A Systematic Review of Research on Open Source Software in Commercial Software Product Development

Martin Höst
Department of Computer Science
Lund University
P.O. Box 118, SE-221 00, Lund, Sweden
serg.cs.lth.se
Martin.Host@cs.lth.se

Alma Oručević-Alagić
Department of Computer Science
Lund University
P.O. Box 118, SE-221 00, Lund, Sweden
serg.cs.lth.se
Alma.Orucevic-Alagic@cs.lth.se

Background: **The popularity of the open source software development in the last decade, has brought about an increased interest from the industry on how to use open source components, participate in the open source community, build business models around this type of software development, and learn more about open source development methodologies.**
Aim: **The aim of this study is to review research carried out on usage of open source components and development methodologies by the industry, as well as companies' participation in the open source community.**
Method: **Systematic review through searches in library databases and manual identification of articles from the open source conference.**
Results: **19 articles were identified.**
Conclusions: **The articles could be divided into four categories: open source as part of component based software engineering, business models with open source in commercial organization, company participation in open source development communities, and usage of open source processes within a company.**

open source software, proprietary, commercial, component based software engineering, business models

1. INTRODUCTION

The traditional software development is often perceived as a proprietary, in-house software development, with developers working in a geographically centralized or distributed company's location. Open source software is developed free of charge through a community driven development process, and as such, it is also provided to public at no cost, but under certain usage and distribution conditions. Many of the traditional software companies have tried to take advantage of the free software, not just by using the software, but also by creating quite sophisticated business models and strategies around the open source software.

For example, in the mobile industry there are several attempts to form open source communities for development of software, such as the Android project and the Symbian project. While the Android project consists of many independent open source projects, its core packages were developed in a proprietary environment and then open sourced. The project has created a community of leading mobile phone manufactures and operators, open handset alliance, which along other community members, participates in development of the Android operating system. This example shows how using and relying on open source software can be seen as an alternative way to reduce development costs and stay competitive. Hence, in a way, it can be compared to other similar business methods and strategies, such as outsourcing or acquirement of off the shelf components.

This open source business ecosystem, which has been growing over the past two decades, is quite complex and there exists a need to better understand many of its aspects. Some of the aspects are interesting in (at least) two different ways. Firstly, an organization can include open source components in its proprietary software product. This is comparable to including any other third party component, although the difference is that the component is now obtained from an open source community instead from a commercial organization. Secondly, an organization can provide its own proprietary software to open source community and that way reduce development costs in long run, reposition itself on the market, create a new source of income through new services, etc.

Already in 2001, Lerner and Tirole (2001) identified ”opening proprietary code” as an important research area, and observed that large open source projects often start based on software provided by ”academic or semi-academic institutions”. This motivates systematically investigating what research has been published in the area.

The outline of this paper is as follows. In Section 2 background on open source software and some related work is presented. In Section 3 the methodology with respect to search strategy and inclusion/exclusion criteria are presented, while the resulting set of articles is presented in Section 4. Finally, there is a discussion in Section 5, and conclusions presented in Section 6.

2. BACKGROUND AND RELATED WORK

2.1. Open Source Software

Open source software has been around since the very beginning of electronic computing. In the early days of information technology it was quite natural and financially sound for developers to share source code among very few and very expensive computing machines. As the machines became smaller, more diversified, and cheaper, the number of developers grew, and the source code, in general, became more complex. Development of free software was especially flourishing in the academic environments. Barkley Software Distribution (BSD) is license developed for distribution of BSD version of Unix operating system developed by University of California Berkeley from 1977 to 1995 in collaboration with AT&T labs, as described by Raymond (2001). At the beginning of the development, code was shared between AT&T and Berkeley. Due to a divestment in 1984 it became a proprietary AT&T product.

Since the beginning of 1980s, the idea of close-sourced/proprietary software became mainstream, taking the place that free software sharing has held for a long time. The open source supporters went to found their own organizations such as free software foundation (FSF) founded by Richard Stallman, as described by Weber (2004). The FSF did not have desired impact on bringing back open source software development to the mainstream. However, this situation was about to change with the successful release of Linux kernel. The system initially developed by the Linus Torvalds as a part of academic project, with the support of the developers community came to produce very complex, sophisticated software that was free for everyone to use. Eric Raymond was very much inspired by this set of events, that in his now famous book ”The Cathedral and the Bazaar” he talks about the importance of Linux, as it was the very first time the open source developer community showed that not only complex and sophisticated software can be built in such way, but also that business models can be built around such way of software development and distribution.

In 1998, Raymond was one of the main contributors to the Open Source Initiative (OSI), an organization that is envisioned as open source educational and advocacy organization. Many companies have followed the suit, and decided to open source a piece of their proprietary software as a part of business strategy to deal with the competition. Thus, among the initial suitors we can find Netscape corporation, who by open sourcing Netscape internet browser tried to compete against closed source and free distribution of Microsoft’s Internet Explorer (Raymond (2001)).

In the past 10 years, many companies have entered the open source business arena, using some of the business models proposed by Raymond (2001). Unfamiliar with the environment, companies had very quickly to readjust their way of doing business in order to ripe some perceived benefits of open source trends. Besides open sourcing software, companies tend to participate and contribute to open source projects, as well as adopt some of software development methodologies such distributed and voluntary based development community as open source utilizes.

2.2. Related Work

Stol and Ali Babar (2009) have made a review of the broad area of ”open source” from the conference on Open Source Systems, OSS. They manually selected empirical papers from the conference and investigated them. The scope of the review that we present in this paper is more narrow (open source in commercial organizations) but we searched a broader set of articles (we searched articles in library databases with a search string and we selected articles from the OSS conference manually, as explained in more detail below).

3. REVIEW METHOD

This research is carried out as a systematic literature review, as described by Kitchenham and Charter (2007).

3.1. Research Questions

The objective of this research is to investigate what research has been carried out on the usage of open source software and open source software development in proprietary software development organizations. Before the review, this was broken down to the following research questions:

1. What approaches and processes are applied by commercial organizations to introduce open source products in their proprietary products?
2. What approaches and processes are applied by commercial organizations to provide their software products to the open source community?
3. What experience is available from identified approaches and processes, for example, with respect to quality of the software products, cost of development for the providing organization, time taken to introduce new functionality, etc.?
4. What are the main motivations and business incentives for the procedures and processes identified in question 1 and question 2?

It should be noted that the objective of the research has not been to derive quantitative knowledge of which methods perform the best. If the field was more mature, and it could be expected to find a large number of empirical studies investigating the performance of alternative methods, it would of course be interesting to synthesize this knowledge. However, it is not realistic to find this many studies of this type. The objective of this work is instead to survey the research that has been conducted, and in particular what kind of experience that is available for these kind of questions. This is seen as a reasonable step at this stage, and the objective is to identify important research questions for the future based on these findings. That is, the research has elements of a mapping study (e.g. Kitchenham and Charter (2007)). Research questions for the future are formulated based on information in the identified papers, and based on the difference between the formulated research questions above and the available research results that are found.

Open source software is related to a number of research questions that to some extent are relevant to the survey, but where it was necessary to decide whether to include them in the study or not. One aspect that is often mentioned concerning open source is the importance of ”legal aspects”, such as licensing, intellectual property, etc. For example, there is a large number of licenses, all complying with the definition of open source by Feller and Fitzgerald (2002), and the implications of choosing different licenses could be an important research field. This is an important and interesting field, which can affect both the adoption of open source practices and open source software components. However, for this study it was seen as out of scope for two main reasons. Even if software engineering is a multi-disciplinary area which includes legal aspects, we thought that it is of another kind than more traditional software engineering topics. To some extent the research questions that have to do with legal aspects are not the same as traditional research questions. If legal aspects were included, then there are other areas that also would be reasonable to include, such as marketing and sales. Second, it would probably require extensive cooperation with researchers in legal aspects to make sure that the correct search terms were used, and that the right publication fora were searched. These aspects in combination mean that legal aspects were not included in the study.

An area where there are a number of research results available is on comparisons of usage of open source software, such as Open Office, and similar proprietary software systems. This was not seen as highly related to the research questions in this study and therefore excluded. It is of course interesting to know about the differences, but it was not seen as relevant enough to the question of transforming developed software to open source or to the inclusion of open source software in developed software. Neither are studies on adoption of open source programs, for example as presented by Goode (2005), included. That is, this study is more on development of software than on the usage of existing software. In the same way it was decided not to include research results on usage of open source tools, such as Eclipse, in software development.

3.2. Search methodology

Two main sources were searched for relevant articles: a broad search in academic databases; and a manual search through all articles of the Conference on Open Source Systems.

3.2.1. Searched academic databases

First, the INSPEC and the COMPENDEX databases were searched. Both of these databases intend to provide complete coverage of the area, and include articles from all major conferences, journals, and publishers (e.g. IEEE, ACM, Springer, and IEE). We believe that these two databases give a good coverage of articles in ”computer science” and ”electrical engineering and electronics”, which includes typical questions in software engineering, at least in more well known journals and conferences. However, the coverage of more business-related articles and articles on legal aspects is, as described above, more uncertain. Both databases were accessed through Engineering Village (`http://www.engineeringvillage2.org`).

In order to conduct the search in the databases the following search string was formulated:

```
(({open?source} wn ALL) OR
 (opensource wn ALL) OR
 (libre wn ALL) OR
 (OSS wn ALL) OR
 (FLOSS wn ALL))
AND
((proprietary wn ALL) OR
 (commercial wn ALL) OR
 ({non?open?source} wn ALL) OR
 ({non?opensource} wn ALL))
AND
((empirical* wn ALL) OR
 (experiment* wn ALL) OR
 ({case?study} wn ALL) OR
 (survey wn ALL))
```

The search string contains three main parts, separated by AND-clauses. The first part states that the article should include the term ”open source” or some other synonym term that is often used, such as ”OSS”. The second part states that the article should include terms about commercial software development. The third part makes sure that the article is empirical, by looking for terms like ”empirical” and ”experiment”. The intention of the ”*” after experiment is to include also search terms as ”experimental” and ”experimentation”. According to Dieste et.al. (2007) this should be sufficient in order to find most relevant articles in this respect.

A few more terms and details in the search string may have to be explained. The ?-sign denotes any character, which, for example, means that both articles with the term ”case study” and the term ”case-study” are found. The term wn means that the phrase left of it should be found in the entity to the right of it, in this case ALL, which means all fields of database entries, such as title, abstract and key words. It would have been possible to list other entities such as abstract and title, but in this case ALL was chosen. Text within {}-parentheses are searched as phrases and a search is not case sensitive.

3.2.2. *Manual identification of relevant articles*

In addition to the database searches, all articles in all OSS-conferences (International Conference on Open Source Systems[1]) were inspected. The conference has been held for 5 years (2005-2009) and all articles are available in full text (2005 online and 2006-2009 from library databases). The selection was based on the formulated research questions in Section 3.1, which thereby means that articles matching the same kind of content as the identified with search string presented in Section 3.2.1 were found, but there was no check that the articles matched exactly.

3.3. Selection of articles

Articles are selected in a number of steps. First, all articles identified from the databases with the search string were listed with title and abstract. Based on this, articles that are not relevant were removed. After this, the remaining articles were downloaded and read in full text. Based on this, more articles were seen as non-relevant and therefore removed.

After analysis of articles selected with the search string, articles from the OSS conference were selected manually. Since this selection was carried out after the analysis of articles identified with the search string, it was possible to use knowledge that was gained from analysis of articles identified with the search string.

All selected articles were analysed with respect to research methodology implementation and presentation. Three different classes were used:

Class A: In this type of article the research is presented in a way that makes it very likely that it was conducted according to normal requirements on empirical research methods in software engineering.

Compared to the quality assessment criteria presented used by Dybå and Dingsøir (2008) and Chen et.al (2008) the answer is positive to most evaluation questions, especially concerning whether it is research or merely experience report, if there is a clear statement of aims, if there is an adequate context description, if the research design is appropriate for the questions, if the data collection was appropriate for the questions, if the data was analyzed with sufficient rigor, and if there is a clear statement of findings.

This class includes both articles that do reference empirical software engineering research method descriptions, such as Runeson and Höst (2009), and articles that do not explicitly reference this kind of descriptions.

Class B: This class of articles may not be presented as a typical article on empirical software engineering even if the overall impression of it is that it was carried out in this way. That is, all aspects of a typical article of class A may not be included, but the main impression of the paper is that the research was carried out according to normal requirements on empirical software engineering.

Class C: For this type of article we have interpreted them as the researchers have not followed any traditional research method during the research. The reason may be that the presentation forum is not suitable for presentation of structured research methods or there may be other reasons.

These steps are further presented in Section 4.1 and Section 4.2.

3.4. Data Extraction and Synthesis

Articles of class A and class B were treated equally, while articles of class C were not further included in the review. Data from the identified articles was derived by defining categories of articles and summarizing the research in each category. Both authors first defined categories individually and then a final set of categories was defined based on discussion between the authors.

[1] For more information see IFIP Working Group 2.13, http://www.ifipwg213.org/

The summaries were developed first by one author and then updated based on discussion between the authors.

4. RESULTS

In this section the actual results of the steps presented in Section 3 are presented.

4.1. First steps

First the academic databases were searched 18 May 2009, which resulted in 357 articles. However, among these articles there were a number of duplicates because the searches were made in different databases. Duplicates were identified with a simple java-program based on titles. After this, 237 articles remained.

After this, unrelated articles were removed based on both title and abstract. First an attempt was made to remove articles based only on title and then based on abstract, but it was not seen as possible to remove an article only based on the title. Therefore both titles and abstracts were studied during this process. After this, 45 titles remained.

After this, one additional duplicate was identified where the title was written slightly differently in different databases (”&” instead of ”and”), which means that 44 titles remained.

The first author of this paper first conducted these steps, and then the second author reviewed the result. None of the previously excluded articles were reintroduced.

In these steps there were some articles for which it was hard to decide whether to keep them or not based on the title and abstract. In these cases we decided to keep them to the next step instead of removing them. That is, articles that were hard to judge based on only title and abstract were kept to the next step, where the whole articles were read.

4.2. Analysis of articles based on complete articles

When all remaining articles had been obtained from the library database, they were reviewed in full text. Here, many articles were removed since they were not really related to the research questions. Some of the removed articles were about developing open source in general, which was not seen as relevant for this work. Some were about using open source software in general, without seeing the context as an IT system that is built.

To this list of articles, relevant articles from the OSS conference were added. There was no overlap between these manually found articles and the articles that were found through the search in the databases. After this, a final set of 19 articles remained. These articles are listed and summarized shortly in Appendix.

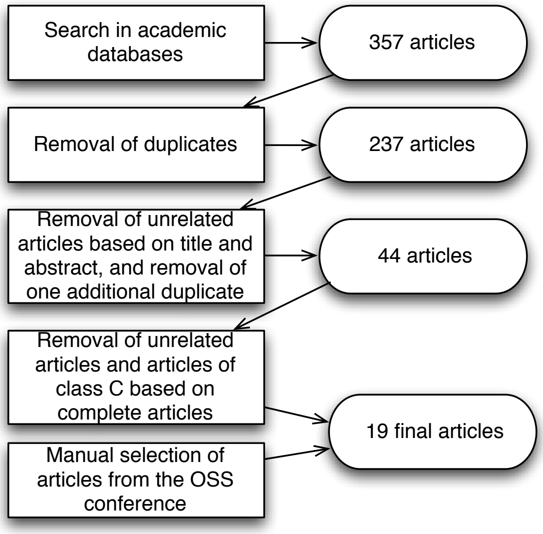
Search in academic databases
357 articles
Removal of duplicates
237 articles
Removal of unrelated articles based on title and abstract, and removal of one additional duplicate
44 articles
Removal of unrelated articles and articles of class C based on complete articles
19 final articles
Manual selection of articles from the OSS conference

Figure 1: *Summary of article selection process.*

The process of identifying and selecting related articles is summarized in Figure 1. In the rest of this paper, the selected articles are referred to with the keys that are presented in bold in Appendix. For example, the first identified article is referred to as [Arhippainen03].

In the analysis of the papers it was clear that both anticipated and unanticipated areas were covered by the identified articles. That is, some papers dealt with questions that we thought of before, and therefore were aware of when the research questions were formulated. Other areas were more unexpected, mainly the articles about transferring the open source development process to the internal work in a non-open source product. Articles of both types were of course included in the study as long as they were seen as relevant compared to the formulated research questions.

One paper for which it was hard to judge the relevance for this study is the paper by Krishnamurthy (2006), which concerns motivation of developers, to some extent discussing both unpaid and paid developers. Even if the question of motivation for open source developers in general is out of scope of the review, the discussion about paid and unpaid developers makes it more relevant. However, we decided not to include the paper since it was seen as a too small part of the article. Also, concerning the paper by Leavesley (2002) it could be argued that this type of article should be included. The main focus of it is on design of a modular system for data analysis, but they also conclude that working with the system as OSS improves the possibility of collaborating between different universities, government agencies, and private industry, both nationally and internationally. However, this was stated as a minor part of the paper, which means that the article was not included.

***Table 1:** Research methodologies*

Article	Research methodology
[Arhippainen03]	case study
[Ayala09]	survey
[Bonaccorsi05]	survey
[Bonaccorsi06]	survey
[Bonaccorsi07]	survey
[Gurbani06]	case study
[Hauge07]	survey
[Hauge09]	case study
[Li05a]	survey
[Li05b]	survey
[Li06a]	survey
[Li06b]	survey
[Li09]	summary
[Lindman08]	case study
[Lindman09]	case study
[Lundell06]	survey
[Robles07]	case study
[West03]	case study
[Westenholz06]	case study

The research methodologies that were used in the identified articles are summarized in Table 1. This is our interpretation of the chosen methodology after reading the articles. In some cases it was very clear which methodology was used, but in other cases it was a bit harder. For example when a set of interviews was conducted in different organizations we classified this as a survey, since we wanted to classify according to commonly used methods. However, it could had also been classified as something like ”interview study”.

Since no difference is made in the analysis of articles of type A and type B we make no difference between them in the presentation either.

The article [Li09] requires some further explanation. Since it is a summary of the other articles by the author ([Li05a]–[Li06b]), the methodology is presented as ”summary”, and even if it is presented as a popular science article we have classified it as class A and thereby included it in the study based on the contents of the other articles.

In Figure 2 it can be seen that the oldest identified article is from year 2003, and that the most recent is from 2009. Here it should be noted that the search in the database was conducted in mid 2009, which means that there possibly are more articles published in 2009. It is not possible to observe any clear trend concerning the number of publications, but there were rather many articles published in 2005 and 2006. Four of these were by the same author on the same subject.

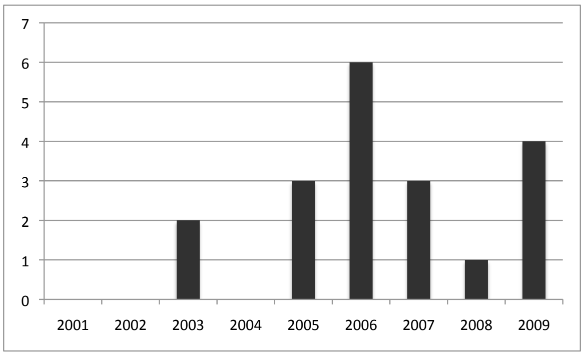

***Figure 2:** Histogram showing the publication year for included articles*

4.3. Investigated research areas

4.3.1. Introduction

The articles can be divided into a number of main areas based on the contents of the articles. The formulation of content areas was done without explicitly adhering to the identified research questions defined in Section 3. This means that each article was sorted into the category where it belonged the most even if it could be argued that some articles to some extent were related to more than one category. However, there was no article where it was really hard to decide the category or where we thought that it was equally related to more than one category. This is probably natural since the categories were defined based on the contents of the articles, and the objective during this process was to define categories based on the articles.

The identified categories are listed below. For each category the articles related to it are listed.

- *Company participation in open source development communities:* [Bonaccorsi07], [Hauge07], [Lundell06], [Robles07].
- *Business models with open source in commercial organizations:* [Bonaccorsi05], [Bonaccorsi06], [Hauge09], [Lindman09], [Westenholz06], [West03].
- *Open source as part of component based software engineering:* [Arhippainen03], [Ayala09], [Li05a], [Li05b], [Li06a], [Li06b], [Li09].
- *Using the open source process within a company:* [Gurbani05], [Lindman08].

The research conducted in each area is shortly summarized below.

4.3.2. Company participation in open source development communities

It is clear that there is company participation in many open source projects. For example [Bonaccorsi07] found that in one third of the most active projects on SourceForge there was some form of company participation. Companies can participate as project coordinator, collaborator in code development, and by providing code etc. In [Hauge07] one additional role, which is more concerned with integration of open source components, is identified.

Concerning the number of companies that participate in this kind of development, [Lundell06] suggests that a significant number of the companies marginally participate in open source community. However, the participation has increased especially in SME, compared to earlier conducted studies. Of the companies that use open source projects, 75% can be said to have ”symbiotic relationship” with the OS community. This can be compared to the investigation presented by [Rables07] that show that 6-7% of the code in Linux Debian GNU distribution over the period 1998-2004 has been contributed by corporations. That is, it is clear that a rather large part of the open source code has been provided by commercial organizations, and that those commercial organizations play crucial roles in open source projects. This is especially clear in the larger and more active projects.

It is also clear that if software should be provided to a community it is important to provide enough documentation and information to get the community members going (e.g. [Hauge07]).

[Bonaccorsi07] presents a list of important questions for further research, which is relevant with respect to these questions. For example, are companies participating in open source projects more successful than other companies, and what are the characteristics of companies participating in open source projects? It is worth noting that no identified paper presents much research about how companies' internal processes for collaborating with communities work. This could also be an area for further research.

4.3.3. Business models with open source in commercial organizations

Concerning the business models it is clear that companies involved in open source development besides developing open source products also offer customized software based on open source products. It is also common to offer consulting and training (e.g. [Bonaccorsi06]). It is also clear that business models include hybrid strategies, such as described by [West03], where the focus is on large software vendors. The paper presents an in-depth analysis on historical development of operating systems, computers, and business strategies adopted by vendors. [Hauge09] presents a case study on a small Norwegian company that successfully established a business model around two open source products by establishing three specialized user communities. The paper also concludes that while it is important to attract developers to the community, it is as important to retain some control over the product for commercial benefits.

Bonaccorsi [Bonaccorsi05] investigates reasons why companies participate in open source communities. In particular, the paper analyzes discrepancies between attitudes and behaviors in relation to three primary research questions. The questions deal with motivation to set up an open source business, whether the firms' claims to uphold intrinsic, community-based values are aligned with the firms' actions, and finally, if there is discrepancy, are there any observable patterns. The conclusion points out that there is misalignment in attitudes and behavior of firms in open source market place, confirming earlier research that companies use intrinsic values to attract developers in order to fulfill their own extrinsic goals.

[Westenholz06] offers an insight into challenges of creating a sustainable business around open source business model based on a case study. The study offers insight into shifting of business strategies conducted by the entrepreneur in order to make the business profitable around the combination of open source and proprietary software.

[Lindman09] asserts that business models can sometimes be too generic and undertake an exploratory case study of three different organizations in order to empirically identify different incentives companies have in releasing a product as open source beyond the revenue generating ones. The paper points to challenges in attracting and sustaining a community for a software product that is highly specialized.

4.3.4. Open source as part of component based software engineering

The article from Arhippainen [Arhippainen03] is a case study conducted at Nokia on the usage of the OTS components. The paper presents a detailed analysis on usage of third party components in general, and discusses advantages of using proprietary over open source components and vice versa. It also identifies issues related to software development methodology in terms of including third party components.

The research presented in [Ayala09] assesses the state of reusable components in the open source market based on survey conducted in Spanish and Norwegian companies. The results of the survey also assess the needs of OSS industrial users in component selection and identify challenges that can aid in maturing the open source components market.

The 5 articles [Li05a], [Li09], [Li05b], [Li06a], [Li06b] are based on the analysis of data collected through state-of-the practice survey conducted in Norway, Germany, and Italy. This research conducted on over

100 projects that use proprietary or open source OTS components looks into risks associated with use of such components, reasoning behind using OTS components, and impact on development process when using the OTS components. Some of the findings suggest that selection of OTS components is very informal process, that OTS components are selected throughout the development process life cycle even though early selection yields benefits. It is also suggested that estimates on effort needed to integrate components is informal and dependent on experience, and as such, is often inaccurate. Furthermore, some general conclusions of the studies point that the OTS components rarely have negative impact on the system. Open source OTS components are used in the same manner as proprietary components, thus without modification. If a problem occurs with the OTS components, it takes substantial amount of effort to correct them.

It can be noticed that in [Li05a]–[Li09] there is no in-depth analysis of what kind of open source components as OTS components, were used by companies. For example, many mainstream proprietary IT workshops, sometimes very ”hostile” to the idea of using open source components, like the ones producing software for big financial houses, use PGP and other Unix based open source components as these over the time have become de-facto standard. Investigating into the diversity and type of open source OTS components that are used in projects can be a question for further research.

4.3.5. Using the open source process within a company

An interesting area that is investigated in [Gurbani05] and [Lindman08] is that of using an open source process within a company. That is, the product is not provided to any community outside the organization, but it is handled as an open source project within the company. One unit of the organizations owns the product and provides this to the rest of the organization. They are allowed to use and modify the code, and changes are approved by the original owners as in any other open source project. [Gurbani05] presents a case study on transferring the open source development model for one software product at Lucent technology. In this case the approach was judged successful, by the authors for example because the product was needed in several products and the architecture was suitable for this. [Lindman08] also investigates the usage of an internal open source development methodology through a case study. This case study is conducted on usage of Nokia iSource portal for hosting projects. The portal became very popular for managing heterogeneous types of projects: SCM, distributed, agile, inter-company collaboration projects. The research results showed that implementation of open source project management tools can facilitate innovation within the company.

It should be noticed that both case studies are conducted at large companies, which probably is natural. A number of further research questions can be identified. One concerns how contributions can be included in this kind of product when different developers have different needs for the developed product.

5. DISCUSSION

In this review, 19 articles were identified. We do not think that this is a large number of articles compared to the importance of the field, and the general amount of discussion about how open source can be used by commercial organizations.

There is, of course, a risk that some articles have been missed in the search, either because the search string has not identified all relevant articles, or because the set of searched journals was not complete. The search string was developed through a ”trial and error” approach in order to find as many relevant articles as possible, but it is, of course, impossible to guarantee that all articles have been found. The same is true for the coverage of the search. It is not possible to guarantee that all relevant articles have been found. Here the largest risk is probably for articles not in the traditional software engineering literature, such as articles on business models, which is more general than traditional software engineering. Since there is a risk that all articles have not been found it is reasonable to discuss the effects of missing articles. Of course, the more complete the selection of articles is the better it is. However, in this case the objective is more to identify conducted research and experience than to carry out meta-analysis, which probably means that the effect of missing single articles is lower.

Compared to the research questions in Section 3.1, it can be seen that there are more research on question 1 than on question 2. The area of participating in open source communities is related to research question 2, but it would probably be possible to investigate this more specifically.

Many of the studies are in the form of surveys, which gives a broad and necessary understanding. Based on this it would probably be possible to conduct more studies investigating specific cases of implementation of methodologies for dealing with different aspects of open source in industry. More case studies could probably be conducted on all aspects of the research questions. More case studies could probably also provide more knowledge of research question 3 and research question 4. That is, research could be carried out to understand more about the cost and advantages of different approaches, and why different approaches are chosen. It is also worth noticing that there are no controlled experiments at all in the identified articles.

6. CONCLUSIONS

Research articles were found in the areas of using open source products and processes. Here the focuses of the articles are on i) participation in open source development communities, ii) business models with open source, and iii) treating open source software as components in component based development. Besides this there are articles on iv) how open source processes can be used within a company.

These are important areas for research and it is interesting to see that research is available in all these areas. The question of how to use open source practices within a closed company (iv) is for example an interesting area for further research. Based on this review we also propose that further research is conducted on how companies can transform their proprietary software to open source and build a community on it. Further research related to all four research questions in Section 3.1 could involve more case studies on implementation of specific methodologies for dealing with different aspects of open source in industry.

ACKNOWLEDGMENT

This work was partly funded by the Industrial Excellence Center EASE – Embedded Applications Software Engineering, (http://ease.cs.lth.se).

7. REFERENCES

Chen, L., Ali Babar, M., and Cawley, C. (2008) A status report on the evaluation of variablity management approaches, In *Proceedings of Evaluation and Assessment in Software Engineering (EASE)*.

Dieste, O., Griman A., and Juristo, N. (2007) Developing search strategies for detecting relevant experiments, *Empirical Software Engineering*, 14, pp. 513-539.

Dybå, T., and Dingsøir, T. (2008) Strength of evidence in systematic reviews in software engineering, In *Proceedings Empirical Software Engineering and Measurement (ESEM)*, Kaiserslautern, Germany, pp. 178-187.

Feller, J., and Fitzgerald B. (2002) *Understanding open source software development*, Addison Wesley.

Goode, S. (2005) Something for nothing: management rejection of open source software in Australia's top firms, *Information & Management*, 42, pp. 669-681.

Kitchenham, B., and Charter, S. (2007) *Guidelines for performing systematic literature reviews in software engineering, v. 2.3*, Technical report, Keele University and University of Durham.

Krishnamurthy, S. (2006) On the intrinsic and extrinsic motivation of free/libre/open source (FLOSS) developers, *Knowledge Technology and Policy*, 18, pp. 17-39.

Leavesley, G.H., Markstrom, S.L., Restrepo, P.J., and Viger, P.J. (2002) A modular approach to addressing model design, scale, and parameter estimation issues in distributed hydrological modelling, *Hydrological Processes*, 16, pp. 173-187.

Lerner, J., and Tirole, J. (2001) The open source movement: key research questions, *European Economic Review* 45, pp. 819-826.

Raymond, E. (2001) *The cathedral and the bazaar*, O'Reilly Media.

Runeson, P., and Höst, M. (2009) Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering, 14, pp. 131-164.

Stol, K-J., and Ali Babar, M. (2009) Reporting empirical research in open source software: the state of practice, in *Proceedings International Conference on Open Source Systems*, Skövde, Sweden, pp. 156-169.

Weber, S. (2004) *The success of open source*, Harvard University Press.

APPENDIX: ARTICLES INCLUDED IN REVIEW

[Arhippainen03]: Arhippainen, L. (2003) Use and integration of third-party components in software development. Technical report, VTT Publubilcation 489:84. In this report a case study on component based development, including the use of open source components, at Nokia is presented.

[Ayala09]: Ayala, C., Hauge, Ø., Conradi, R., Franch, X., Li, J., and Sandanger Velle, K. (2009) Challenges of the Open Source Component Marketplace in the Industry, In proc. OSS, pp. 213-224. The paper analyzes the state of open source market place and how companies interact to reuse components that the market place offers.

[Bonaccorsi05]: Bonaccorsi, A., and Rossi, C. (2005) Intrinsic Motivations and Profit-oriented Firms in Open Source Software. Do firms practise what they preach?. In proc. OSS, pp. 241-245. The articles investigates true motivation behind companies involvement in open source activities based on data gathered through a survey of 146 Italian companies supplying open source solutions.

[Bonaccorsi06]: Bonaccorsi, A., Giannangeli, S., and Rossi, C. (2003) Entry strategies under competing standards: Hybrid business models in the open source software industry. Management Science, 52(7):1085-1098. This is a further analysis of the same survey as presented in [Bonaccorsi05]. They have developed a regression model explaining the ”friendliness” to open source based on a set of factors.

[Bonaccorsi07]: Bonaccorsi, A., Lorenzi, D., Merito, M., and Rossi, C. (2007) Business firms' engagement in community projects, empirical evidence and further developments of the research. In First International Workshop on Emerging

Trends in FLOSS Research and Development, FLOSS'07. This is a survey based on a sample of 300 projects from SourceForge. In 97 of the projects there was at least one company participating. The three main types of involvement were ”project coordinator”, ”collaboration”, and ”provision of code”.

[Gurbani06]: Gurbani, V.K., Garvert, A., and Herbsleb, J.D. (2006) A case study of a corporate open source development model. International Conference on Software Engineering, pp. 472-481. This paper presents a case study on transferring the open source development model for one software product to a commercial environment at Lucent technology, keeping the software proprietary in the company.

[Hauge07]: Hauge, Ø., Sørensen, C.F., and Røsdal, A. (2007) Surveying Industrial Roles in Open Source Software Development. In proc. OSS, pp. 259-264. This paper defines different industrial roles in open source community: provider, integrator, participant, and inner source software participant. Through a survey, it investigates motivation, challenges, and development practices of the companies taking on these roles within the ITEA COSI project.

[Hauge09]: Hauge, Ø., and Ziemer, S. (2009) Providing Commercial Open Source Software: Lessons Learned. In proc. OSS, pp. 70-82. This paper presents a study on a small Norwegian software company that has built its business around own OSS products and compares the findings to other cases reported in literature.

[Li05a]: Li, J., Conradi, R., Slyngstad, O.P.N., Bunse, C., Khan, U., Torchiano, M., Torchiano, M., and Morisio M. (2005) An empirical study on off-the-shelf component usage in industrial projects. In Product Focused Software Development and Process Improvement (Profes), pp. 54-68. The paper presents survey conducted a large number of companies from Norway, Germany, and Italy on the off-the shelf (OTC) components usage. It focuses on factors that influence the choice in terms of whether the OTS component is open source or proprietary.

[Li05b]: Li, J., Conradi, R., Slyngstad, O.P.N., Bunse, C., Khan, U., Torchiano, M., and Morisio, M. (2005) Validation of new theses on off-the-shelf component based development, International Software Metrics Symposium, pp. 231-240. The paper focuses on validating six theses related to usage of off-the-shelf components within companies.

[Li06a]: Li, J., Conradi, R., Slyngstad, O.P.N., Bunse, C., Torchiano, M., and Morisio, M. (2006) An empirical study on decision making in off-the-shelf component-based development. In proc. International Conference on Software Engineering (ICSE), pp. 897-900. This article investigates research questions resembling those in [Li05a], but with a larger sample of projects.

[Li06b]: Li, J., Torchiano, M., Conradi, R., Slyngstad, O.P.N., and Bunse, C. (2006) A state-of-the-practice survey of off-the-shelf component-based development processes. In Reuse of off-the-shelf Components, International Conference on Software Reuse, pp. 16-28. This paper focuses on the development process when OTS components are used.

[Li09]: Li, J., Conradi, R., Bunse, C., Torchiano, M., Slyngstad, O.P.N., and Morisio, M. (2009) Development with off-the-shelf components: 10 facts. IEEE Software, 26(2):80-87. This article basically summarizes the findings from the earlier articles by the same group of authors. The conclusions are presented in the form of 10 facts learned about development with OTS components.

[Lindman08]: Lindman, J., Rossi, M., Marttiin, P. (2008) Applying Open Source Development Practices Inside a Company. In proc. OSS, pp. 131-387. This paper investigates characteristics of using open source and agile development practices within a company. It argues that usage of such practices can unlock innovation potential within a company.

[Lindman09]: J. Lindman, J.P. Juutilainen, M.Rossi. (2009) Beyond the Business Model: Incentives for Organizations to Publish Software Source Code. In proc. OSS, pp. 47-56. The paper investigates incentives for companies to release software as open source through three exploratory case studies at different stages of code release.

[Lundell06]: Lundell, B., Lings, B., and Lindqvist, E. (2006) Perceptions and Uptake of Open Source in Swedish Organizations. In proc. OSS, pp. 155-163. The paper investigates usage of open source within Swedish companies from the perspective that goes beyond mere adoption of an open source software product. It focuses on participation of the companies within the open source communities in roles of code contributors on existing third party projects or its own products.

[Robles07]: Robles, G., Dueñas, S., and Gonzalez-Barahona, J.M. (2007) Corporate Involvement of Libre Software: Study of Presence in Debian Code over Time. In proc. OSS, pp. 121-132. The paper investigates corporate involvement in Linux Debian GNU distribution over the period from 1998-2004 based on copyright attributions in the source code. The results of the research show that 6-7% of the code has been contributed by corporations.

[West03]: West, J. (2003) How open is open enough? melding proprietary and open source platform strategies. Research Policy, 32(7):1259-1285. The authors present a timeline for what has happened with respect to ”hybrid” software systems, that is software systems that consists of a mixture of open source software and proprietary software, such as an Apple computer. Three case studies are presented.

[Westenholz06]: Westenholz, A. (2006) Institutional Entrepreneurs and the Bricolage of Intellectual Property Discourses. In proc. OSS, pp. 183-193. This paper is a case study on an institutional entrepreneur who builds his own company on a business model that mixes open and closed source software practices.