yourself.

7. Significant enabling factors for creating a successful business around open source are ownership of code and embeddedness (a program's location in the software stack). These same factors also largely determine what business models one can choose from. Figure 1 provides a flowchart to help choose a business model based on ownership, embeddedness, and intentions for further development. If the flowchart recommends against starting a business, consider either partnering, or releasing the code (e.g., under an Apache or BSD license) for someone else to continue developing the software.

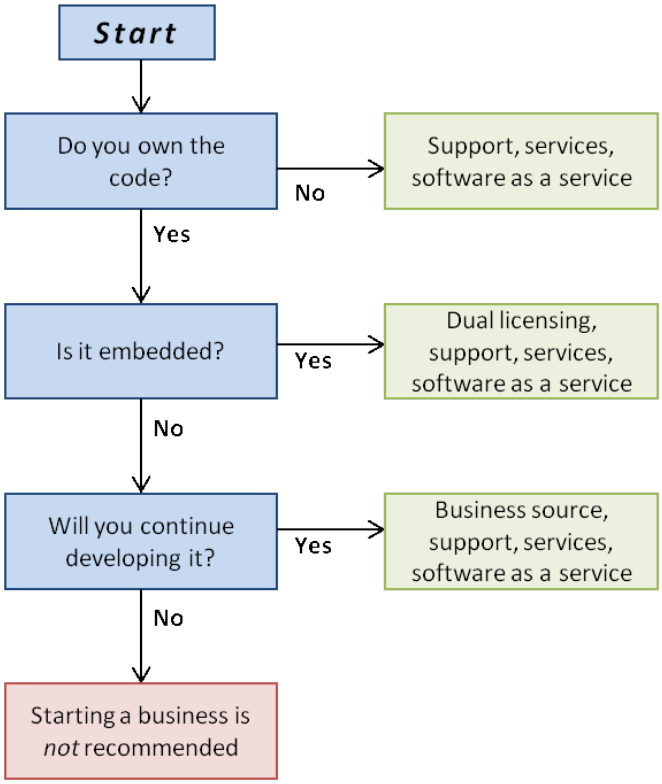

Figure 1. Flowchart for choosing an open source business model

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

Conclusion

Through this primer, we have given a brief answer to the question: "How can one make money on open source?" To the uninitiated, financing a business based solely around the development of open source code may perhaps seem somewhat enigmatic. Although challenging, it is nonetheless possible. Our goal in this article was to clarify this enigma by explaining some of its most significant parts.

The possibilities for monetization of a program are dependent on many factors, and key among them are ownership of code, choice of license (including the issue of license compatibility), and location in the software stack. These factors in turn affect the choice of business model.

As a primer, this article will hopefully provide a useful introduction to the business of open source. It is not intended to cover every aspect of open source businesses in full detail, nor can it provide conclusive recommendations that will apply in every case. However, in Table 3, we have included a list of recommended reading for those that want to dive deeper into the topic.

About the Authors

Michael "Monty" Widenius is the founder and original developer of MySQL and MariaDB. He has been an entrepreneur since 1979 and is the founder of MySQL Ab, Monty Program Ab, SkySQL, the MariaDB Foundation, and Open Ocean capital.

Linus Nyman is a doctoral researcher at the Hanken School of Economics in Helsinki, Finland, where he is researching code forking in open source software. A further research interest of his is free-to-play gaming. He also lectures on corporate strategy, open source software, and the new business models of the Internet age. Linus has a Master's degree in economics from the Hanken School of Economics.

Citation: Widenius, M. and L. Nyman. 2014. The Business of Open Source Software: A Primer. *Technology Innovation Management Review*. January 2014: 4–11.

Keywords: open, open source business models, open source software development, open source licenses, dual licensing, business source, open core, entrepreneurship

The Business of Open Source Software: A Primer

Michael "Monty" Widenius and Linus Nyman

Table 3. Recommended reading

Topic	References
Open source licensing	• Välimäki (2005; tinyurl.com/ahljzwu) • International Free and Open Source Software Law Review (ifosslr.org)
Open source license selection in relation to business models	• Daffara (2011; timreview.ca/article/416)
License compatibility, compliance, and legality concerns	• Wheeler (2007; tinyurl.com/cbc579) • Hammouda et al. (2010; tinyurl.com/bfp82mw) • Lokhman et al. (2013; tinyurl.com/n64q3wm)
Popularity of various licenses	• Black Duck Knowledgebase (tinyurl.com/kp25s8s)
More on specific licenses	• Open Source Initiative (opensource.org/licenses) • Free Software Foundation (tinyurl.com/4e7wm)
Open source and business models	• West (2007; tinyurl.com/dxsemd) • Bailetti (2009; timreview.ca/article/226) • Hecker (1999; tinyurl.com/28n7o3) • Prowse (2010; timreview.ca/article/366) • Suehle (2012; timreview.ca/article/513)
Matching licenses with business models	• Lindman et al. (2011; tinyurl.com/na3e6fd)
Business source	• Widenius and Nyman (2013; timreview.ca/article/691)
Partnership strategies	• Riekki-Odle (2010; timreview.ca/article/364)
Business models for companies partnering with an open source vendor	• Groganz (2011; timreview.ca/article/463)
Collaboration models between open source projects and their communities	• Noori and Weiss (2013; timreview.ca/article/647) • Weiss (2011; timreview.ca/article/436)
Customer value propositions for corporate open source software	• Shanker (2012; timreview.ca/article/635)
Open source support and its requirements	• Peters (2007; timreview.ca/article/54)
Establishing a community	• Byron (2009; timreview.ca/article/258)
Participation architecture in corporate open source	• West and O'Mahoney (2008; tinyurl.com/66fly95)

The Businesses of Open Data and Open Source: Some Key Similarities and Differences

Juho Lindman and Linus Nyman

> *"It's difficult to imagine the power that you're going to have when so many different sorts of data are available."*
>
> Tim Berners-Lee
> Inventor of the World Wide Web

Open data and open source are phenomena that are often automatically grouped together, perhaps because they share the word "open". A careful analysis of what open means in each of these cases is a stepping stone towards building viable businesses around both open source applications and on open data. Although there are, indeed, elements they share through their openness, the ways in which they differ are significant. In this conceptual paper, we aim to outline the differences and similarities of the two phenomena from a commercial perspective.

Introduction

Open source and open data both have a focus on "openness", and most developers and researchers could easily identify similarities between the two phenomena. For example, both open source and open data are enabled – or at the very least greatly helped – by the Internet, which provides a backbone for collaborative development efforts, communication infrastructure, as well as a means to support the sharing of both application and data. However, open source and open data are distinct phenomena with significant differences, and these differences clearly impact how commercial success can be achieved in each domain.

Given that TIM Review readers tend to be more familiar with open source than open data, our goal is to explore the concept of open data through a comparison with open source and with an emphasis on the similarities and differences that are relevant to technology businesses. We focus on three key questions:

1. Are the phenomena similar?

2. Are the licenses of software and data similar?

3. Are the businesses and revenue models similar?

Understanding these two phenomena is useful to managers and entrepreneurs interested in the business potential of the released data sets. This understanding is also useful for open data proponents who are interested in the business aspects of open source and the lessons that the business of open source offers. Designers of related services may be interested what potential open data and open source have to offer in terms of novel and better service opportunities.

The structure of the article is as follows. We first describe key characteristics of open source and open data. We then compare these two phenomena from three business-oriented perspectives: licensing, commercial aspects, and relevant actors. Finally, we provide some takeaways for managers and entrepreneurs.

Comparing Open Source and Open Data

In computer science, in theory as well as in practice, the distinction between data and application is critical. Therefore, the most obvious and fundamental difference between open data and open source is that the former focuses on the data and the latter focuses on applications.

Data has multiple meanings, including any end-product of measurement, but in this investigation, we use a slightly more technical definition of data: data refers to stored symbols. Data is considered a resource – raw material for the application. Open data means data that is technically and legally made available for re-

The Businesses of Open Data and Open Source: Some Key Similarities and Differences

Juho Lindman and Linus Nyman

use and republication. The underlying idea is that the increased transparency will help to create trust in users and developers, as well as offer a way to create new services based on the collected data. In many cases, the data is collected by government entities for various purposes and thus additional economic value would be created when the published data is put to use. However, open data includes open government-collected data as well as data released by private actors.

An application, on the other hand, is compiled source code that operates on data. Open source refers to a legal and technical arrangement related to software production that results in open source code that is accepted under an license that complies with the Open Source Definition (opensource.org/definition). These licenses are based on the copyright protection of the code; thus, the "open" of open source refers to the source code.

To summarize: the first significant difference between open data and open source is that of data versus application. Data can be numbers, locations, names, etc. In and of itself, data does nothing. Source code, or rather an application, is something that uses or produces data. These two aspects, although they rely on one another for their significance, are different in both essence and purpose. Indeed, it is some of these differences that this article seeks to point out and clarify.

Comparing Licensing Aspects

The key similarity between open data and open source lies in the prerequisite of openness. But what, exactly, is it that is open, and are there degrees, or types, of openness? For open source software, the openness primarily means a guaranteed access to the application's source code as well as an arrangement that makes sure that the code can be forked, modified, and redistributed. (For more on the significance of the right to fork, see Nyman and Lindman [2013; timreview.ca/article/644].)

For open data, a similar "access principle" provides access to the data (and metadata) and it provides the opportunity to reuse it in applications. Data also needs to be maintained and updated. The actor that collects or mashes up the data from different sources usually has the option to stop providing access or maintenance to the data (i.e., to "close" the data).

Source code can be copyrighted (or copylefted: tinyurl.com/qygb2) but, in some cases, data falls outside copyright protection. Whether particular data can be copyrighted is subject to national legislation. However, copyright is not the only law that applies to data; depending on its content, other laws may also regulate its use. Laws may govern the collection, storage, maintenance, access, use, and representation of data. For example, laws relating to sensitivity, privacy concerns, or national security may apply to different datasets.

Table 1 compares different licensing aspects of open source versus open data. The legal arrangements (i.e., copyright, licenses, original publisher, and role of contracts) for the two phenomena are different. For a more thorough discussion on data licensing, see the "Guide to Open Data Licensing" (tinyurl.com/lkhg6df), which is published by the Open Definition project.

Table 1. Licensing of open data versus open source

	Open Source (Application)	**Open Data (Data)**
Copyright	Applies to all source code	May apply
Licensing	Licenses must comply with the Open Source Definition (opensource.org/osd-annotated)	Relevant if protected by copyright. Possible licensing options include the Wikimedia commons (commons.wikimedia.org) and the Creative Commons (creativecommons.org)
Original publisher	Several versions (distributions) of application and forking are possible	Data often collected, maintained, and controlled by data publisher
Contracts	Normally not required; the license agreement defines the rights of the developers and users	Data publisher may have an incentive to monitor data use or to create feedback loops to reusers of their data

The Businesses of Open Data and Open Source: Some Key Similarities and Differences

Juho Lindman and Linus Nyman

Comparing Commercial Aspects

The business of open source is in itself a diverse field, with companies generating income from various sources related to open source products. The two main categories of income are those from support and services related to a program (e.g., training, consulting, feature development) and from selling closed source versions of an open source program, a practice called "dual licensing". For a primer on the business of open source, see Widenius and Nyman (2014) in this issue of the TIM Review.

In contrast, the business of open data is a young field, but it holds promise for service innovation. Discussion on the viable revenue sources is still ongoing. The data publisher or the user of the application pays the costs related to the collection, maintenance, and enrichment of data, but customers normally do not pay subscription fees for accessing data.

In the following subsections, we compare the expense-related and income-related considerations of businesses that rely on open data or open source .

Who pays the bill, and why?
Open source can save firms money if they are able to attract free community participation. However, companies may also be willing to support open source development, for example by paying a developing group or foundation, or by assigning its own developers to an open source project. Even though anyone, even the firm's competitors, can then benefit from any improvements to the code, this approach is common in open source development. Typically, firms use this strategy to develop aspects of their product offering that would be considered "table stakes" (tinyurl.com/5u4aut). By collaborating – even with competitors in the same or similar markets – to develop non-unique foundational aspects of an application, companies save time and development effort, which can then be redirected into developing the aspects of their offering that will differentiate them from their competitors.

In addition to the costs of collecting data, open data providers must often spend money and effort both to clean up the data for publishing as well as for keeping it open. With such tasks, providers may benefit from community participation, just as in the case of open source software. Issues related to "community management" are therefore similar in both cases.

Who makes money, and how?
Openness usually means that an application or dataset can be acquired free of charge. In the case of open data, the publishers are normally considered to have given permission for others to build services on top of their released dataset. The services provided by these other parties may add value above and beyond just the provision of data, and the costs of designing the applications, collecting the data, and maintaining the services are covered by various different arrangements depending on the motivations of the other parties, their possible business models, and the nature of the value created.

Value can stand for both economic value (i.e., money) or a wider benefit. The openness of both open source and open data can potentially offer either one of these two types of value. However, it is notable that the dynamics that produce value are different. In Table 2, we list some of the benefits perceived by the key actors in each case. The table is not exhaustive list, but it provides an illustration of topical issues.

One of the main differences in the business aspects of open data and open source is that, at least currently, it is rare for the data provider to make money on open

Table 2. Examples of key value sources in open source versus open data

	Actor	**Economic value**	**Other value**
Open Source	Companies	Dual licensing, support and services	Product innovation, platform innovation
	Customers	Cost savings	Evade vendor lock-in
	Actor	**Economic value**	**Other value**
Open Data	Data owner	Sales of premium access	Public service, receive additional developmental resources
	3rd party	Sell applications	Increased transparency, novel services

The Businesses of Open Data and Open Source: Some Key Similarities and Differences

Juho Lindman and Linus Nyman

data. The main perceived sources of benefits are related to public service or to situations where the data needs to be collected and maintained anyway. Releasing the datasets would then enable others to benefit from them and may result in new and useful services. Typically, the funding for the collection and maintenance of the data in such situations also comes from public sources. Normally, a company waives the possibility of data sales when it decides to release a dataset. However, we speculate that open data could also be accompanied by a "premium access" option, meaning access to more real-time data, faster access, or access to datasets that contain both open and closed data.

For open source, the commercial actors have established business models that rely on, for example, dual-licensing. Open source has tried and true ways to cut costs and evade lock-in situations. However, many open source contributions are driven not solely by commercial interests, but by the desire for useful software that addresses specific needs, among other motives.

Open data can offer opportunities for downstream service provision. In such situations, some actors that provide open data might be keen to share their costs with the downstream actors that make a profit. It is possible to sell downstream applications, such as closed source software subscriptions. Developers might also have other motivations in writing software that uses open data, such as increased transparency, new visualizations, service provision, etc.

Comparing Elements and Actors

In comparing the elements and actors involved in open data versus open source, we limit our investigation to the four questions illustrated in Figure 1:

1. Who are the main actors?
2. How is the output developed?
3. What is the output of the development?
4. Who is interested in the output?

In an open source project, both a corporate community and an open source community can participate in the software development. When data is being developed for release, data consultants as well as those who clean data participate to the process. Output of the processes are released as open source programs and as published open data.

As shown in Figure 1, the main difference concerning the processes is that the open source process is more open than the open data process. The developers are able to participate in open source software development with varying motivations. For community driven projects in particular, motivations commonly are not financial. In open data, the data publisher is usually expected to carry costs related to releasing the data, such as the costs of collection, aggregation, and anonymization. If these services are outsourced, there is new business opportunity for companies that provide them.

The output is also different: in open data, the data ultimately remains the same through the process, whereas the open source development process aims to change the software. The software is an end in itself, whereas the released open dataset is just the first step in providing the service to the customer.

Conclusions

Although open source licensing has established its value for developers in enabling a viable development model, the business of open data needs further study. Nonetheless, both open source and open data hold potential for business. Their main difference as phenomena is that of data versus application.

Proponents of both phenomena promote the openness of the output, which offers transparency but also changes the competition dynamic. The open source alternative hampers traditional software subscription sales. Open datasets can be easily copied, but the original data collector still has a prominent role in the maintenance of the said dataset: a copied dataset, if not maintained properly, may soon become obsolete.

The commercial potential of open source has been tested and proven over the years, and several business models have emerged. However, the business of open data is still in a pioneering phase. The publisher's role is critical in any open data business, but the publisher might not be the actor that benefits most from data publication; the greater opportunity for entrepreneurship may lie downstream, where value may be captured from services built upon the open data.

Despite their differences, both open source and open data aim to attract development efforts far beyond the originator of the project. Contributors may be driven by a variety of motivations, not excluding economic gain. Economic gains are feasible and attainable, but capturing them requires entrepreneurs and managers to understand the differences in the development process and economic value capture.

The Businesses of Open Data and Open Source: Some Key Similarities and Differences

Juho Lindman and Linus Nyman

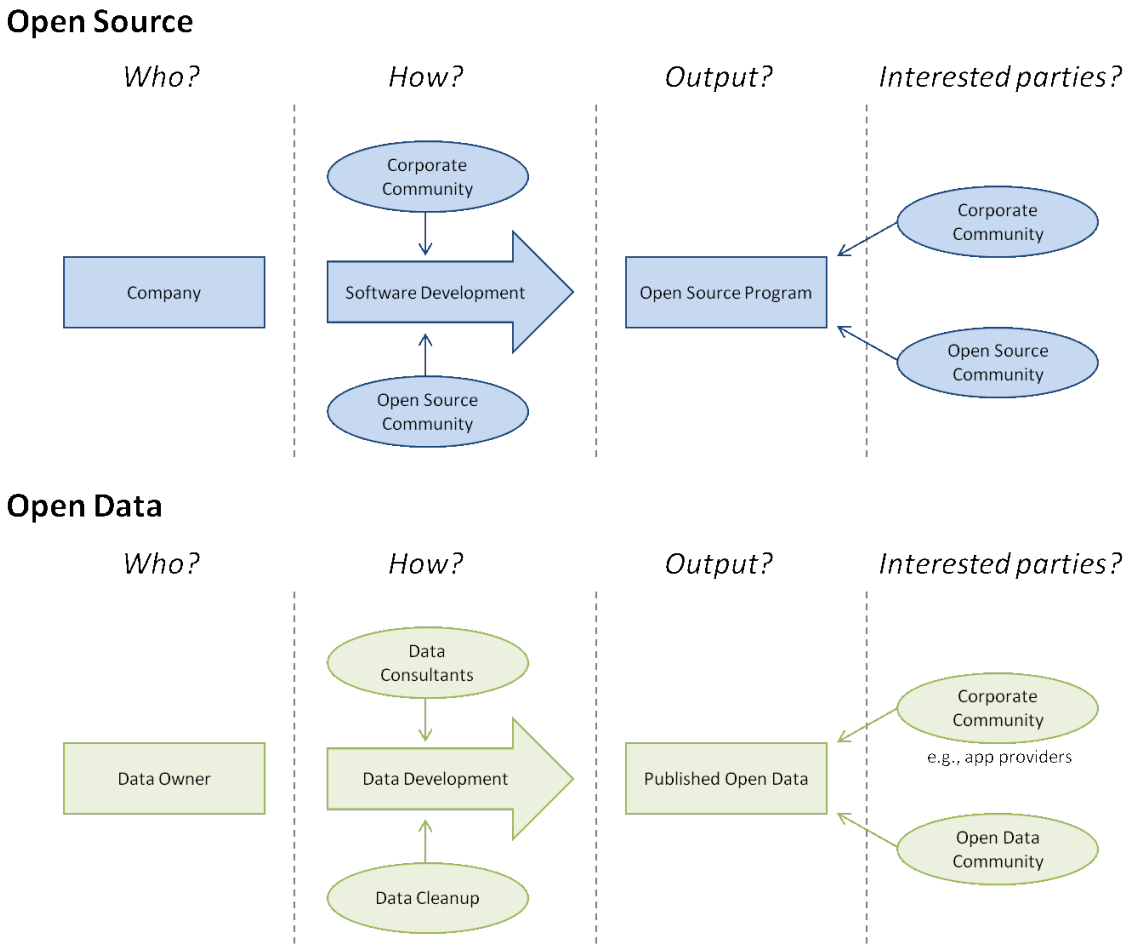

Figure 1. Comparing actors and activities of an open data versus open source project

Insights for managers and entrepreneurs

When evaluating business models based on open data, managers and entrepreneurs should consider the following key questions:

1. Do you have sufficient familiarity about the dynamics of open data ecosystems and the required technical capabilities? Open data is a significantly different field from open source; in-depth knowledge of one does not automatically guarantee sufficient knowledge of the other, although open source experts will easily find similarities.

2. If the data is not yours, how certain are you that it will continue to be provided openly in the future? How can you safeguard the relationship between the data collector/maintainer?

3. What is the legal status of the data? Does it allow fees, and what would happen if fees were introduced?

4. What is the license of the software application? Is it possible to gain software subscription revenue from a proprietary application built on the open data stack?

5. What national and international legislation poses obstacles to the service? Whereas software is relatively free of these concerns, many datasets may contain sensitive information. Open datasets can also be combined with other (also private) datasets, and this combination may raise new legal issues.

6. How will you attract developers? What are the dynamics of the development community? How will you support a relationship that takes into account different motivations to participate and benefits the different actors?

In conclusion, a final similarity we can note regarding open source and open data is that open data is now in a

The Businesses of Open Data and Open Source: Some Key Similarities and Differences

Juho Lindman and Linus Nyman

position not dissimilar to that of open source some two decades ago: a new, interesting phenomenon with promise, but also skepticism and challenges regarding an entrepreneurial potential for revenue creation and value capture.

The business models and strategies surrounding open source did not evolve overnight. Open data is an emerging field that may have many opportunities yet to be discovered. We believe that the world of open data holds great untapped potential for knowledgeable entrepreneurs that can identify opportunities for its use.

About the Authors

Juho Lindman is an Assistant Professor of Information Systems Science at the Hanken School of Economics in Helsinki, Finland. Juho's doctoral dissertation from the Aalto University School of Economics in Helsinki focused on open source software development organization In the field of information systems, his current research is focused in the areas of open source software development, open data, and organizational change.

Linus Nyman is a doctoral researcher at the Hanken School of Economics in Helsinki, Finland, where he is researching code forking in open source software. A further research interest of his is free-to-play gaming. He also lectures on corporate strategy, open source software, and the new business models of the Internet age. Linus has a Master's degree in economics from the Hanken School of Economics.

Citation: Lindman, J. and L. Nyman. 2014. The Businesses of Open Data and Open Source: Some Key Similarities and Differences. *Technology Innovation Management Review*. January 2014: 12–17.

Keywords: open source, open data, business models, licensing, entrepreneurship

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

> *"Yes, we are all different. Different customs, different foods, different mannerisms, different languages, but not so different that we cannot get along with one another. If we will disagree without being disagreeable."*
>
> J. Martin Kohe
> Author and Psychologist

Leveraging open source practices provides value to businesses when entrepreneurs and managers understand how to collaborate effectively in an open source ecosystem. However, the complex mix of different actors and varying barriers to effective collaboration in the ecosystem pose a substantial challenge. How can a business create and capture value if it depends on effective collaboration among these different groups? In this article, we review the published research on open source collaboration and reveal insights that will be beneficial to entrepreneurs and managers. We organize the published research into four streams based upon the following actor groups: i) governance actors, ii) competitors, iii) complementors, and iv) the core community. Then, through induction and synthesis, we identify barriers to collaboration, first by ecosystem and then by actor group. Finally, we offer six recommendations for identifying and overcoming barriers to collaboration in an open source ecosystem.

Introduction

Collaboration is the act of working with another individual or grouping to create something. It involves working jointly and openly with others. When collaborators bring their own diverse skills and experience – and new perspectives – the potential for innovation is great. However, collaboration is far from easy, and the diversity that brings benefits to the experience can also present barriers to collaboration. In an open source ecosystem, where collaboration is essential and the diversity of contributors is often high, these barriers can be substantial.

Open source collaboration is the act of working with different group of actors on a project to produce and create open source software (Nan and Kumar, 2013; tinyurl.com/k5a8yt3). For companies that wish to leverage open source software as part of their business models, effective collaboration is essential. However, the open environment introduces business- and people-related issues that can restrict or prevent open collaboration among the different groups of actors. Entrepreneurs and managers are faced with questions such as: How open should an actor be with sensitive or confidential business information, and with whom can it safely be shared? How can we collaborate openly with a competitor? How do we collaborate with actors from around the world, where cultural differences may affect our interactions? These business- and people- related issues inherent in an open source ecosystem create barriers to effective collaboration if entrepreneurs and managers fail to understand and overcome the barriers.

In this article, we examine the barriers to effective collaboration in an open source ecosystem, and we ask whether the different groups of actors in such an ecosystem face the same or different barriers. To success-

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

fully collaborate in an open source ecosystem, entrepreneurs and managers must understand: i) the degree of similarity or dissimilarity between barriers to collaboration in such an ecosystem and ii) how the barriers relate to the different groups of actors.

In the existing literature, previous research has focused on specific actors in open source ecosystems, such as: foundations, communities, governments, complementors, competitors, leaders, developers, adapters, users, and expert users. However, there is a lack of research from the broader perspective of the collaboration barriers that may arise within and between these various groups of actors. To achieve business success through leveraging an open source ecosystem, entrepreneurs and managers need to:

1. Identify and understand barriers to collaboration common to all groups of actors in an open source ecosystem

2. Identify and understand barriers to collaboration unique to specific groups of actors in an open source ecosystem

3. Overcome all these barriers for overall effective collaboration in an open source ecosystem

In this article, we make five contributions. First, we identify barriers to collaboration common to all groups of actors in an open source ecosystem. Second, we categorize actors in an open source ecosystem into four different groups. Third, we identify open source collaboration barriers unique to the different groups. Fourth, we assemble the research articles relevant to the topic of collaboration in open source ecosystems into four streams based upon our four different groups of actors. Finally, we provide recommendations to entrepreneurs and managers to identify and overcome barriers to collaboration from a different group of actor perspective.

The article is organized into four sections. The first section summarizes the results of our literature review concerning open source collaboration. The second section provides a definition of open source collaboration as it relates to open source business and an open source ecosystem. This section also identifies open source collaboration barriers common in the ecosystem and unique to the four different groups of actors. The third section provides recommendations for entrepreneurs and managers to overcome collaboration barriers in an open source ecosystem. A final section concludes the article.

Literature Review

Table 1 summarizes the 15 papers we reviewed following a search of literature relating to open source collaboration. Based on patterns we observed, we organized the literature into four streams based on the relevance to four groups of actors in an open source ecosystem: i) governance actors, ii) competitors, iii) complementors, and iv) the core community. Organizing the literature in this way revealed insights into the common barriers in an open source ecosystem and the barriers that are unique to each group of actor.

1. Governance actors

Five of the articles we reviewed in Table 1 relate collaboration with governance actors. Lack of governance can be an overall barrier (Muegge, 2011; timreview.ca/article/495), but one of the main barriers is the disparate interests or divergent interests between actors in the ecosystem (O'Mahony and Bechky, 2008; tinyurl.com/lothrqs). Other relevant barriers include the lack of vision and standards relating to the joint efforts in the ecosystem (Kshetri and Schiopu, 2007; tinyurl.com/n74oeem; Skerrett, 2009; timreview.ca/article/219). A lack of openness and transparency (Smith and Milinkovich, 2007; timreview.ca/article/94) can also be a barrier that restricts collaboration in an open source ecosystem. Governance actors manage different interests, solidify and converge interests, and overall reduce business differences between actors in an open source ecosystem (O'Mahony and Bechky, 2008; tinyurl.com/lothrqs). Openness and transparency (Skerrett, 2009; timreview.ca/article/219) are also required for access to shared resources and gaining commitment from the actors (Smith and Milinkovich, 2007; timreview.ca/article/94). Collaboration with governance actors is a process of compromise to establish an ecosystem structure that enables business activities (O'Mahony and Bechky, 2008; tinyurl.com/lothrqs). A vision and standards are required for international collaboration (Kshetri and Schiopu, 2008; tinyurl.com/n74oeem); they enable the actors in an open source ecosystem to create value and collaborate.

2. Competitors

Three articles summarized in Table 1 relate to open source collaboration with competitors. Barriers to overcome when collaborating with competitors include a lack of trust, the need to identify shared objectives, and the need to share access to resources in contrast to a closed approach where a company looks to assimilate key resources into its business (Shamsuzzoha et al., 2013; tinyurl.com/nnvmcr2). Other barriers include concerns related to releasing confidential information

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

Table 1. A summary of open source collaboration literature relevant to open source ecosystems

Stream	Author (Year)	Source	Open Source Actor(s)	Focus
Governance Actors	Muegge (2011) timreview.ca/article/495	*Technology Innovation Management Review (TIM Review)*	Community members, developer, foundation, users, and adopters	A systems perspective on communities, institutions, companies, and individuals
	Smith & Milinkovich (2007) timreview.ca/article/94	*TIM Review*	Foundation	Openness, transparency, and meritocracy with resources and commitment; ability of the ecosystem to create value
	O'Mahony & Bechky (2008) tinyurl.com/lothrqs	*Administrative Science Quarterly*	Foundation (boundary organizations)	Managing the boundaries of collaboration where disparate and convergent interests of different actors are present. Recommendations include the need to identify critical differences and establish a governance structure, membership, ownership, and control over the project for enabling collaboration.
	Kshetri & Schiopu (2007) tinyurl.com/n74oeem	*Journal of Asia-Pacific Business*	Foundation (government)	Effective international collaborations requirements, including a vision and influence on a technology trajectory, setting standards, and promotion
	Skerrett (2009) timreview.ca/article/219	*TIM Review*	Foundation	Collaborative software development with competitors in a foundation governed open source community
Competitors	Lindman & Rajala (2012) timreview.ca/article/510	*TIM Review*	Competitor	Focus interactions with the user to gain user involvement. Ensure access to important external resources rather than assimilate or build new internal resources. Take advantage of open innovation process but think about the purpose of external contributions. Make the goal of collaboration clear so that it becomes easier to collaborate and reveal confidential information where it makes business sense.
	Schreuders et al. (2011) timreview.ca/article/413	*TIM Review*	Competitor	Hybrid business model based upon an open source licensing model, which allows different users different access and collaboration based upon the selected open source licensing model. The hybrid business model removes restrictions and attracts actors to the open source community, building a resource to test and validate products.
	Shamsuzzoha et al. (2013) tinyurl.com/nnvmcr2	*International Journal of Computer Integrated Manufacturing*	Competitor	Building trust and overcoming fears relating to confidentiality and the exchange of information between competitors. This requires common objectives and rules/procedures for exchanging information and foundations for cooperation. Individual objectives must be clear and aligned to the shared objectives
	Muegge (2013) timreview.ca/article/655	*TIM Review*	Complementors, suppliers, customers, competitors, developers, and users	Identifies the actors in the open source ecosystem such as customers, suppliers, competitors, and many other stakeholders.

Continued...

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

Table 1 ***(continued)***. A summary of open source collaboration literature relevant to open source ecosystems

Stream	Author (Year)	Source	Open Source Actor(s)	Focus
Complementors	Skerrett (2011) timreview.ca/article/409	*TIM Review*	Multi-vendor, complementors, users, adopters, and contributors	License strategy is important to engage a larger portion of community. A copyleft license will generate a wider collaboration. License strategy can also earn trust and gain access to a larger portion of community. Use a vendor-neutral governance structure.
	Muegge (2013) timreview.ca/article/655	*TIM Review*	Complementors, suppliers, customers, competitors, developers, and users	Identifies the actors in the open source ecosystem such as customers, suppliers, competitors, and many other stakeholders.
Core Community	Muegge (2011) timreview.ca/article/495	*TIM Review*	Community members, developer, foundation, users, and adopters	A systems perspective on the community, institutions, companies, and individuals. Shared governance and participation are important to collaboration in an open source community.
	Evans & Wolf (2005) tinyurl.com/l9dgpea	*Harvard Business Review*	Community members, leaders, and competitors	Members belong to different organizations with no defined role or responsibility; a mix of amateurs and professionals with different skills. Competitors collaborate: must think about options and adaptability not integration and static efficiency. Build trust in the community and collaborate freely and productively. Leaders are valuable; they instruct community members, articulate clear goals, and connect people. Trust is currency and reputation is power.
	Skerrett (2011) timreview.ca/article/409	*TIM Review*	Multi-vendor, complementors, users, adopters, and contributors	License strategy is important to engage a larger portion of community. A copyleft license will generate a wider collaboration. License strategy can also earn trust and gain access to larger portion of community. Use a vendor-neutral governance structure.
	Sarker et al. (2009) tinyurl.com/l32zjuf	*IEEE Transactions on Professional Communication*	Leaders	Leadership in the community. Leaders are not reassigned; rather, they emerge from the project and are required for effective collaboration. Information systems development ability, greater contributions are identified with leadership.
	Nan & Kumar (2013) tinyurl.com/k5a8yt3	*IEEE Transactions on Engineering Management*	Developers	Size and format of a team of developers in association with the level of structural interdependency are key for effective collaboration. Positive impact on a project with a high level of structural interdependency may be achieved with centralized teams of developers and larger teams. Smaller teams required for positive impact on projects with a low level of structural interdependency; centralized teams can impact project performance on such projects.
	Colazo (2010) tinyurl.com/mwuovbm	*International Journal of Innovation Management*	Developers	Developer density is negatively associated with quality and positively associated with productivity. Centralization is positively associated with both quality and productivity. Collaborating beyond boundaries is positively associated with quality but negatively associated with productivity.
	Hemetsberger & Reinhardt (2009) tinyurl.com/qz3mszp	*Organization Studies*	Expert Users	Coat tailing is the pursuit of individual and collective needs; it requires achieving the best balance between individual and collective needs.
	Muegge (2013) timreview.ca/article/655	*TIM Review*	Complementors, suppliers, customers, competitors, developers, and users	Identifies the actors in the open source ecosystem such as customers, suppliers, competitors, and many other stakeholders.

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

(Lindman and Rajala, 2012; timreview.ca/article/510); removing restrictions that prevent open collaboration with competitors (Schreaders et al., 2011; timreview.ca/article/413); and the need to attract and build a larger community of competitors that may be leveraged (Schreuders et al., 2011; timreview.ca/article/413).

3. Complementors
Two articles summarized in Table 1 relate to collaboration with complementors. Barriers include intellectual property in the form of a right that prevents use and collaboration as well as an inability to share confidential information, lack of trust, moving from closed to open, vendor dominance, and trust (Skerrett, 2011; timreview.ca/article/409) and lack of transparency (Muegge, 2013; timreview.ca/article/655).

4. Core Community
Eight articles summarized in Table 1 relate to collaboration with the core community (i.e., users, adopters, contributors, leaders, developers and expert users). Barriers to overcome when collaborating with the core community include:

- *a lack of leadership*; leaders are not appointed but evolve out of the community, and there may be cultural differences in leadership (Sarker et al., 2009; tinyurl.com/l32zjuf). Collaboration with the core community requires identification of a leader or leaders of the project to ensure productivity and balance the needs of the actors in the core community

- *the size and format of the community*, which affects collaboration as well as the degree of centralization or decentralization (Nan and Kumar, 2013; tinyurl.com/k5a8yt3). Going beyond boundaries can be a barrier to productivity (Colazo, 2010; tinyurl.com/mwuovbm)

- *the individual needs of expert users*, which may not align with the needs of the community (Hemetsberger and Reinhardt, 2009; tinyurl.com/qz3mszp)

- *the mix of developer attributes*, such as the amount of skill or a lack of skill. In addition, a developer's role and responsibilities, which tend to be undefined in an open source ecosystem, (Evans and Wolf, 2005; tinyurl.com/l9dgpea) also create barriers to effective collaboration.

Open Source Barriers to Collaboration

Open source business shifts "the focus from the production value to the use value of the software artifact and emphasizes services and meta-services surrounding the artifact" (Feller et al., 2006; tinyurl.com/34eppr5). The open source ecosystem includes a number of different actors involved in the artifact and the services to provide a complete solution to customers. The ecosystem has little organization and order. The ecosystem may further include a platform where actors interact to create products or offer services. The platform offers value to the actors in the ecosystem but the platform also brings together many different technology, people and business relates issues (Kilamo et al., 2012; tinyurl.com/n5gnrgu). The actors are motivated by common interests or business models (Manikas and Hansen, 2013; tinyurl.com/lsrljj5). Common interests include a motivation to join the ecosystem where there is overlap in business and the actors work collectively towards a common task, asset, or resource. A business model is also a motivation where the business model in part relies on a non-differentiating common task, asset, or resource.

As we identified earlier, the collection of actors includes four main groups: i) governance actors, ii) competitors, iii) complementors, and iv) the core community. Governance actors are important and manage the boundaries of collaboration (O'Mahony and Bechky, 2008; tinyurl.com/lothrqs), which are essential to the operation of an open source ecosystem. The type of governance actor varies with the open source ecosystem and may include a foundation (O'Mahony and Bechky, 2008; tinyurl.com/lothrqs; Skerrett, 2009; timreview.ca/article/219; Smith and Milinkovich, 2007; timreview.ca/article/94), a federal government (Kshetri and Schiopu, 2007; tinyurl.com/nnvmcr2), or the community (O'Mahony and Bechky, 2008; tinyurl.com/lothrqs). The core community comprises the actors in the open source ecosystem that work on the open source project to develop and test the product (Kilamo et al., 2012; tinyurl.com/n5gnrgu). Actors in the core community include: leaders (Sarker et al., 2009; tinyurl.com/l32zjuf), developers (Nan and Kumar, 2013; tinyurl.com/k5a8yt3; Colazo, 2010; tinyurl.com/mwuovbm), users (Muegge, 2013; timreview.ca/article/655), adopters (Skerrett, 2011; timreview.ca/article/409; Muegge, 2011; timreview.ca/article/495) and expert users (Hemetsberger and Reinhardt, 2009; tinyurl.com/qz3mszp). Leaders provide overall leadership to the development portion of the community. Developers work on the open source project creating and testing the software. Users are important for providing requirements. Adopters are key to using the open source project. Expert users provide insight into the project for both present and future needs.

Governance actors resolve differences in the community to ensure the health of the ecosystem (Smith

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

and Milinkovich, 2007; timreview.ca/article/94). Resolving differences involves collaboration with all other groups. Competitors create and share value while collaborating with the core community, complementors, and other competitors. Complementors help build a larger community that can be leveraged to create and share value (Skerrett, 2011; timreview.ca/article/409); they collaborate with the core community as well as competitors and other complementors. The core community includes actors such as leaders, developers, and expert users and the associated collaboration barriers are different within the core community.

Table 2 is a summary of collaboration barriers from the literature pertaining to these four main groups, including the different types of actors within each group. The barriers in this table were identified through a close reading of the articles in the literature review. For example, the barrier relating to lack of project representation, which applies to governance actors, is derived from O'Mahony and Bechky (2008; tinyurl.com/lothrqs): "Though project members were not eager to impose a 'command and control structure onto the community', this desire for 'republics' led the projects to adapt a governance structure that established project representation and preserved pluralistic control." Similarly, the governance-actor barrier relating to challenges in defining the scope of collaboration is derived from Skerrett (2009; timreview.ca/article/219): "Determining the scope of collaboration is often the most challenging aspect of starting an open source project. The key challenge is to understand which areas of technology are core and which are non-core to the business value of that organization." Another example is the inequality barrier faced by complementors, which was derived from (Skerrett, 2011; timreview.ca/article/409): "For single-vendor-dominated communities, copyright assignment was required to allow the receiving vendor the ability to create revenue streams by implementing a dual license for the project code. MySQL is the most common example of this strategy. Unfortunately, this approach creates a revenue stream that is unique to one company. In turn, this inequality creates a barrier to involvement by other companies."

These examples show how the collaboration barriers listed in Table 2 were identified from the literature. Table 3 is a synthesis of the barriers from Table 2 to reveal which barriers are common across the groups and which barriers are unique to each group.

There are four barriers to collaboration that are common to all groups in an open source ecosystem: i) intellectual property, ii) moving from closed to open, iii) openness, and iv) a lack of transparency. The literature shows that trust is important to most groups in the ecosystem with the exception of the governance actors. A particular challenge for competitors and the core community is the diverse mix of people, and the potential for undefined roles and responsibilities. A particular challenge for complementors and the core community is inequality in the ecosystem. The governance actors and the core community tend to have a broad range of different barriers to collaboration.

Recommendations for Entrepreneurs and Managers

We offer six recommendations for entrepreneurs and managers seeking to overcome collaboration barriers for successful collaboration in an open source ecosystem.

1. Identify the common and unique barriers to collaboration in your open source ecosystem

The barriers to collaboration in an open source ecosystem include barriers common to all groups of actors in the ecosystem and barriers unique to specific groups of actors. Entrepreneurs and managers need to seek out, understand, and pay attention to these very different barriers for successful collaboration in an open source ecosystem. The specific nature of a given barrier will depend on the unique circumstances of your ecosystem, but Table 3 can help you systematically identify the types of common and unique barriers to collaboration.

2. Strike a balance between open and closed

Intellectual property that is not differentiating to a business should be released into the open. This requires early and ongoing identification of assets and information that may be open to the ecosystem and other assets and information that should be kept confidential. This includes patents, copyrights, designs and potential inventions. Trademarks and know-how become more valuable and strategic to open source business.

Move quickly and make informed business decisions in the open environment of the ecosystem and be open and transparent with competitors based upon your business decisions when collaborating with any group of actors in an open source ecosystem.

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

3. Seek representation and effective governance
Understand the collaboration barriers that relate to representation, pluralistic control, economic balance, disparate and divergent interests, ability to cooperate, vision, and standards. Then, ensure your business has a fair share of representation and become proactive with pluralistic control. Identify different interests early and compromise when collaborating with other groups of actors. Setting a clear vision and standards will assist you with effective collaboration with the governance group of actors.

4. Collaborate effectively with the core community
Understand the barriers with the core community that relate to equality, leadership, team structures, cultural differences, and new member integration. Be fair and equitable with the core community. Attempt to identify the leader, or assist developing a leader in the community. Ensure you have an appropriate team structures to assist collaboration with the core community and pay attention to cultural differences associated with a global community, and assist with integrating new members into the community.

5. Compete and collaborate with competitors
Understand the collaboration barriers with competitors that relate to objectives, roles, responsibilities, and shared values. In advance of engaging competitors in the open source ecosystem, ensure you have clear objectives and defined roles, responsibilities, and values. Make a point of understanding your competitor's objectives, roles, responsibilities, and values. Ensure or negotiate an appropriate understanding of joint roles, responsibilities, and values for collaborating in the open source ecosystem.

6. Recognize the challenges of diversity
Be aware of the diversity in the competitor and core community groups; it can be beneficial, but it also introduces collaboration challenges. Diversity includes different levels of knowledge or education and different levels of skills. There may be actors from many different parts of the world, which creates the potential for cultural barriers to collaboration.

Conclusion

There are many barriers to collaboration in an open source ecosystem. Some of these barriers are common to all actors in the ecosystem, but others are unique to specific groups of actors in the ecosystem. Barriers common to all groups are: intellectual property, business- and people-related issues when moving from a closed system to an open system, understanding where and how to be open in business with other actors in the ecosystem including competitors, and a potential lack of transparency that can impact success. Effective collaboration requires an open approach to all groups in the ecosystem based upon informed business decisions and an understanding of the barriers in an open source ecosystem. Effective collaboration requires that entrepreneurs and managers identify and understand the collaboration barriers both common and unique to each of the four groups of actors in the ecosystem and then overcome these barriers.

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

Table 2. Open source ecosystem actors, sub-groups, and collaboration barriers

Category	Collaboration Barriers*	Specific Actors
Governance	Lack or type of governance (Muegge, 2011; O'Mahony & Bechky, 2008)	Foundation, community
	Lack of transparency (Smith & Milinkovich, 2007; Skerrett, 2009)	Foundation
	Intellectual property concerns (Muegge, 2011; Skerrett, 2009)	Foundation
	Moving from closed to open (Smith & Milinkovich, 2007)	Foundation
	Disparate interests; divergent interests (O'Mahony & Bechky, 2008)	Foundation, community
	Lack of project representation (O'Mahony & Bechky, 2008)	Foundation, community
	Pluralistic control (O'Mahony & Bechky, 2008)	Foundation, community
	Differences preventing collaboration (O'Mahony & Bechky, 2008)	Foundation, community
	Challenges in defining the scope of collaboration (Skerrett, 2009)	Foundation
	Reluctance to be openness (Skerrett, 2009; Muegge, 2013)	Foundation
	Economic imbalance (Kshetri & Schiopu, 2007)	Foundation, government
Complementors	Lack of transparency (Muegge, 2013)	
	Initial lack of trust (Skerrett, 2011)	
	Inequality (Skerrett, 2011)	
	Moving from closed to open and developing in the open (Skerrett, 2011)	
	Selecting the best open license strategy for wider collaboration that does not introduce a high business risk (Skerrett, 2011)	
Competitors	Reluctance to reveal confidential information (Lindman & Rajala, 2012)	
	Lack of clear goals and objectives of collaboration (Lindman & Rajala, 2012)	
	Restrictive licensing model (Schreuders et al., 2011)	
	Initial lack of trust (Evans & Wolf, 2005; Shamsuzzoha et al., 2013)	
	Inability to cooperate (Shamsuzzoha et al., 2013)	
	Lack of clearly aligned and shared objectives (Shamsuzzoha et al., 2013)	
	Lack of clear, well-defined formalized roles (Shamsuzzoha et al., 2013)	
	Unwillingness to share knowledge and competencies (Shamsuzzoha et al., 2013)	
	Lack of shared values (Shamsuzzoha et al., 2013)	
	Mix of amateurs and professionals (Evans & Wolf, 2005)	
	Mix of skills and levels of skills (Evans & Wolf, 2005)	
	Members with no defined role or responsibilities (Evans & Wolf, 2005)	
Core Community	Difficulties in relationship building (Nan & Kumar, 2013)	Developer
	Lack of a formal team structure (Nan & Kumar, 2013)	Developer
	Degree of centralization (Nan & Kumar, 2013; Colazo, 2010)	Developer
	Dealing with volunteer members (Colazo, 2010)	Developer
	Lack of transparency (Muegge, 2013)	Developer, user, adopter
	Intellectual property concerns (Muegge, 2011)	User, adopter
	Initial lack of trust (Skerrett, 2011; Evans & Wolf, 2005)	User, adopter, leader
	Inequality (Skerrett, 2011)	User, adopter
	Moving from closed to open and developing in the open (Skerrett, 2011)	User, adopter
	Selecting the best open licensing strategy for wider collaboration that does not introduce a high business risk (Skerrett, 2011)	User, adopter
	Negative selfish interest of users (Muegge, 2013)	User
	Dispersed work (Hemetsberger & Reinhardt, 2009)	Expert user
	Mix of different skills expertise (professional and amateur) (Hemetsberger & Reinhardt, 2009; Evans & Wolf, 2005)	Expert user, leader
	Challenges in integrating new members (Hemetsberger & Reinhardt, 2009)	Expert user
	Members without defined roles or responsibilities (Evans & Wolf, 2005)	Leader
	Leadership in a virtual environment (Sarker et al., 2009)	Leader
	Perception of leadership (Sarker et al., 2009)	Leader

*See Table 1 for links to cited articles.

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

Table 3. Barriers to collaboration in an open source ecosystem, by actor group

	Actor Group			
Potential Barriers to Collaboration	**Governance**	**Complementors**	**Competitors**	**Core Community**
Intellectual property concerns	✓	✓	✓	✓
Moving from closed to open and sharing information and knowledge with everyone in the ecosystem	✓	✓	✓	✓
Reluctance to be open	✓	✓	✓	✓
Lack of transparency	✓	✓	✓	✓
Lack of trust		✓	✓	✓
Challenges of interacting with a diversity of people with different knowledge and experience			✓	✓
Lack of clearly defined roles/responsibilities			✓	✓
Inequality		✓		✓
The business' lack of representation in the ecosystem	✓			
Pluralistic control	✓			
Economic imbalance between actors of different size	✓			
Disparate/divergent interests	✓			
Inability to cooperate	✓			
Lack of vision/standards	✓			
Lack of clear objectives			✓	
Lack of shared values			✓	
Lack of leadership				✓
Lack of a format team structure to achieve a project objectives (size/format/degree of centralization or decentralization)				✓
Cultural differences				✓
Challenges with integrating new members into the ecosystem, team, or project				✓

Overcoming Barriers to Collaboration in an Open Source Ecosystem

Derek Smith, Asrar Alshaikh, Rawan Bojan, Anish Kak, and Mohammad Mehdi Gharaei Manesh

About the Authors

Derek Smith is the founder and principal of Magneto Innovention Management, an intellectual property consulting firm that assists entrepreneurs and small businesses with difficult intellectual property issues. He has over 20 years of experience working as an intellectual property management consultant and patent agent for IBM Canada, Bell Canada and, most recently, Husky Injection Molding Systems where he was Director, Global Intellectual Property. Prior to entering the field of intellectual property, he was an advisory engineer at IBM Canada where he was involved in a variety of leading-edge software development projects. Derek is currently a graduate student in the Technology Innovation Management (TIM) program at Carleton University in Ottawa, Canada. He also holds a BEng degree in Systems and Computer Engineering from Carleton University and is a registered patent agent in both Canada and the United States.

Asrar Abdulqader Alshaikh is a graduate student in the Technology Innovation Management (TIM) program at Carleton University in Ottawa, Canada. She holds a Bachelor of Accounting degree from King Abdulaziz University in Jeddah, Saudi Arabia. Her work experience includes customer service in a sale for distribution and communication company as well as working for the Alahli Bank (NCB) in Jeddah, Saudi Arabia. Her main area of research interest is collaborative consumption.

Rawan Mohammad Bojan is a graduate student in the Technology Innovation Management (TIM) program at Carleton University in Ottawa, Canada. She has professional experience in the banking industry and holds a Bachelor of Science in Accounting from King Abdulaziz University in Jeddah, Saudi Arabia.

Anish Kak is a graduate student in the Technology Innovation Management (TIM) program at Carleton University in Ottawa, Canada. He holds a BEng degree in Computer Science Engineering, from Birla Institute of Technology in India. Anish has two years of experience in the information technology services sector, which he gained while working for Hewlett-Packard in India. His research interests include the electronic sports ecosystem.

Mohammad Mehdi Gharaei Manesh is a graduate student in the Technology Innovation Management (TIM) program at Carleton University in Ottawa, Canada. He holds an MBA degree from Carleton University's Sprott School of Business and also has a degree in Biomedical Engineering from Tehran Polytechnic University in Iran. He has 5 years of working experience in a medical equipment company and his main area of interest relates to crowdsourcing and international business.

Citation: Smith, D., A. Alshaikh, R. Bojan, A. Kak, and M. M. G. Manesh. 2014. Overcoming Barriers to Collaboration in an Open Source Ecosystem. *Technology Innovation Management Review.* January 2014: 18–27.

Keywords: business ecosystem, open source, communities, governance, core community , competitors, complementors, collaboration, collaboration barriers

TIM Lecture Series
The Business of Open Source

Michael Weiss

> *"Today, if you don't think of open source as part of your technology business, you are doing something wrong."*
>
> Michael Weiss
> Associate Professor, Carleton University

Overview

The TIM Lecture Series is hosted by the Technology Innovation Management program (carleton.ca/tim) at Carleton University in Ottawa, Canada. The lectures provide a forum to promote the transfer of knowledge from university research to technology company executives and entrepreneurs as well as research and development personnel. Readers are encouraged to share related insights or provide feedback on the presentation or the TIM Lecture Series, including recommendations of future speakers.

The seventh TIM lecture of 2013 was presented by Michael Weiss, Associate Professor in the Technology Innovation Management program at Carleton University, who examined the business of open source, with a focus on common patterns followed by open source businesses. The event was held at Carleton University on December 12th, 2013.

In the first part of his lecture, Weiss provided an overview of the business of open source, described the key elements of early-stage open source businesses, and presented common patterns followed by open source businesses. In the second part of the lecture, he closely examined late-stage open source businesses, the impact of licensing and architecture, and what the future may hold for open source. The slides from this lecture are available here: tinyurl.com/loz5yof

Summary

Open source has become an integral part of commercial software development. Whereas in the past, open source software development was considered to be driven by volunteer effort, today most of it is carried out by companies. How companies leverage open source ranges from the adoption of open source development practices, the use of open source development tools, and the integration of open source components into products to active contributions to existing open source projects, and the initiation of their own company-led open source projects. Open source furthermore enables companies to collaborate on the creation of common assets that they can jointly use in product development.

An open source business is a business built around an open source offer. Thus, open source is not by itself a business model; rather, it used by the business as a strategy to strengthen its business model (Bailetti, 2009; timreview.ca/article/226). Typically, open source businesses use open source to: i) develop new products; ii) build products or services around open source offers; iii) initiate their own open source projects; or iv) leverage open source as a form of co-opetition (i.e., cooperation with competitors). Indeed, close study of the way in which businesses have leveraged open source has led to the identification of common patterns used by open source businesses.

Patterns are proven solutions to common problems, and have been popular in the fields of architecture and software design. More recently, they have also been used to document business strategies, including those used by open source businesses. The patterns that Weiss described in this lecture aimed to provide entrepreneurs, managers, and students of business models with a language for creating new business models around open source, and for incorporating open source into existing business models. Some of these patterns are unique to the stage of a company's engagement

TIM Lecture Series – The Business of Open Source

Michael Weiss

with open source (i.e., from primarily just using open source assets, to contributing to projects, to championing particular development projects, and finally to collaborating strategically within the ecosystem), whereas a given pattern relating to licensing and architecture typically may be used in all stages (Figure 1).

Examples of open source business patterns discussed in the lecture include:

1. **Bootstrap:** This use-stage pattern refers to the reuse of existing open source components to develop products. This pattern allows a company to shorten the time it takes to create a first version of their product, while keeping costs low; however; it also increases the complexity of the software and the breadth of knowledge needed.

2. **Contribute back:** By contributing resources (e.g., code, people, money) to the projects they use, companies can: i) build trust with the community, ii) influence the development of the project, and iii) demonstrate their competence.

3. **Credible promise:** When championing a project, building a critical mass of functionality into the project from an early stage can mobilize contributors to a company's project – it helps the company demonstrate that the project is doable and has merit.

4. **Feed the community:** A company can build legitimacy with a project community by nurturing the community without any expectation of immediate return, for example by: i) giving to the community (e.g., contributing code, writing documentation, participating in the discussion forum); ii) establishing a clear licensing practice; iii) establishing a clear process for making contributions; iv) making decisions in the open; and v) not treating community members as prospects.

5. **Sell complements:** An open source product can be monetized through the selling of services (e.g., hardware or support) that complement the open source product.

6. **Run a tight ship:** To keep control of a project's direction, companies often retain full ownership of the code.

7. **Dual product (open core):** To entice commercial users to pay for an open source software product, a company may sell a commercial version of the open source produce with exclusive features.

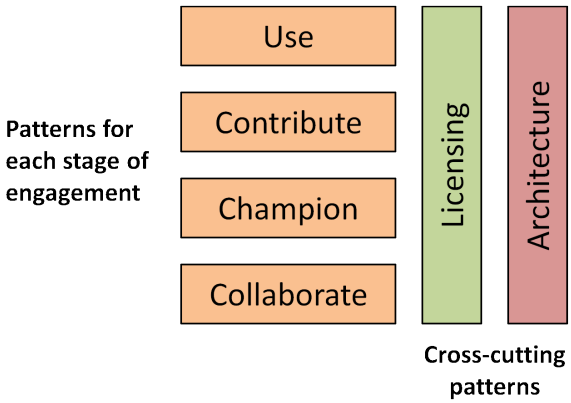

Figure 1. Open source business patterns

8. **Pool resources:** To optimize the use of resources in the collaboration stage, a company may jointly develop a common stack of open source assets with other companies. Each company can then develop their own differentiated products based on these common assets.

9. **Foundation:** A company can attract other companies to contribute its open source project by transferring ownership of the code to an independent foundation. A foundation creates an arms-length relationship between the project creator and the project itself, and it centralizes common functions that all members can access (e.g., legal, marketing, project management). Creating a foundation builds trust and facilitates collaboration among the contributors.

Examples of patterns related to licensing include:

1. **Play by the rules:** The risk of a licensing problem arising after a product is released can be mitigated by ensuring license compliance for open source components that are combined.

2. **Dual license:** To encourage commercial users to pay for an open source software product, a company can offer the same product under two licenses: commercial and open source. Non-commercial users still benefit from the rights under the open source license, and buyers of the commercial license are released from some of the obligations of the open source license.

3. **IP modularity:** Alignment of its intellectual property with the product architecture can help a company manage the complexities of having both open source and proprietary versions of a dual product.

TIM Lecture Series – The Business of Open Source

Michael Weiss

Examples of patterns related to architecture include:

1. **Modular architecture:** External contributors may have difficulty contributing if they require deep knowledge of the project. A company can overcome this problem by partitioning the code base so that different parts (or modules) can be worked on and managed independently.

2. **Manage complements:** A governance model and regulatory tools can help manage the quality of complements developed by the community.

To conclude the lecture, Weiss speculated about what the future might hold for open source. The discussion focused on the shift away from desktops and servers onto what are – for open source – non-traditional platforms of mobile, embedded systems, and hardware. The role of open source in the "Internet of Things" (tinyurl.com/5qr2nq) was also explored. Finally, Weiss reflected upon the ongoing growth of "open" beyond software (e.g., open hardware, open data, and open science).

Lessons Learned

In the discussions that followed each portion of the presentation, audience members shared the lessons they learned from the presentation and injected their own knowledge and experience into the conversation.

The audience identified the following key takeaways from the presentation:

1. Patterns help you recognize and find solutions beyond your expertise, and they help you collaborate

2. There are now real and practical examples of companies employing these strategies – open source business is now a well-travelled road.

3. Different patterns are suited to different contexts/opportunities.

4. Although open source can reduce costs, the increased complexity it often brings is a challenge.

5. The value to companies is significant if they share R&D for non-core development; they can then focus their efforts on what makes them different from the competition.

6. Companies go though phases in their engagement with open source. They start out as users, then start contributing. Eventually, they may champion their own project and then collaborate with other companies, perhaps even by forming a foundation. It's a learning curve.

7. Community management only works when you have appropriate governance in place.

8. The value of the code is a function of: i) how quickly you can learn about it; ii) how modular it is; and iii) how many applications it can serve. These factors determine how quickly you can ramp up.

9. Companies have options and can make conscious choices about how to use open source in their strategies and business models.

10. The value of the disruption that open source represents is threefold: i) it allowed companies to make money in new ways; ii) it enabled software to be produced and distributed in completely new ways; and iii) it reduced the barriers to entry while increasing the potential for collaboration.

11. There is a trend towards the creation of open source toolsets that allow for the early and simple creation of applications. This technical capability enables others to quickly solve real-world domain-specific problems.

12. In the future, we need new ways of innovating around business models. The landscape is now different; we need to look forward and not worry about what did or did not work in the past.

TIM Lecture Series – The Business of Open Source

Michael Weiss

Additional Resources

1. Slides from this lecture: tinyurl.com/loz5yof

2. Related papers by Michael Weiss:

- "Performance of Open Source Projects" (2009; tinyurl.com/l2zf8qz)
- "Profiting from Open Source" (2010; tinyurl.com/m974nya)
- "Profiting Even More from Open Source" (2011; tinyurl.com/ktt37jy)

This report was written by Michael Weiss and Chris McPhee.

About the Speaker

Michael Weiss holds a faculty appointment in the Department of Systems and Computer Engineering at Carleton University in Ottawa, Canada, and is a member of the Technology Innovation Management program. His research interests include open source, ecosystems, mashups, patterns, and social network analysis. Michael has published on the evolution of open source business, mashups, platforms, and technology entrepreneurship.

Citation: Weiss, M. 2014. TIM Lecture Series – The Business of Open Source. *Technology Innovation Management Review*. January 2014: 28–31.

Keywords: open source software, business models, patterns, engagement, entrepreneurship, licensing, architecture, community

Author Guidelines

These guidelines should assist in the process of translating your expertise into a focused article that adds to the knowledge resources available through the *Technology Innovation Management Review*. Prior to writing an article, we recommend that you contact the Editor to discuss your article topic, the author guidelines, upcoming editorial themes, and the submission process: timreview.ca/contact

Topic

Start by asking yourself:

- Does my research or experience provide any new insights or perspectives?
- Do I often find myself having to explain this topic when I meet people as they are unaware of its relevance?
- Do I believe that I could have saved myself time, money, and frustration if someone had explained to me the issues surrounding this topic?
- Am I constantly correcting misconceptions regarding this topic?
- Am I considered to be an expert in this field? For example, do I present my research or experience at conferences?

If your answer is "yes" to any of these questions, your topic is likely of interest to readers of the TIM Review.

When writing your article, keep the following points in mind:

- Emphasize the practical application of your insights or research.
- Thoroughly examine the topic; don't leave the reader wishing for more.
- Know your central theme and stick to it.
- Demonstrate your depth of understanding for the topic, and that you have considered its benefits, possible outcomes, and applicability.
- Write in a formal, analytical style. Third-person voice is recommended; first-person voice may also be acceptable depending on the perspective of your article.

Format

1. Use an article template: .doc .odt
2. Indicate if your submission has been previously published elsewhere. This is to ensure that we don't infringe upon another publisher's copyright policy.
3. Do not send articles shorter than 1500 words or longer than 3000 words.
4. Begin with a thought-provoking quotation that matches the spirit of the article. Research the source of your quotation in order to provide proper attribution.
5. Include a 2-3 paragraph abstract that provides the key messages you will be presenting in the article.
6. Only the essential references should be included. The URL to an online reference is preferred; where no online reference exists, include the name of the person and the full title of the article or book containing the referenced text. If the reference is from a personal communication, ensure that you have permission to use the quote and include a comment to that effect.
7. Provide a 2-3 paragraph conclusion that summarizes the article's main points and leaves the reader with the most important messages.
8. Include a 75-150 word biography.
9. If there are any additional texts that would be of interest to readers, include their full title and location URL.
10. Include 5 keywords for the article's metadata to assist search engines in finding your article.
11. Include any figures at the appropriate locations in the article, but also send separate graphic files at maximum resolution available for each figure.

Issue Sponsor