Usage of Open Source in Commercial Software Product Development - Findings from a Focus Group Meeting

Höst, Martin; Orucevic-Alagic, Alma; Runeson, Per

Published in:
Product-Focused Software Process Improvement/Lecture Notes in Computer Science

DOI:
10.1007/978-3-642-21843-9_13

2011

in proceedings of International Conference on Product Focused Software Development and Process Improvement (PROFES), pp. 143-155, 2011. http://link.springer.com/chapter/10.1007%2F978-3-642-21843-9_13
The original publication is available at www.springerlink.com

Usage of Open Source in Commercial Software Product Development – Findings from a Focus Group Meeting

Martin Höst, Alma Oručević-Alagić, and Per Runeson

Department of Computer Science, Lund University
P.O. Box 118, SE-211 00 Lund, Sweden
{martin.host,alma.orucevic-alagic,per.runeson}@cs.lth.se
http://serg.cs.lth.se/

Abstract. Open source components can be used as one type of software component in development of commercial software. In development using this type of component, potential open source components must first be identified, then specific components must be selected, and after that selected components should maybe be adapted before they are included in the developed product. A company using open source components must also decide how they should participate in open source project from which they use software. These steps have been investigated in a focus group meeting with representatives from industry. Findings, in the form of recommendations to engineers in the field are summarized for all the mentioned phases. The findings have been compared to published literature, and no major differences or conflicting facts have been found.

Keywords: open source, industrial, off-the-shelf components

1 Introduction

Open source software denotes software that is available with source code free of charge, according to an open source license [1]. Depending on the license type, there are possibilities to include open source components in products in the same way as other components are included. That is, in a large software development projects, open source software can be used as one type of component as an alternative to components developed in-house or components obtained from external companies.

There are companies that have experience from using well known open source projects. Munga et al. [2], for example, investigate business models for companies involved in open source development in two case studies (Red Hat and IBM) and concludes that "the key to their success was investing resources into the open source development community, while using this foundation to build stable, reliable and integrated solutions that were attractive to enterprise customers". This type of development, using open source software, is of interest for several companies. If open source components are used in product development there are

a number steps that the company needs to go through, and there are a number of questions that need to be solved for each step.

First potential components must be identified, which can be done in several ways. That is the company must decide how to identify components. Then, when potential components have been identified, it must be decided which component to use. In this decision there are several factors to consider, and the company must decide how to make this decision. Using the components there may be reasons to change them, which gives rise to a number of questions on how this should be done and to what extent this can be recommended. A company working with open source components must also decide to what extent to get involved in the community of an open source project.

There is some research available in this area [3], although there is still a need to collect and summarize experience from companies working in this way. In this paper, findings are presented from a workshop, in the form of a focus group meeting, where these topics were analyzed by industry representatives.

The outline of this paper is as follows. In Section 2 the methodology of the research is presented, and in Section 3 the results are presented. The results are compared to results presented in the literature in Section 4, and in Section 5 the main conclusions are presented.

2 Methodology

2.1 Focus group

The workshop was run as a focus group meeting [4, 5]. At the workshop, participants informally presented their experience from development with open source software, for example from using open source components in their product development, or from participating in open source communities. The intention was to give all participants both an insight into how others in similar situations work with these issues, and to get feedback on one's own work from other organizations. The result of a similar type of workshop was presented in [6].

Invitations to the workshop were sent to the network of the researchers. This includes earlier participants at a seminar on "research on open source in industry" where rather many (≈ 50) people attended, and mailing lists to companies in the region. This means that the participants cannot be seen as a representative sample of a population and generalizations cannot be made in traditional statistical terms. Instead analysis must be made according to a qualitative method, e.g. as described by Fink [7, p. 61-78]. This is further discussed in Section 2.4.

2.2 Objectives and discussion questions

The main research questions for the study were:

- How should open source components for inclusion in products be selected? Is there a need to modify selected components, and if so, how should this be done?

– To what extent is code given back to the open source community, and what are the reasons behind doing so?

Discussion questions could be derived from the objectives in different ways. One possibility would be to let the participants focus on a specific project and discuss issues based on that. The advantage of this would be that it would probably be easy for the participants to know what actually happened since it concerns a specific project. The difficulties with this approach are that there is a risk that participants have valuable experience from more than one project and therefore cannot express all experiences they have since they should focus on one specific project. There is also a risk that data becomes more sensitive if it is about a specific project. Another alternative is to ask about more general experience from the participant and let them express this in the form of advice to someone working in the area. That is, the participants use all the experience and knowledge they have, without limiting it to a specific project or presenting details about projects, customers, etc. This was the approach that was taken in this research.

Based on the objectives of workshop, the following discussion questions were phrased:

1. How should one identify components that are useful, and how should one select which component to use?
2. How should one modify the selected component and include it in ones product?
3. How should one take care of updates from the community?
4. How should one handle own modifications/changes? What are the reasons for giving back code (or not giving back code)?

In order to get a good discussion, where as many relevant aspects as possible were covered, it was monitored in the following way. For each discussion question, the participants were given some time to individually formulate an answer, or several answers, on a Post-it note. When individual answers had been formulated each participant presented their answer to the others, and the notes were posted on the wall. During the discussions, the researchers also took notes.

2.3 Analysis procedure

The main data that was available for analysis were the notes formulated by the participants ("P-notes" below) and the notes taken by the researchers ("R-notes" below). The analysis was carried out in a number of steps, which are summarized in Figure 1 and explained below.

First all P-notes were transcribed into electronic form. In this step one note was transformed into one line of text. However, in some cases the participants wrote lists with more than one note at each piece of paper. In these cases this was clearly marked in the transcript. When interpreting the notes, the researcher were helped by the fact that the participants had presented the notes at the meeting earlier.

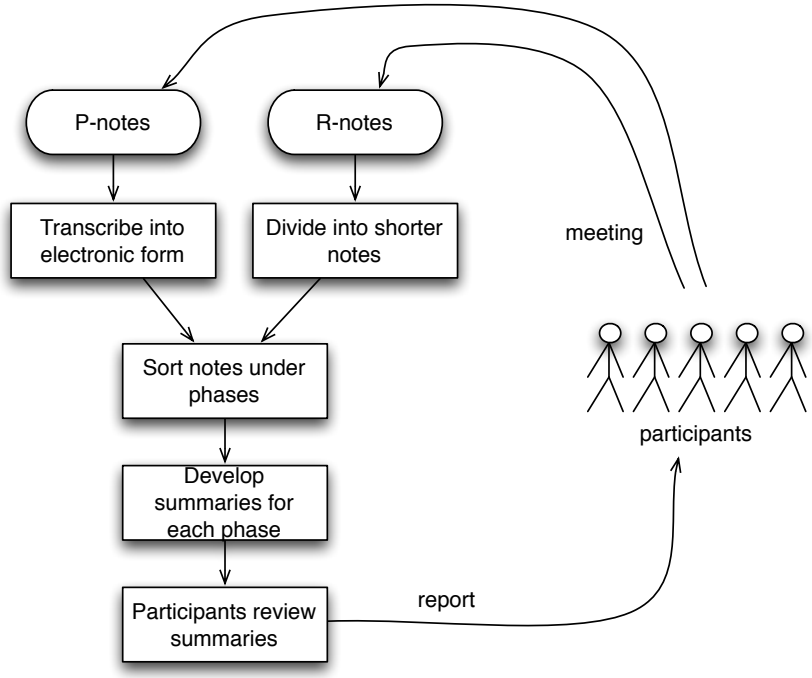

Fig. 1. Main analysis steps

The R-notes were derived by dividing a longer text into single notes. After this the P-notes and the R-notes were on the same form.

After this a set of phases were defined, based on the lifecycle phases in software development. These phases were based on the areas covered by the questions, but not exactly the same. Then, all notes could be sorted under the phases in which they are relevant.

Next, all notes were grouped in related themes within phases, and based on these summaries were developed. This means that one presentation summary was developed for each phase. The final version of these summaries are presented in Section 3.

Based on this, a report was developed with the summaries. The participants were given the possibility to review and adapt the summaries in the report. This resulted only in minor changes.

This procedure results in a summary, as presented in Section 3. The results were given back to the participants in the form of a technical report. This result is also compared to the literature in Section 4 of this article.

2.4 Validity

Since the collected data is analyzed qualitatively, the validity can be analyzed in the same way as in a typical case study, which in many cases also is analyzed qualitatively. Validity can for example be analyzed with respect to construct validity, internal validity, external validity, and reliability [4, 8].

Construct validity reflects to what extent the factors that are studied really represent what the researcher have in mind and what is investigated according to the research questions.

In this study we believe that the terms (like "open source", "component", etc.) that are used are commonly used terms and that the risk of not meaning the same thing is low. It was also the case that the participants formulated much of the notes themselves, which means that they used terms that they fully understood. Besides this, the researchers participated in the whole meeting, which means that it was possible for them to obtain clarifications when it was needed. Also, the report with the same material as in Chapter 3 of this paper was reviewed by the participants.

Internal validity is of concern when causal relations are examined. In this study no causal relations are investigated.

External validity is concerned with to what extent it is possible to generalize the findings, and to what extent the findings are of interest to other people outside the investigated case.

The study was conducted with a limited set of participants from a limited set of organizations. This means, of course, that the results cannot automatically be generalized to other organizations. Instead it must be up to the reader to judge if it is reasonable to believe that the results are relevant also for another organization or project. The results are compared and validated to other literature and the type of results is not intended to be specific for a certain type of results.

It should also be noticed that the findings from the focus group are based on the opinions of the participants. There may be a risk that the opinions are very specific for one participant or for the organization he/she represents. The nature of a focus group meeting helps avoiding this problem. According to Robson there is a natural quality control and participants tend to provide checks and react to extreme views that they do not agree with, and group dynamics help in focusing on the most important topics [4, Box 9.5].

Reliability is concerned with to what extent the data and the analysis are dependent on the specific researchers.

In order to obtain higher validity with respect to this, more than one researcher were involved in the design and the analysis of the study. Also, as mentioned above, the report with the same material as in Chapter 3 of this paper was reviewed by the participants.

Another aspect that is relevant to this is how the questions were asked and what type of data the participants are asked to provide. In order to avoid problems with confidentiality, the participants were asked to formulate

answers more as advice to someone who is working in the area than as concrete experiences from specific (and named) projects. We believe that this makes it easier to provide data for this type of participants.

3 Results from focus group meeting

3.1 Participants

At the workshop the following participants and organizations participated:

A. Four researchers in Software Engineering from Lund University, i.e. the authors of this paper and one more person
B. One researcher in Software Engineering from another university
C. Two persons from a company developing software and hardware for embedded systems.
D. One person from a company developing software and functionality based on an embedded system
E. One person from an organization developing software and hardware for embedded systems with more than 10 years tradition of using open source software
F. One person from an organization with the objective of supporting organizations in the region to improve in research, innovation and entrepreneurship in mobile communications

That is, in total 10 persons participated, including the authors of this paper.

3.2 Identification

Previously, companies were used to choose between making components themselves or to buying them. Now the choice is between making or buying, or using an open source component. That is, there is one more type of component to take into account in the identification process. It should also be pointed out that it is a strategic decision in terms of whether the product you are developing should be seen as a closed product with open source components or as an open source product.

When components are identified it is important that this is based on a need in the development and that it maps to the product requirements. When it comes to the criteria that are used when identifying components, they should preferably be identified in advance.

In the search process, the advice is to start with well-known components and investigate if they fulfill the requirements. There is also a lot of knowledge available among the members in the communities, so if there are engineers in the organization that are active in the community, they should be consulted. A further advice is to encourage engineers to participate in communities, in order to gain this kind of experience. However, the advice to consult engineers in the

organization is not depending on that they are members of the communities. A general knowledge and awareness of existing communities is also valuable.

The next step is to search in open source forums like sourceforge and with general search engines like google. The advice here is to use technical terms for searching (algorithm, protocols, standards), instead of trying to express what you try to solve. For example, it is harder to find information on "architectural framework" than on specific techniques for this.

3.3 Selection

The more general advises concerning the selection process is to, again, use predefined criteria and recommendations from colleagues. It is also possible to conduct a basic SWOT-analysis in the analysis phase.

A more general aspect that is important to take into account is if any of the identified components can be seen as an "ad hoc standard", meaning that they are used in many products of that kind and if it will increase interoperability and the ease communication with other components. One criterion that is important in this selection concerns the legal aspects. It is necessary to understand the constraints posed by already included components and, of course, other aspects of the licenses.

Other more technical criteria that are important include programming language, code quality, security, and maintainability and quality of documentation. It is necessary to understand how much effort is required to include the component in the architecture and it is necessary to understand how the currently used tool chain fits with the component. A set of test cases is one example of an artifact that is positive if it is available in the project.

A very important factor concerns the maturity of the community. It is necessary to investigate if the community is stable and if here is a "backing organization" taking a long-term responsibility. It is also important to understand what type of participants in the community that are active. The roadmap of the open source project is important to understand in order to take a decision that is favorable for the future of the project.

3.4 Modification

First it should be emphasized that there are disadvantages of making changes to an own version of the components. The disadvantages are that the maintenance costs increase when updates to new versions of the components are made, and it is not possible to count on extensive support for specific updates from the community. So, a common recommendation is to do this only if it is really necessary.

There are some reasons why modifications must be made. Especially adaptation to specific hardware is needed, but also optimizations of different kind. When these changes are made it is in many cases favorable to give back to the community as discussed in the next section but if this is not possible an alternative is to develop "glue software" and in that way keeping the API unchanged.

If changes should be made it is necessary to invest effort in getting a deep knowledge of the source code and architecture, even if a complete set of documentation is not available.

3.5 Giving back code

It is, as discussed in the previous section, in many cases an advantage to commit changes to the open source project instead of working with an own forked version. In this way it is easier to include updates of the open source component. In order to manage this it is in many cases an advantage to become an active member of the community, and maybe also take a leading role in it. When modifying an open source component it is, of course, an advantage if ones own changes can be aligned with the future development of the open source component.

However, there are some reasons not to give back changes too. The most important reason is probably that you want to protect essential IPR's and core competences in the organization. That is, key competence must in some situations be hidden from competitors. It should, however, be noticed that there may be requirements from the license to give back code. Also, after some time, all software will be seen as commodity, which means that this kind of decision must be reconsidered after a while. Another reason not to make changes public is that possible security holes can be made public. In some cases it is easier to get a change accepted if test cases are supplied.

3.6 Summary of results

The main findings from the workshop, in terms of recommendations for the four phases, are summarized in Figure 2.

4 Comparison to literature

The area of open source in product development has been investigated in the literature, and it is therefore possible to compare the results in the workshop to the results reported in literature.

The two first authors conducted a systematic review of open source in commercial organizations [3]. In that work it was found that presented research in the area could be divided into four main areas:

1. Company participation in open source development communities
2. Business models with open source in commercial organizations
3. Open source as part of component based software engineering
4. Using the open source process within a company

Of these different topics, the first one about company participation in open source projects, and the third one about open source as one type of components are the most relevant for this study.

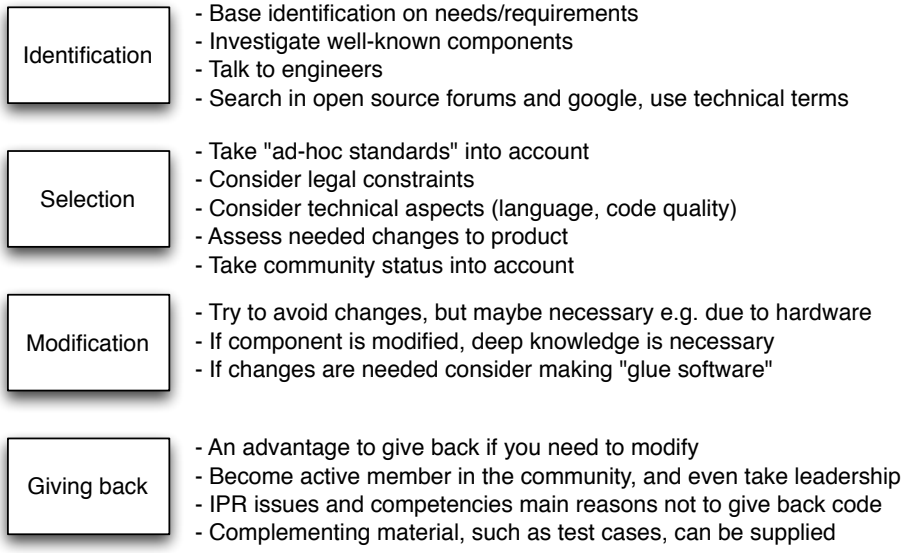

Fig. 2. Main findings from workshop

4.1 Company participation in open source development communities

There is company participation in many open source projects. For example, Bonaccorsi et al. [9] report that in a third of the most active projects on SourceForge there is or was some kind of company participation. Companies can participate as project coordinator, collaborator in code development, and by providing code etc. This can be compared to the related result of this study, where it was argued by the participants that it is important to become an active member of the community and even take leadership.

Hauge et al. [10] identify one additional role, which is more concerned with integration of open source components. The need for this role can also be confirmed in this study, since the participants acknowledge the need to use the components without changing them too much. That is, it can be seen as it is better to integrate components than to change them.

For example Hauge et al. [10] emphasize that in case software should be provided to a community it is important to provide enough documentation and information to get the community members going. This was not really discussed at the meeting, but the recommendation to get involved in the community and maybe even to take a lead in the open source project do also point at the importance of taking part in developing the community of a project.

From the data presented by Lundell et al. [11] and Robles et al. [12] it is clear that 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. This can, of course, not be confirmed in a statistical way by a small sample like in this study, but it can be noted that the involvement in open source projects was seen as important by the participants. The amount of participation can be summarized as follows: It is suggested that a significant number of the companies marginally participate in open source community, although 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 [11]. This can be compared to the investigation presented by Robles et al. [12] that show that 6-7% of the code in Linux Debian GNU distribution over the period 1998-2004 has been contributed by corporations.

One risk that is identified by companies is that people working in the organization would reveal too much information to the outside of the organization if they work with an open source community. However, the revealing behavior of this kind of software engineers was investigated by Henkel [13] and it was found that even if the engineers identified with the community they were significantly less identified and ideological about open source than the control group of non-commercial developers. The conclusion from that research is that there are indicators of commercially harmful behavior in this kind of development. In the focus group it was found that intellectual property is important, which points in the same direction as the results presented by Henkel [13].

4.2 Open source as part of component based software engineering

The report from Arhippainen [14] presents a case study conducted at Nokia on the usage of the OTS components. The report presents an analysis on usage of third party components in general, and discusses advantages of using proprietary over open source components and vice versa. The results of the presented case study focus to some extent on development of "glue software", which is also in line with the results of the focus group meeting, that is, if changes are needed do this through glue software.

Li et al. [15] present the results of a set of surveys in the form of 10 facts about development with off-the-shelf (OTS) components, of which open source components is one type. Various aspects, and more details, have also been presented in other articles by the authors. The results of this study are compared to the 10 facts below. Note that the study by Li et al. has a broader scope than this study, e.g. since it investigates all sorts of OTS and not only open source, which is one reason why all identified facts have not been investigated in this study.

1. "*Companies use traditional processes enriched with OTS-specific activities to integrate OTS components*": In this study we did not investigate the development in detail, although there was nothing in this study that argued against the fact.

2. "*Integrators select OTS components informally. They rarely use formal selection procedures*": The aspects that were discussed at the focus group meeting were rather "high level", like "take 'ad-hoc standards' into account" and taking legal aspects into account. Research-based formal processes for selection were not mention. Even if this does not mean that this kind of methods is not used, in many cases more ad-hoc methods are probably used. It is probably important to be able to take many different factors of different nature into account when selecting components.
3. "*There's no specific development process phase in which integrators select OTS components. Selecting components in early phases has both benefits and challenges*": In the study presented in this paper we did not investigate this aspect in detail. However there was nothing in the study that argued against this fact. It could also be noted that the participants could use separate phases of working with open source components, like identification, selection, modification, and giving back code, in the discussion. This means that it is possible to divide the work with open source components in different phases.
4. "*Estimators use personal experience when estimating the effort required to integrate components, and they're usually inaccurate. Stakeholder-related factors will dramatically affect estimates' accuracy.*": In the study presented in this paper we did not investigate this aspect in detail. However there was nothing in the study that argued against this fact. However, it was seen as advantage to have good knowledge of the open-source project, which of course affects the possibility to estimate effort for adaption of components.
5. "*OTS components only rarely have a negative effect on the overall system?s quality.*": Since this concerns more the result of using open source components than how to work with them, this was not discussed during the meeting.
6. "*Integrators usually use OSS components in the same way as commercial components – that is, without modification*": OSS was not compared to COTS at the meeting. However, the participants recommended not to make changes if it is not absolutely necessary. That is, the result of the meeting supports this fact.
7. "*Although problems with OTS components are rare, the cost of locating and debugging defects in OTS-based systems is substantial*": The participants pointed out the importance of involving themselves in the open source projects in order to be informed of all aspects of the project. It was not discussed, but the knowledge that is gained through this can be useful in this type of debugging.
8. "*The relationship with the OTS component provider involves much more than defect fixing during the maintenance phase*": The participants pointed out the importance of involving themselves in the open source projects, and maybe even taking the leadership of open source projects. That is, the result of the meeting clearly supports this fact.
9. "*Involving clients in OTS component decisions is rare and sometimes infeasible*": This was not discussed at the meeting.

10. "*Knowledge that goes beyond OTS components' functional features must be managed*": The participants pointed out the importance of involving themselves in the open source projects in order to be informed of all aspects of the project, not only technical aspects. That is, the result of the meeting supports this fact.

In general it can be concluded that the facts presented by Li et al. [15] are in line with this study. No data in this study pointed against the facts, and some facts, like facts 2, 8, and 9, were directly supported by this study.

5 Conclusions

We believe that many of the recommendations from the participants are important to take into account in research and in process improvement in other companies. The most important findings from the workshop are summarized below. The findings are in line with presented research in literature as described in Section 4, although the details and formulations are specific to the results of this study.

In the identification phase it is important to take the needs and the requirements into account, and to investigate well-known components. It is also advised to discuss the needs with engineers in the organization, since they can have knowledge of different components and communities. When forums are searched, an advice is to use technical terms in the search string. When selecting which components to use it is important to, besides taking technical aspects, like programming language, into account, also consider legal constraints and "ad-hoc standards". It is important to investigate the status of the community of a project, and the future of the project, which for example depends on the community. In general it can be said that changing components should be avoided if possible. If it is possible to make adaptations with "glue-code" this is in many cases better since less effort will be required in the future when components are updated by the community. However, there are situations when it is necessary to make changes in the components.

Even if there may be issues with property rights, it is in many cases an advantage to provide code to the community if changes have been made. In general it can be said that it is advised to become an active member in open source projects.

The findings from the focus group meeting were compared to published literature, and no conflicting facts were found.

Together with further research on the subject it will be possible to formulate guidelines for software project managers on how to work with open source software.

Acknowledgments

The authors would like to thank the participants for participating in the study.

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

References

1. Feller, J., Fitzgerald, B.: Understanding Open Source Software Development. Addison Wesley (2002)
2. Munga, N., Fogwill, T., Williams, Q.: The adoption of open source software in business models: A Red Hat and IBM case study. In: Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists,. (2009) 112 – 121
3. Höst, M., Oručević-Alagić, A.: A systematic review of research on open source software in commercial software product development. In: Proceedings of Evaluation and Assessment in Software Engineering (EASE). (2010)
4. Robson, C.: Real World Reserach. 2:nd edn. Blackwell Publishing (2002)
5. Kontio, J., Bragge, J., Lehtola, L.: The focus group method as an empirical tool in software engineering. In Shull, F., Singer, J., Sjøberg, D.I.K., eds.: Guide to Advanced Empirical Software Engineering. Springer (2008)
6. Engström, E., Runeson, P.: A qualitative survey of regression testing practices. In: Proceedings of International Conference on Product-Focused Software Process Improvement (PROFES). (2010) 3–16
7. Fink, A.: The Survey Handbook. 2:nd edn. Sage Publications (2002)
8. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering **14** (2009) 131–164
9. Bonaccorsi, A., Lorenzi, D., Merito, M., Rossi, C.: 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: ICSE Workshops 2007). (2007)
10. Hauge, Ø., Sørensen, C.F., Røsdal, A.: Surveying industrial roles in open source software development. In: International Conference on Open Source Systems (OSS). (259-264) 2007
11. Lundell, B., Lings, B., Lindqvist, E.: Perceptions and uptake of open source in Swedish organizations. In: International Conference on Open Source Systems (OSS). (155-163) 2006
12. Robles, G., Dueñas, S., Gonzalez-Barahona, J.: Corporate involvement of libre software: Study of presence in debian code over time. In: International Conference on Open Source Systems (OSS). (121-132) 2007
13. Henkel, J.: Champions of revealing–the role of open source developers in commercial firms. Industrial & Corporate Change **18** (2009) 435–472
14. Arhippainen, L.: Use and integration of third-party components in software development. Technical Report 489:84, VTT (2003)
15. Li, J., Conradi, R., Slyngstad, O.P.N., Bunse, C., Torchiano, M., Morisio, M.: Development with off-the-shelf components: 10 facts. IEEE Software (2009)