UNIVERSITY OF GOTHENBURG

Resource Allocation in Open Source Projects: A Profile Based Approach

Bachelor of Science Thesis Software Engineering and Management

Chayi Fan
Stefan Ulvdal

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, December 2014

UNIVERSITY OF GOTHENBURG

Resource Allocation in Open Source Projects: A Profile Based Approach

Chayi Fan
Stefan Ulvdal

Examiner: Morgan Ericsson

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden December 2014

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, December 2014

Resource Allocation in Open Source Projects: A Profile Based Approach

Chayi Fan
Software Engineering & Management
University of Gothenburg
Gothenburg, Sweden
Chayi.j.fan@gmail.com

Stefan Ulvdal
Software Engineering & Management
University of Gothenburg
Gothenburg, Sweden
stefan.ulvdal@gmail.com

Abstract

Free/Libre/Open Source Software (FLOSS) projects have steadily been rising in popularity and adoption. This is because more organizations and companies are starting to rely on products created through open source software development. The structure of contributors involved in open source software projects causes challenges to resource allocation which are not present in closed source commercial software projects. In general terms resource allocation refers to the allocation of resources such as personnel, time, and budget to help contributors.

This study aims to explore the relationship between resource allocation strategies in FLOSS projects and contributor profiles. This is done in order to provide a solution for the challenges of resource allocation, which in this context means the allocation of resources such as personnel, time, and budget. The challenges of resource allocation in a FLOSS project lies in the aspect that the contributors are often only loosely connected. Resource allocation can overcome these challenges by efficiently allocating resources to the contributors that need it most. The efficient allocation of resources can be achieved by distributing resources to the contributors based on their profile. The profiles represent the activity patterns of a contributor and can be used to schedule interactions between managers and contributors.

We performed this research as a case study and extracted data from the Git repository of the Linux kernel project. From the extracted data we then constructed profiles of each contributor in the project. The final list of profiles were then grouped by similar profiles. These groups can then be targeted in a resource allocation process depending on the needs of a company or organization.

1. Introduction

The popularity of FLOSS development has steadily been increasing and is impacting everyone that is involved in the software industry [1], it brings with it many possible advantages such as shorter adoption time, reduced cost, and increased innovation [2, 3]. However, FLOSS projects also faces the same problems that traditional software projects face with high failure rates, delays, and quality issues. Previous case studies have found that only about 16% of software projects are on time and within budget [4]. In addition, the U.S. Department of Defense (DOD) spent nearly 8 billion dollars in 2004 to rework software because of quality-related issues [5]. Budget, time and quality are the factors which decides the success of a software project. Resource allocation, which in this context means the allocation of resources such as personnel, time, and budget can have an important impact on the factors that influence the success of a project. All three factors can be especially be influenced by efficiently allocating resources with personnel that have the right capabilities [6].

In a FLOSS project, efficient resource allocation is a big challenge because of the complex situation regarding contributors. Contributors are people that contribute source code or other resources to the project. The complexity comes from the relationship between the project and its contributors. In a FLOSS the contributors are often a mix of volunteers and employed developers. The contributors may also be located all over the world and their involvement will often be sporadic if they are working on other projects as well. Contributors that are volunteering and who does not have a clear affiliation with a company or organization are called unknown contributors. These unknown contributors and paid contributors often have different incentives to be working on the project. The volunteers are working on the project without

compensation and on their free time. While the paid contributors are often working during office hours. These differences make it possible to characterize the activity patterns of contributors. The activity patterns then represent the profiles of contributors and show when and where a contributor is active. The data accumulated in the source code repository of a FLOSS project often provides the enough information to create these profiles. The profiles can then aid in finding the best time for allocating resources to the contributors in the form of support. This support can mean different things depending on the FLOSS project and the community. The communities of FLOSS projects may be very diverse. In the sense that there are many contributors with a wide array of skills and capabilities. Therefore, if the contributors can be properly identified through profiles it will help to smooth the progress of a project [7]. This study aims to explore the relationship between resource allocation strategies in FLOSS projects and contributor profiles. In this study we will answer the following two research questions:

Q1: To what extent are resource allocation strategies aligned with contributor activities?

Q2: How to make use of contributor profiles for resource allocation tasks?

This study will be conducted as a case study using real world data generated by people in various ways. The data is in the form of commits made by contributors to a FLOSS project. The commits are submitted changes to the projects source code. We will use this data to establish a link between the commits and the activity of contributors [8]. Our findings shows that through profiling and mapping the activity of contributors it is possible to target specific groups of contributors based on their profiles and traits. This enables organizations or companies with interest in a project to efficiently allocate their resources to the people which get the most benefits out of it.

This paper is divided to various sections such as Background & Related Work, Research Methodology, Data Analysis & Results, Discussion, and Conclusion. The next section introduces the reader to the most important concepts in this study. In the Research Methodology section we explain how this case study was done and how we proceeded to extract the data we needed. At the end the reader will find our results, discussion, and a conclusion.

2. Background & Related Work

In the following section we explore the background behind this study. We also introduce the reader to the concepts of resource allocation and profiling and what it means in the context of this study.

2.1. Resource Allocation

The definition of resource allocation can be summarized as the process that the management of a project uses to decide where the finite resources should be allocated. There are plenty of studies touching the subject of resource allocation which introduces multiple models for efficiently implementing resource allocation in a controlled environment. This is shown by Rahman and Ruhe [7] who introduce the importance of knowing the productivity and traits of the people working on the project and how this can be taken advantage of in the resource allocation process. However, few of the studies concern the adaptability of the methods to open source projects. Also, in research done by Duggan et al [9] they developed a multi-objective optimization model for software task allocation based on genetic algorithms. All these approaches depends on how familiar the organization is with the contributors of the project. A certain degree of information is needed in order to successfully apply a resource allocation process in a project. Thus it becomes a challenge to do this in a FLOSS project. Therefore, the resource allocation process in a FLOSS project can be recognized as the management of interactions between the organization and the contributors. The management process is divided to two parts. First, identifying the contributors who are valuable enough to be allocated with resources. Second, plan the interactions between organizational helpers and selected contributors. The interactions in a FLOSS project can be done in different ways depending on the nature of the project. For example, if a project is mainly backed by a company which develops a FLOSS product as part of its business model it becomes important how that company interacts with the community that arises around the project. One example of such a company is a Vaadin from Finland who develops a web framework with the same name. They are spending a lot of time and effort on tracking information and measuring how the community evolves over time since it is a vital part of their business model [10].

As we mentioned in the introduction. Identifying the activity of contributors can be used to assess the value of a contributor and determine if they are worth to be allocated with resources. Findings from other studies shows that the competency levels were associated with expected activity per day and that the expected numbers of defects were related to level of activity. Thus the development of the procedures for allocating personnel to software tasks can be based on the assessment of behavioral aspects [11]. Findings

also show that the required skill levels of some tasks can be estimated as average numbers of software lines of code (SLOC) per day which depends on the productivity of a contributor [12]. This inspires us that the identification of productivity and activity in this study can be based on data in the form of commits made by contributors. On the other hand, one of the concerns regarding contributors in FLOSS project is strongly related to the development process. Aspects such as how branches are merged, how contributed code changes are signed off by responsible branch managers affect both the quantity and quality of contributions. The interactions between managers and contributors require a lot of effort from a large number of people like project branch commanders, branch maintenance staffs, sign off staffs and a number of contributors. These interactions are mainly recognized as resource allocation activities in FLOSS project. Good management of interactions can improve not only the productivity of contributors but also the productivity and quality of the entire project.

2.2. Profiling

The process of profiling and analyzing a community can be done as a series of steps [13]. In this study we have followed these steps by extracting data from the source code repository of a large FLOSS project. The product of this process are the final profiles. These profiles will represent aspects of the people in the community in question relevant to the goal of the analysis. There are four major steps in the process.

1. Identify the goal of the analysis
2. Data collection
3. Data processing
4. Profile interpretation

These steps were used in a number of related research projects. One such endeavor was aimed at profiling the contributors of different Q&A sites. They measured the behavior, motivation, and expertise through profiles of the contributors in five different Q&A sites [14]. They used clustering analysis to identify groups of contributors with similar traits or behavior. Through this profiling process they came out with a number of different profiles that represent grouped behaviors of different contributors. The resulting profiles are highly dependent on the context of the of the community and the reason for doing the analysis. They could then also use these profiles to see if contributors change over time and transfer to a different profile. Their findings show that there are contributors that change profile from time to time the distribution of contributors among the profiles changes very little. Capiluppi and Izquierdo-Cortazar [15] carried out an investigation of the data in the Linux kernel by associating contributor activity with time slots. Their research show that there is a lot of knowledge that can be extracted from the repository of the Linux Kernel. In this study we are using the same approach for extracting the data in order to categorize the developers and generate the profiles.

In a study done by Aaltonen and Jokinen they investigated community profiles in the context of influence in the linux kernel by mining data from the Linux Git repository [16]. Their aim was to explore how the influence in development communities are distributed. The idea behind this is that the hierarchies in different organizations look different with different levels of involvement among the different actors. They found that the influence in the Linux Kernel project is centered around a small number of core companies and developers which fits the description of core developers in the onion model. While the rest and the large group of contributors which are in many cases only involved temporarily and in a specific part of the project. The onion model [17] describes the contributors as in layers with the innermost layer containing a small group of core contributors and with outer layers that gets bigger and bigger but the involvement of each contributor in these outer layer get smaller and smaller.

Koch [18] did a research project where he profiled a large amount of data regarding contributors and projects that was extracted from sourceforge. They explored the different projects on sourceforge and their size, both in lines of code and number of contributors. They found that the relation between the amount of projects and the number of contributors is skewed so that the majority of the projects have only one contributor and very few projects have even more than 10 contributors.

3. Research Methodology

We are conducting this research project as a case study [19] of the Linux Kernel Project. We chose the Linux Kernel Project because it is one of the largest FLOSS projects. It is a strong example of the two most important aspects that cause the challenges of resource allocation in FLOSS projects. Firstly, the Linux kernel project has become an enormous project developed on a massive scale by companies and individuals that are fierce competitors in other areas. There are about 7944 contributors and 855 companies [20] continuously following the development of the project, and there are about 23% of unknown contributors with whom the linux foundation are unable to determine a corporate affiliation. A closely related group of contributors are those which are known to be doing this work on their

own, with no financial contribution happening from any company. Together they make up over 60% of the total contributions to the kernel which means they take a critical and irreplaceable position in the development of the Linux kernel project. Secondly, up until version 2.6.0 which was released in 2003 the Linux kernel releases were divided into two parts, stable and development versions. After the 2.6.0 this changed and releases are made available in intervals of three months and new features are included if they are ready [21]. If a feature is not ready on time it will be delayed. It is very important that every new feature that is included in a release is of good quality in order to promote maintainability in the Linux kernel [22].

We chose to do a case study since we are working with data directly generated by real people and it is important that the results can be replicated [23]. The data we used from the Linux Kernel Project came entirely from their software repository located at kernel.org. It's in the form of a Git repository. In order to come to our results, we have to be able to access the commits of the Linux Kernel Project repository and extract the information from commits and filter the information in order to answer our research questions. Due to Linux Kernel Project is open-source and the development data is publicly available, we were able to download the entire repository by using the "*git clone*" command. This gave us a local copy of their software repository to work with on our own hard drive. The data we used contained commits between 2005-04-16 and 2014-04-18. This is because they only switched to using git in 2005 and most of the development history from before that is not included. Even so, there is 442127 commits and 14194 contributors in the data set.

3.1. Data Collection

After we made a local clone of the Git repository on our own computer we used CVSAnalY [24, 25, 15] to extract the data from the Git repository, When CVSAnalY extracts the data it saves it into an SQLite database. This database then contains all the information about the development history arranged into different tables. We used this database to further process the data into what we specifically needed.

3.2. Data Processing

When the data from the git repository had been extracted and placed into the SQLite database we looked at the commits to see what parts of the data could be used in our analysis. Each commit in this database was represented by data fields which we made use of. The data fields and their possible values can be seen in figure 1.

Email address	Time zone	Date & time
xxxx@xxx.xx	(-)HH	*YYYY-MM-DD HH:MM*

Figure 1 - Data fields

The email address for each contributor will be used to determine if the contributor is representing a company or if he is a volunteer. We will also use the email to determine if the contributor is working for one of the top 10 companies that are contributing to the Linux kernel. The categories we used were the following [20]:

- Top 10 Companies: Contributors that commit with an e-mail address that can be associated with a top 10 contribution company. These contributors are hired by the company aiming at development of linux kernel project.

- Other Companies: Contributors that commits with company email address where the companies are excluded of top 10 companies. The motivation for separately setting up this category is that some companies are much smaller with less resources compared to a top 10 company.

- Unknown Contributor: Contributors that commits using a public email address, those contributors are recognized as unknown contributors.

The email will be used to classify it in a specific category, unknown, company, or top 10 contributor. We can also see the timestamp and the time zone. The timestamp can be used to put the commit in relation to other commits in time and see when it was made. We will use the the timestamp to extract which weekday the commit was made and during which hour of the day, so that we then can see where the contributor have made his commits. The time zone information of the commit will be used to identify in which part of the world the contributor is active.

Each contributor will have a variable amount of tuples like the one displayed in figure 2 associated to him. The email address identifies each contributor so that it is possible to aggregate results based on each data field from several tuples. These different tuples represent individual commits. Then, when we want to create a profile for each contributor we use the values

Email	Time zone	Date	Comment
torvalds@g5.osdl.org	+8	12/22/2005 9:33:04	When compiled-in...

Figure 2 - Example commit

in the different data fields for each commit to determine the profile of the contributor. The different values of each data field for the commit needs to be combined into a single value that represents the range of values of the different commits. To best generalize the values into one value we take the mean of the set of values for time zone, day, and hour for each contributor. The hours are placed in one of three timeslots, office hours (09:00-17:00), after office (17:00-01:00), and late night (01:00-09:00). For the time zone value we will also determine which region it belongs to. Therefore we have divided the time zones of the world into three groups spanning the different continents. The possible values of the data fields that are used to define the profiles for each contributor creates a large number of possible profiles. The contributors have been placed into one of three possible categories depending on their email address, they are also associated with one of three possible geographical regions, one out of seven weekdays and one out of three possible time slots. The combination of these data fields creates the possibility of 189 different possible profiles.

```
Contributor profile =
       f(category(email)
        ,mean(time_zone)
        ,mean(day)
        ,mean(hour))
```

We filtered out contributors from the top 10 companies by selecting all contributors that used an emails that could be linked to one of these top companies. The selected contributors were then put into top 10 companies. The email addresses that represents the current top 10 companies and how many contributors that are contributing using that email address are displayed in figure 3 in the top 10 companies column. When these contributors had been put into top 10 companies we selected the contributors that contributed with their personal emails and put them into unknown contributors. The contributors that is not placed in unknown contributors are considered other companies. The emails we used to match personal emails are displayed in figure 3 in the unknown contributors column.

Unknown contributors	
@gmail.com	1626
@googlemail.cc	121
@web.de	47
@hotmail.com	29
@free.fr	46
@163.com	6
@yahoo.com	72
@zoho.com	1
@gmx.(com|de|	168
@inbox.com	1
Total	2117

Top 10 companies	
@redhat.com	302
@novell.com	36
@intel.com	598
@ibm.com	468
@oracle.com	76
@nokia.com	100
@fujitsu.com	90
@ti.com	247
@broadcom.cor	67
@google.com	147
Total	2131

Figure 3 - The lists of emails in the different categories and how many contributors are using them.

3.3. Data analysis

Because of the large amount of data that is being analyzed it is not feasible to look at individual contributors and their activities but we must look for generic traits that are shared by larger groups of contributors. The combination of values in the different data fields represents the profile for each contributor. We then aggregated all the profiles with similar combinations in the different data fields. This makes it possible to the rank all the profiles based on the most common combinations. We can now pick out specific profiles based on what we are looking for. The ranking of the profiles based on the number of contributors in each profile was used to answer the second research question. In order to answer the first research question we also aggregated the number of contributors who had profiles that specified the day when they are most likely to commit as Friday.

4. Results

In this section we describe the results and how the data was analyzed. We will answer the research questions that were presented in the introduction.

4.1. Research Question 1

Q1: To what extent are resource allocation strategies aligned with contributor activities?

One of the popular methods for resource allocation in FLOSS projects is to set up meetings and events with the community regularly. A typical example is “Community Friday” used by a Finnish company called Vaadin [10]. On each Friday afternoon, this company holds the meeting with the community surrounding their product. If the contributors are online they will be able to participate in the event and be able to discuss the progress, represent the difficulties of their works and ask for resource to help with the difficulties if possible. However, this method requires a high level of activity and involvement from the contributors during these “Community Friday” events. It is not certain that Fridays are always the best choice for interactions with the community but it depends on the collective characteristics of the community. Therefore, for other FLOSS project communities it may be appropriate to use a different day for their community events. In this study we have used data from the Linux kernel. From this data we have been able to determine how many contributors who have most of their activity on Fridays. The results show that around 1216 out of 14194 contributors have

Category	Region	Day	Time slot	Contributors
Other company	South & North America	Wed	Office hours	154
Other company	South & North America	Fri	Office hours	143
Other company	South & North America	Wed	After office	139
Unknown contributor	Europe & Africa	Fri	Office Hours	128
Other company	South & North America	Thu	Office Hours	127

Figure 4 - Top 5 contributor profiles.

Category	Region	Day	Time slot	Contributors
Top 10 company	Europe & Africa	Tue	Office hours	1
Top 10 company	South & North America	Wed	Office hours	1
Other company	South & North America	Wed	Office hours	1
Top 10 company	South & North America	Mon	Office hours	1
Top 10 company	Europe & Africa	Mon	Office hours	1

Figure 5 - Bottom 5 contributor profiles.

Friday as their most active day. That is around 8.5% which means that there are days which have more activity on them since there are 7 days in the week. Therefore, a different weekday should be selected which have higher overall activity. This shows that the effectiveness of a resource allocation strategy that involves interactions between managers and contributors is influenced by the activity of the contributors. Therefore, if the expected activity of the contributors can be charted it can be used to decide the appropriate time to schedule interactions.

4.2. Research Question 2

Q2: How to make use of contributor profiles for resource allocation tasks?

One of the biggest usages of profiles for resource allocation is that it identifies the different activity patterns of contributors. By knowing the activity habits of the contributors the interactions between the contributors and project management can be scheduled in an efficient way. The profiles in figure 4 and 5 are relevant because the managers have a limited amount of time to dedicate to interactions with the community. In this time they might need to answer questions or give feedback on contributions that are waiting to be integrated but needs further modifications or other changes. If the delays in the communication between the parties are large it will impact the speed with which contributions can be integrated or questions answered. For example, if the managers can get a hint about when it is most useful to be active on mailing lists or other communication mediums it can contribute to getting more contributions ready to be integrated on time. This requires knowledge about when the important contributors that is likely to need interaction are most likely to be active as in day, timeslot, and part of the world. For example, they might want to target contributors working for the top 10 companies in North or South America. Then they can inspect the list of profiles as seen in figure 4 and 5 and see which day and timeslot is best suited for interaction based on how many contributors have that profile. In figure 6 we can also see what the activity within a profile looks like in each year between 2006 and 2013. In this case the profile used is the top profile in figure 4. The years of 2005 and 2014 are excluded in that chart since the data only covers a few of months in those years. This information can be used to determine if the activity of a profile seems to be decreasing or increasing. If the activity is decreasing then a different profile should probably be targeted.

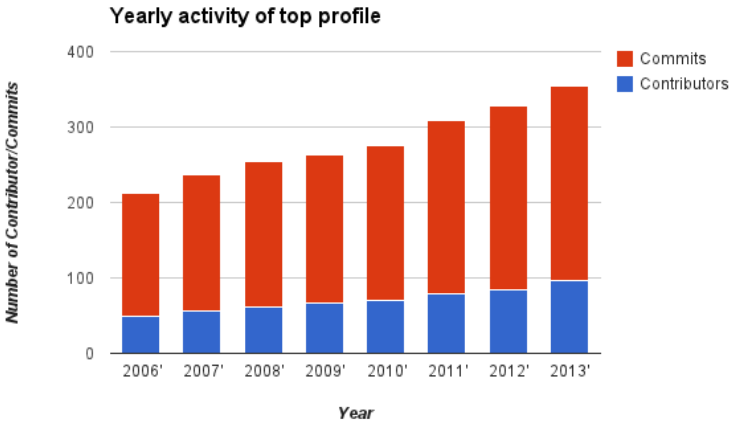

Figure 6: Displays the yearly number of active contributors and commits made in the top profile from figure 4. Year 2005 and 2014 are left out because we only have data covering a few months in those years.

5. Discussion

The best strategies for doing resource allocation in a FLOSS project will be different for almost every project. The rate with which the project gains contributors and where these contributors come from will be different depending on use case for the software and the technology used. It is hard for FLOSS project owners to decide on what they should put more attention on, company contributors or unknown contributors. This is influenced by two factors:

- Commercial value: Projects like linux kernel get a lot of attention and resources from companies because the linux kernel takes valuable position to the other companies products. Some of companies associate their products with linux kernel in the beginning of this project. Others become involved later after they see how the value and importance of the linux kernel relates to their product or business goals.
- Technology assets: Many volunteers get involved in FLOSS projects just because of their own interest in the technology.

The point of these two factors is that they show that both company contributors and unknown contributors take a significant position in a success of a FLOSS project. The FLOSS project owners should therefore evaluate the two factors to determine the development strategy for their project where this strategy is influencing the following resource allocation directions. For example, the company Vaadin which themselves are a commercial company and have a financial interest in their product. They are very likely interested in prioritizing support to customers who rely on their product as a part of their own product.

As an extension to the second research question there is a more specific scenario where contributor profiles can be relevant. In the Linux kernel project, the development is divided into different branches to allow a large number of people to contribute to the source code repository at the same time. The contributions to each branch is taken care of by different integration managers. These managers have limited capabilities in time and effort. Therefore, if they improve the efficiency of their workflow they can improve their productivity. In this example we are considering the time and effort of the branch integration managers as a resource. This resource is then used for interactions between integration managers and the contributors. The integration managers can then use the profiles presented in figure 4 and 5 to allocate their time and effort. These profiles are relevant because the integration managers have a limited amount of time to do their work. For example, if the integration manager can get a hint about when it is most useful to be active on the mailing list it can contribute to getting more contributions ready to be integrated on time. For the integration manager to be able to better plan his interaction activity he needs knowledge about when the important contributors he is likely to interact with are most likely to be active. The importance of the contributors depends on what kind of work is being done. This is connected to the two already mentioned factors of why contributors get involved in the project. It is then up to the integration manager to decide who to prioritize.

5.1. Validity Threats

In this section we address different threats to the validity of the results in this study. We will cover internal, external, and construct validity threats.

5.2. Internal Validity Threats

The quality of commits: in our study, we analyze the contributor activity based on the quantity of commits. The quality of the commits are not taken into consideration. This threat is realized when a contributor commits a large number of commits but with less quality. In the analysis process used here we will still identify such a contributor as a high activity one.

The accuracy of the time zone data. The relative time according to the time zone differences between the organization member and targeted contributor can not be calculated if the contributor often move between different time zone.

5.3. External Validity Threats

There could be duplicate authors in the data set. In other words there may be duplicate sets of names. Commits can be linked to a specific contributor either by the name or email. If we link a commit to a name there is the problem of different contributors with the same name, if this approach was followed then commits from different contributors would be bunched up into the same pattern. A different approach is to link commits to the email address used. This will ensure that each commit really is linked only to one contributor. But there is also the possibility that a contributor has submitted different contributions through different email addresses and this will result in a single contributor may create more than one pattern. In this study we used the second approach. If other research projects are made on the same data set with the purpose of replicating the results in this study it is very important that the same approach to identify individual contributors is used.

5.4. Construct Validity Threats

In our definition of the categories we have tried to classify the incentive of contributors based on the domain name in their email address. Since there is many thousand contributors in our data set and a large number of different domain names used in the email addresses the categories will inevitably contain certain imperfections. For example, some contributors may

work for a company but only use public email address to commit. These contributors will be categorized into unknown contributors instead of one of the other categories in which they belong. Other contributors may be unknown contributors but be missed in our definition of public email address, this will cause those contributors to be placed in the category of other companies.

6. Conclusion

This study provides theoretic access to profiling and identifying the activity of contributors in FLOSS projects. The category definition is more complex than the official report from Linux Foundation [19]. This research is also the first that targets the identification of activity patterns through profiling the contributors in the Linux Kernel project. Ideally, the results we found strengthens the process of resource allocation and scheduling based on the profiles. The management approaches explained here are able to increase the quality and productivity of the project and can bring the project more close to successful completion.

6.1. Future work

Further research can be done on how to categorize the volunteers based on the information of which branch or technical trees their contributions are related to. This future work will give more concrete and accurate identification of activity and more precise suggestions for scheduling the interactions.

Additionally, in order to precisely and effectively apply the systematic approach of resource allocation which stated by Otero et al [25] on contributors, future work can be research on how to make tasks assignment and scheduled agreement between decision makers and targeted contributors, and how the decision makers can gather the details of workforce of contributors beside just communicating through email.

7. References

[1] Hauge, Ø., Ayala, C., Conradi, R.: Adoption of open source software in software intensive organizations–a systematic literature review. Information and Software Technology 52(11), 1133–1154 (2010)

[2] Bonaccorsi, A., Rossi, C.: Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. Knowledge, Technology & Policy 18(4), 40–64 (2006)

[3] Morgan, L., Finnegan, P.: Benefits and drawbacks of open source software: an exploratory study of secondary software firms. In: Open Source Development, Adoption and Innovation, pp. 307–312. Springer (2007)

[4] Linberg, K. R. (1999). Software developer perceptions about software project failure: a case study. *Journal of Systems and Software*, *49*(2), 177-192.

[5] U.S. General Accounting Office (GAO) (2004). Defense acquisitions: Stronger management practices are needed to improve DOD's software intensive weapon acquisitions. Document number GAO-04-393.

[6] May, L. J. (1998). Major causes of software project failures. *CrossTalk: The Journal of Defense Software Engineering*, *11*(6), 9-12.

[7] Rahman, M. M., & Ruhe, G. (2010). *Resource allocation and activity scheduling: bug fixing perspective*. Technical Report, Software engineering decision support laboratory, University of Calgary.

[8] Koch, S. (2008). Effort modeling and programmer participation in open source software projects. Inform Econ Policy 20(4):345-355. Empirical issues in open source software.

[9] Duggan, J., Byrne, J., & Lyons, G. (2004). A task optimizer for software construction. IEEE Software, 76–82.

[10] Kilamo, T., Aaltonen, T., & Heinimäki, T. J. (2010). BULB: Onion-based measuring of OSS communities. In *Open Source Software: New Horizons* (pp. 342-347). Springer Berlin Heidelberg.

[11] Tsai, H., Moskowitz, H., & Lee, H. (2003). Human resource selection for software development projects using Taguchi's parameter design. European Journal of Operational Research, 153(1), 167–180.

[12] Custers, B. (2004). The power of knowledge. Ethical, legal, and technological aspects of data mining and group profiling in epidemiology. Nijmegen: Wolf Legal Publishers.

[13] Furtado, A., Andrade, N., Oliveira, N., & Brasileiro, F. (2013, February). Contributor profiles, their dynamics, and their importance in five q&a sites. In Proceedings of the 2013 conference on Computer supported cooperative work (pp. 1237-1252). ACM.

[14] Capiluppi, A., & Izquierdo-Cortazar, D. (2013, 12). Effort estimation of FLOSS projects: A study of

the Linux kernel. Empirical Software Engineering, 18(1), 60-88. doi: 10.1007/s10664-011-9191-7

[15] Aaltonen, T., & Jokinen, J. (2007). Influence in the Linux kernel community. In *Open Source Development, Adoption and Innovation* (pp. 203-208). Springer US.

[16] Crowston, K., & Howison, J. (2005). The social structure of free and open source software development (originally published in Volume 10, Number 2, February 2005). First Monday.

[17] Koch, S. (2004). Profiling an open source project ecology and its programmers. Electronic Markets, 14(2), 77-88.

[18] Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical software engineering 14(2), 131164 (2009)

[19] Corbert, J., Kroah-Hartman, G., McPherson, A. (2012) Linux Kernel Development. Linux Foundation. Retrieved from http://go.linuxfoundation.org/who-writes-linux-2012

[20] Palix, N., Thomas, G., Saha, S., Calvès, C., Lawall, J., & Muller, G. (2011, March). Faults in Linux: Ten years later. In ACM SIGARCH Computer Architecture News (Vol. 39, No. 1, pp. 305-318). ACM.

[21] Thomas, L. G., Schach, S. R., Heller, G. Z., & Offutt, J. (2009). Impact of release intervals on empirical research into software evolution, with application to the maintainability of Linux. IET software, 3(1), 58-66.

[22] Shull FJ, Carver JC, Vegas S, Juristo N (2008) The role of replications in empirical software engineering. Empir Softw Eng 13:211-218

[23] CVSAnalY (May, 2014), http://metricsgrimoire.github.io/CVSAnalY/

[24] Robles G, González-Barahona JM, Izquierdo-Cortazar D, Herraiz I (2009) Tools for the study of the usual data sources found in libre software projects. Intl J of Open Source Softw and Proc (IJOSSP) 1(1):24-45

[25] Otero, L. D., Centeno, G., Ruiz-Torres, A. J., & Otero, C. E. (2009). A systematic approach for resource allocation in software projects. Computers & Industrial Engineering, 56(4), 1333-1339.