UNIVERSITY OF GOTHENBURG

A Study of Trust in Open Source Software Communities

Master of Science Thesis in Software Engineering

ALLY TAHIR BITEBO

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
Göteborg, Sweden, September 2014

A Study of Trust in Open Source Software Communities

ALLY TAHIR BITEBO

Supervisor : Imed Hammouda
Examiner: Richard Berntsson Svensson

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 September 2014

Acknowledgements

I would like to thank you all people around me for their help, support and advices during the whole period of doing this thesis and my studies as well. This thesis was successfully finished due to their good cooperation with me academically, socially and financially.

First, I would like to acknowledge the financial support i got from University of Dar es salaam, especially department of Centre for Virtual Learning (CVL) under College of Information Technology (CoICT) for offering me this scholarship to study in Sweden.

Secondly, I would like to sincerely thank my supervisor Dr. Imed Hammouda for his constructive feedback, encouragements and proper guidance during the time of conducting this thesis work. Also, I would like to thank my examiner Dr. Richard Berntsson Svensson for his support and guidance too.

Thirdly, I would like to thank Mr. Peter Degen Portnoy from Blackduct software for helping me to get easy access of downloading more data from www.ohloh.net website.

Finally, I would like to thank my family and friends for their advices, support, encouragement and prayers.

Ally Tahir Bitebo Gothenburg, Sweden, 2014

Abstract

This study developed an algorithm which can be used to identify trust network from evaluation network. The algorithm developed uses global trust value of the members and their evaluation network to approximate local trust between members who are not directly connected to each other. Moreover, the computed approximated local trust was used to examine to what extent evaluation network can approximate trust information within OSS community and the results show that it is possible to approximate trust information by using evaluation network. Furthermore, this study analyses the likeliness of evaluation between members having different trust rank status. So, clustering of members was done and evaluation between groups shows that "Richer gets rich"phenomenon and about 72% of member evaluated other members through their members account profiles and 28% evaluated other members through their accounts as contributors. This means that a lot of members are likely to evaluate other member because they have much of information about their personal details rather than their contribution details in different projects. Finally, the study uses one of the contribution metric known as man month to analyses the evolution of trust ranks against time based on members contributions. Furthermore, results show that the developers contribution will make him or her to be trusted in OSS community. Qualitative study was conducted to analyses the data collected from OpenHub data repository. This is because OpenHub data repository offers data of different projects, developers activities in OSS communities and trust information like kudo rank which are significant base data used to conduct this study.

Contents

Acknowledgements **iii**

Abstract **iv**

List of Figures **viii**

List of Tables **x**

Abbreviations **xi**

1 Introduction **1**

1.1 Background 1
1.2 Problem statement 2
1.3 Purpose 3
1.4 Research questions 4
1.5 Thesis outline 4

2 Literature review and Related work **5**

2.1 Literature Review 5
2.1.1 Evaluation Network 5
2.1.2 Trust in OSS 6
2.1.2.1 Developer perspective 6
2.1.2.2 Code reuse perspective 7
2.1.2.3 Organizational perspective 7
2.1.3 Trust 8
2.1.3.1 Transitivity 9
2.1.3.2 Asymmetry 9
2.1.3.3 Personalization 9
2.1.4 Local Trust and Global Trust Values 10
2.1.5 Trust in Web Based Social Networks 10
2.1.5.1 Trust Network and Trust Metrics 11
2.1.5.2 Challenges of computing trust in social networks 12
2.2 Related Work 13

3 Methodology **15**

3.1 Data Source 15
3.2 Data Collection 16

3.3 Data processing 17
3.4 Research Goals 17
3.4.1 RQ1: How likely a developer will become trusted in the community based on his or her contributions within the community? 17
3.4.1.1 Data Collection 17
3.4.1.2 Data Analysis 21
3.4.2 RQ2: How likely that a developer will evaluate other developer of different trust value? 22
3.4.2.1 Data Collection 22
3.4.2.2 Data Analysis 22
3.4.3 RQ3: How to identify trust network from evaluation network in the open source software community? 23
3.4.3.1 Data Collection 23
3.4.3.2 Data Analysis 23
3.4.3.3 Algorithm 24
3.4.4 RQ4: To what extent can evaluation network approximate trust information in the open source software community? 27
3.4.4.1 Data Collection 27
3.4.4.2 Data Analysis 27
3.5 Data refinement process 27

4 Result Analysis 28
4.1 Results Analysis 28
4.2 Threats to Validity 33
4.2.1 Construct Validity 33
4.2.2 Internal Validity 34
4.2.3 External Validity 34
4.2.4 Reliability 34

5 Discussion 35
5.1 Discussion 35

6 Conclusion and Future Work 37
6.1 Summary 37
6.2 Conclusion 37
6.3 Future Work 39

A Process of sending kudo to other member 41
A.1 The following screen captures shows how a ohloh member can send a kudo to another member 41
A.2 The following screen captures show how a member can send a kudo to a specific project contributor. 46
A.3 The following screen captures show how a member can take back kudo he or she sent before 52

B Table showing summary of developers contributions based on first commit dates 53

Bibliography **69**

List of Figures

1.1 new member 2

2.1 Evaluation network 6
2.2 Onion ring 8
2.3 Trust types and properties 9
2.4 trust metric 12

3.1 xml file 16
3.2 xml file 17
3.3 sample data 24
3.4 evaluation network 24
3.5 adjacent matrix 25
3.6 directed graph 26
3.7 mean local trust 26

4.1 contributor data 29
4.2 different kudorank clusters 29
4.3 project account 30
4.4 same different project 30
4.5 same different project percentage 31
4.6 Adjacent matrix 32
4.7 Estimated local trust 32
4.8 MLT KUDO RANK 33

A.1 user list 42
A.2 user page 42
A.3 kudo message page 43
A.4 kudo confirmation page 43
A.5 kudo summary page for a member 44
A.6 API call xml 45
A.7 project list results 46
A.8 project search list results 47
A.9 project contributors list results 47
A.10 project contributor search list results 48
A.11 project contributor 48
A.12 kudo sent confirmation page 49
A.13 contributor xml API call 50
A.14 contributor xml API call 51
A.15 kudo taking back confirmation page 52

A.16 kudo taking xml 52

List of Tables

2.1 Evaluation betwen developers 5

3.1 Evaluation betwen developers 22

4.1 Evaluation betwen developers 33

Abbreviations

OSS	**O**pen **S**ource **S**oftware
LoC	**l**ines **o**f **C**odes
KR	**K**udo **R**ank
KP	**K**udo **P**osition
NKR	**N**umber of **K**udo **R**eceived
TCRB	**T**otal number of **C**ontributors
TC	**T**otal **C**ommits
TLC	**T**otal **L**ines of **C**ode
MWECB	**M**ature **W**ell **E**stablished **C**ode **B**ase
YECB	**Y**oung but **E**stablished **C**ode **B**ase
VLDT	**V**ery **L**arge **D**evelopment **T**eam
ASDT	**A**verage **S**ize **D**evelopment **T**eam
SD / SDT	**S**ingle **D**eveloper / **S**mall **D**evelopment **T**eam
SNA	**S**ocial **N**etwork **A**nalysis

Chapter 1

Introduction

1.1 Background

Open source software development has emerged as a popular way of developing software in recent years. And, the outcome from these open source software communities is been acknowledged by academy, businesses and government sectors [1–3]. Developers from different areas around the world collaborate to develop software in virtual community which is called open source software community. In addition, contributing to these OSS communities is voluntary work without direction from managerial hierarchy [4]. These voluntary work nature and distribution of developers made trust to be a vital issue within OSS community [4]. A new community member always considered as less trusted member within the community [4]. This is because, he or she needs to show determination and positive contribution before he or she can be trusted in the community [4]. And, one of the factors which motivate a developer to continuously contribute to OSS community is social reputation, which is based on positive evaluation from other developers within a community [5]. Another factor is interpersonal trust between developers within OSS communities which plays important role on team effectiveness on OSS development process [3]. So low level of trust within OSS communities is associated with decreased number of contributors in particular project [2].

To study trust with in OSS community, this study will model trust as follows. A community members having high reputation value are considered to be more trusted and community members having low reputation value are less trusted [6] as illustrated in

Figure 1.1 below.This assumption was adapted from online information system domain experience. For example in e-business and recommendation systems where a user is more likely to be trusted due to the large number of positive evaluations and less trusted with negative evaluations [6].

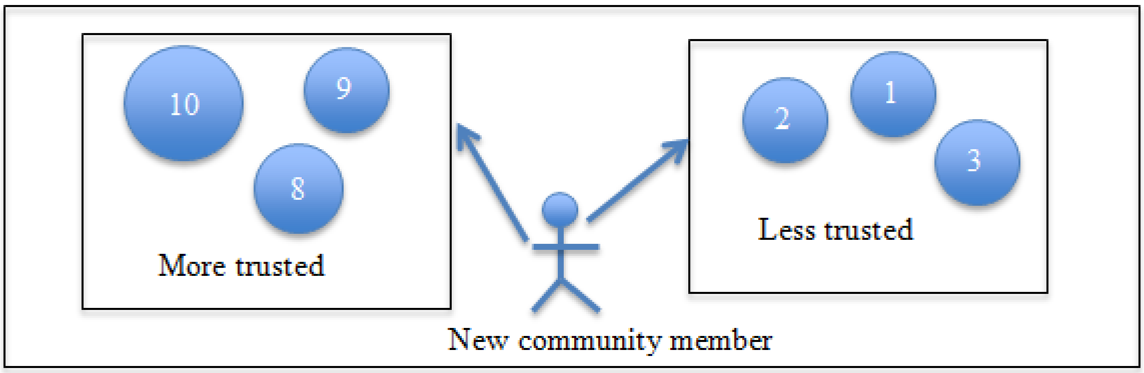


FIGURE 1.1: shows how a new community member will categorize existing member in the community. The circles represent members and numbers within the circles are reputation values

This study will compute approximated local trust between members within a community by using evaluation networks. The algorithm developed manages to compute approximate local trust between members who are not directly connected within the network.

1.2 Problem statement

Trust is an important issue in OSS communities [4] [1]. This is because, it is not possible to interact with all of the members contributing to different OSS projects. So, one of the major challenges facing OSS communities is trust. Firstly, is how members can trust each other [4] and this challenge is similar to other web based social networks [7] [8]. However, those researches developed algorithm like Mole Trust [7] and Tidal Trust [8] which were used to predict trust scores of members who are not directly connected in the network. On the other hand, no previous studies applies those algorithm to study trust in kind of evaluation network like OSS community.So, this study will use evaluation network to study trust within OSS community.

Another trust challenge facing OSS community is how to trust a member based on his or her contribution [4]. However, a member can contribute in OSS community by participating in different activities like software development, software testing, writing software documentation, participating in project forum and communicating with other members [9]. But, one of the important factor influencing trust between developers is technical skills [10]. Moreover, some of the research measured developer technical skills by using commits as a metric where they categorize commits based on LoC [11] and number of work weeks a member devoted to projects as team effort [2]. But, this study will use man month metric introduced in OpenHub data repository (http://www.openhub.net/) to measure members technical contribution and study trust within OSS community. Furthermore, there is research gap in evaluation networks of OSS communities [12]. This is because most of the previous researches concentrate more on studying collaboration networks than evaluation networks in OSS communities [12]. However, the results from this study [13] shows that; homophily factor like same country, same location, same programming language and same community status will influence a developer to positively evaluate another. But, still there is some gap in this context in case of phenomena like participation in same project and evaluation of members through their accounts. For instance, members evaluating each other through their personal accounts or through their accounts as contributors.

1.3 Purpose

The purpose of this thesis thesis is to analyse the possibilities studying trust within OSS communities. Firstly, the thesis investigate the possibility of studying trust in relation to members contribution within the community. This part of thesis will contribute to previous studies like [4] were the study discussed the possibility of inferring trust between members based on contribution. Another study is [11] where they discussed about developer contribution in term of LoC in given commits but this study applied another metric which is man month to categorize developers contribution and use it it to study the relationship between members contributions and trust in OSS communities. Secondly, this thesis investigate the effect of evaluation between developers having different community status in the OSS community. Additionally, this study thesis also aim to analyse the evaluation distribution between members and how is affected if members are contributing in the same projects or different projects. This part of the thesis will contribute to the previous study [13] where they found that; evaluation between members in OSS community is affected by homophilic factors like same country, same location, same programming language and same community status. Thirdly, this thesis investigate the possibility of using evaluation network of OSS community to extract trust network. The main goal of this part is to use OSS community evaluation network to study trust by using methods applied in other web based social networks like in these two studies [7] and [8]. This section includes implementation of algorithm which was used to transversing through the network which gives us approximated local trusts between network nodes. Moreover, the mean approximated trust of each nodes was used to analyze to what extent can OSS community evaluation network can approximate trust information within OSS community.

1.4 Research questions

RQ1: How likely a developer will become trusted in the community based on his or her contributions within the community?

RQ2: How likely that a developer will evaluate other developer of different trust value?

RQ3: How to identify trust network from evaluation network in the open source software community.

RQ4: To what extent can evaluation network approximate trust information in the open source software community?

1.5 Thesis outline

The rest of the report is organized as follows. Section 2 describes the overview of previous related researches and Section 3 introduces methodology used to conduct this thesis. Section 4 covers the summaries of the findings and threats to validity of this study. Moreover, in section 5 result discussion is presented. Finally, this study thesis conclusion and discussion of possible future research is presented in section 6.

Chapter 2

Literature review and Related work

2.1 Literature Review

2.1.1 Evaluation Network

Evaluation network is the relationship between developers within the open source community, where developers are represented as nodes and the link between them is evaluation between two developers as illustrated in Figure 2.1 and Table 2.1 below [5] [14] [6]. In ohloh data repository website a developer can send a vote of thanks or appreciation called kudo to another developer due to his or her contribution to form a link between those developers who are evaluating each other [5] [6].

Table 2.1: Evaluation betwen developers

Developer	Evaluated by developer
D1	D6
D2	D6
D3	D6
D4	D5
D5	D3
D6	D6

In Table 2.1 above shows evaluation between developers. For example in the first row developer D6 was evaluated by developer D1.

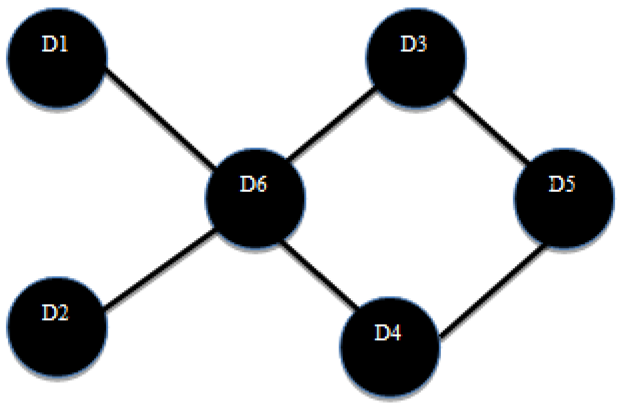


Figure 2.1: An illustration of evaluation network between developers.

In Figure 2.1 below shows the evaluation network between developers where nodes representing developers and link between then representing evaluation between two developers as shown in Table 2.1 above.

2.1.2 Trust in OSS

Open source community developers are located in different countries around the world. And, these developers uses internet as a medium of interacting with each other [5]. But online virtual environment offered by OSS community faces challenges of online anonymity [15][16][5]. Additionally, online anonymity raises the issue of trust among developers interacting in OSS community. Trust has been studied in different perspectives in open source software domain. One of them is in developer perspective which explains interpersonal trust between developers. Trust in open source software code base is another perspective which deals with trust issues in software code written by different contributors in open source community. Finally, in organizational perspectives which show how organizations adapting open source software can build trust with OSS communities.

2.1.2.1 Developer perspective

One of the factors which can lead to interpersonal trust within OSS community members is lacking of managerial hierarchy such as scheduling and deadlines [3]. Additionally, having contributors from different organizations with different motivations within a community [10]. For example, new members of the community are always considered as not trusted and he or she must shows positive contributions in the community so as to build some trust with other developers within the community [4]. On the other hand, interpersonal trust between developers is important to build or strengthen the community [2] [3].Team effectiveness is the ability to attract developers to join an open source software community and continue voluntarily contributing in the project [3]. Interpersonal trust can be affective trust or cognitive trust [2] [3]. Affective trust is related to psychological and emotional attachment between developers within the open source community and this shows how team members treat each other in the open source software community [3]. And, Cognitive trust is based on rational assessment between developers within the community and this shows how a newcomer or existing members that are willing to continuously contribute in the project by assessing the team development ability and project development process [3]. So trust between developers working in the open source community plays an important role to the community health and sustainability.

2.1.2.2 Code reuse perspective

Code reuse is one phenomenon where a developer reuses his or her codes written in the past or reuses other developers code [17]. This is one of the common software engineering practice so as to save development time and cost [14]. Open source software component have been used by different companies products as plugins or modules [16]. Trusting code developed by another developer has been one of the challenges in code reuse software engineering practice [17][14] [16]. For example, there is a risk of integrating a full open software component developed by other developers and a developer is likely to integrate changes from other developers if there is trust between them [16]. In code-search development, where developers tend to assess the search results obtained from both technical and human factor before integrating the codes to his or her work [17].

2.1.2.3 Organizational perspective

Some of the companies reuse open source software components to gain competitive advantages by customizing or uses value added services [16]. However there is a risk of integrating full codes developed by another developer as expressed in previous section. In contrast, there are some companies that wants to release their product as open source software, where they need to build a network of trust within a community before the release of their software as open source [1]. Trust is important to motivate the community to continue developing open source software, because the released software needs sustainable community to survive [1]. In large open source community like Linux Kernel community, they follow a model of onion like shape where a member innermost layer are considered as trusted member or core member and are the ones who control the code base of the software by filtering which code updates can be integrated in main software codes. And, most outer layer are considered as less trusted or passive users [3] and [10].

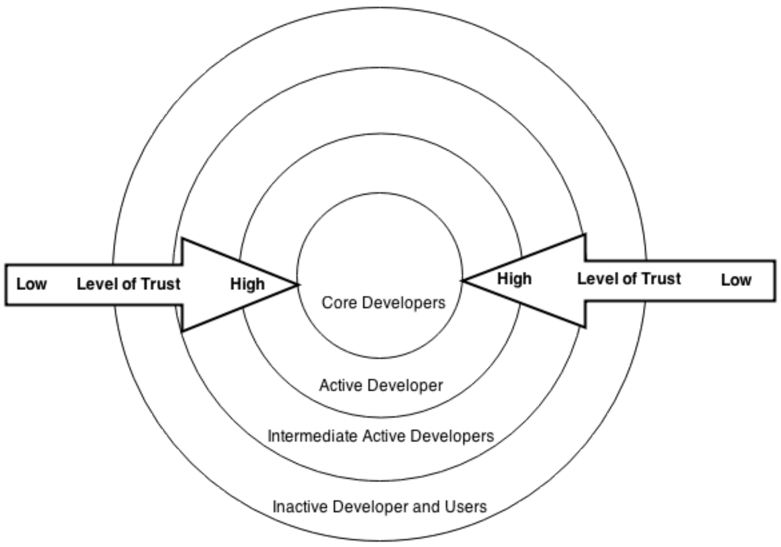


Figure 2.2: Onion ring.

2.1.3 Trust

Trust has been defined as the relationship between people where one person is taking a risk to accept other person action. Goldbeck(2013) defines trust as *A person trusts another if she is willing to take a risk based on her expectation that the trusted persons actions will lead to a positive outcome.* Stewart and Gosain (2006) defines trust as *the extent to which a person is confident in, and willing to act on the basis of, the words, actions, and decisions of the other.* Both of these definitions suits well in the open source software community, since there are risks of accepting unknown developer to contribute his or her code in open source project. We always hoping that, the developer will contribute good codes without going against the specified software features. Trust has three main properties which are transitivity, asymmetry and personalization [8] as illustrated in Figure 2.3 below.

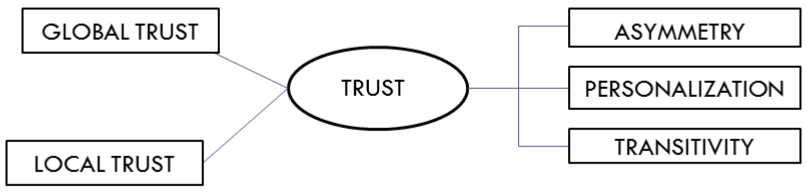


FIGURE 2.3: Diagram to show trust types like global and local trust and trust properties like transivity, asymmetry and personalization .

2.1.3.1 Transitivity

Transitivity is one of the primary characteristic of trust [8]. In this case trust has been considered as been propagated or inferred from source node to sink node through intermediate nodes between source node and sink node. One of the common example used is if Alice trust Bob and Bob trust John, so there is greater chances that Alice will some how trust John [8]. This phenomenon is called Friend Of a Friend (FOAF) [8]. Of course, it is easier to trust a friend of a friend or people whom we trust than a stranger [18].

2.1.3.2 Asymmetry

Trust relationship between two people must not be equal in both sides [8]. For instance, if Alice trust Bob by trust rating 0.9. It is not necessary that Bob will trust Alice with the same value 0.9. One of the real world example is trust between parents and children. Children can trust their parents with a high level of trust but parent will always have low level of trust to their children [19].

2.1.3.3 Personalization

Trust statement between two people is the personal opinion between people base on their interacts and history between them. So, its more likely Alice and Bob to have two different trust statement to John. For example, Alice may trust John by 0.7 and Bob at the same time trust John by 0.3 [8] [20].

2.1.4 Local Trust and Global Trust Values

Local trust value is the personalized score between two members in trust network [7]. This means that how member A should trust member B. On the other hand, Global trust value is aggregate score computed over the network and is visible by all members of the given network [7].

2.1.5 Trust in Web Based Social Networks

In this modern world, people use internet as one of the major source of information. And, internet offers more opportunity for people to work or interact online even though logically are living in different geographical location. Example, in a open source software community where developers from different countries can collaborate and develop software together without physically met or know each other [4]. Even commercial transaction happens between strangers in online websites like Ebay [7]. Furthermore, an increasing number of social networks where most of people use for business, friendships and online collaboration and increase of online contents trust emerges as a vital issue [18] [7] [21]. Another challenge is to filter those millions of web contents information [21]. The challenge can be observed in online websites like Ebay, Epinions or Amazon where people can write reviews of different product in the website. So, the question rises how can a user trust information supplied by another user? [22]. And, how can users trust each other? [18] [7] [22] [19]. Those online websites mentioned above uses trust information and reputation systems to address this challenge of content filtering [22]. Firstly by allowing users to rate each other based on previous transaction. Later, the system computes the reputation score of the user so as to be used by other user to decide whether to interact with this user or not [22]. This aggregated score for specific user is called Global Trust where is visible by all users of the given system. Example of system that uses Global Trust are Ebay user feedback system and Google page ranking system [7] [22] [23]. On other hand, systems like Epinions website users can directly rate a specific user by expressing how much he or she trust other users of the system [7] [23].

2.1.5.1 Trust Network and Trust Metrics

Trust network is a directed graph with nodes and weighted edges [7] [23]. Edge direction indicates the flow of trust statement from source node to destination node and edge weight indicates level of trust which most of the systems can range from 0 to 1. These trust statements are users opinions to other users in the system [7]. However, in most web based social network existing today are composed with many nodes. For example, in most popular open source software community like Linux Kernel where number of contributors can reach 1000. Naturally, It will be difficult to interact with most of them even though they work in virtual online environment. In other words, developer will have small chance to interact with other developers and express his or her trust statement which may be used in trust network. So, most of the network will be with unknown users which are not directly evaluate each other due to the size of the network [7]. And, there is a need to approximate or predict trust statement between unknown nodes in trust network. The process is known as trust propagation or trust inference from source node to sink node through the network path [7][8].

Trust metrics are mathematical computational algorithm used to propagate trust in trust networks. These algorithms are used to calculate trust within trust network. There are two types of trust metrics which are global trust metric and local trust metric . Global trust metric computes global trust values of each node in the network. One of the example of the global trust metric is PageRank algorithm used by google to rank web pages [8]. Additionally, there is local trust metric that propagate local trust between source node and sink node [19]. One of the common cited local trust metric is TidalTrust algorithm introduced by [24]. For example, in Figure 2.4 below source node knows node A and B since they are directly connected to source node. Source node need to pass three different path to reach sink node. First path is through node B which is directly connected to sink node. Secondly, is to pass through node A then node C then it will reach sink node. Finally, is through node A then node D then sink node. To infer trust in Figure 2.1 from source node to sink node, TidalTrust algorithm use modified breadth-rst search algorithm to search the sink node [8]. At first, source node contact its neighbouring node A and node B about the sink node. Since node B has directly connected to a sink node the local trust value will be reported back to source node. Furthermore, node A is not connected directly to sink node. So, it will contact neighbour node C and node D about sink node and since both of them are connected to sink node they will report back the local trust value and the average weight of their trust ratings [23] [19]. This process of contacting neighbour nodes and return their average rating of the sink node will be repeated to every node until the propagated or inferred trust of the sink is obtained [23].

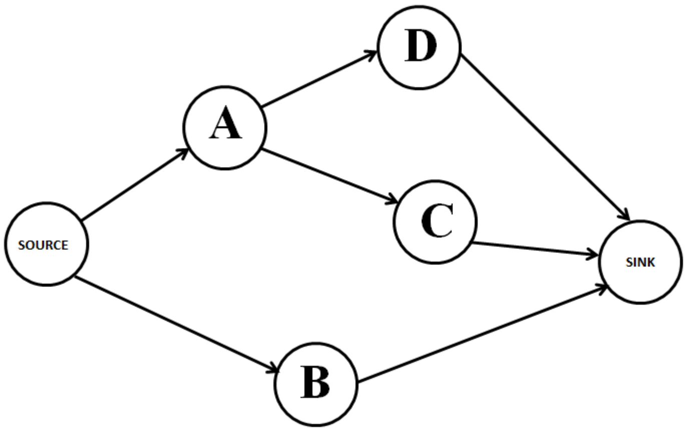


FIGURE 2.4: Trust network to show connection from source node to sink node.

2.1.5.2 Challenges of computing trust in social networks

Modelling of trust for mathematical computational such as algorithmic trust metric is difficult task [20]. Firstly, trust is personal opinion from one person to another and it depends on wide range of factors such as background information between them, reputation they holds in the community and history of their previous interactions [18] [20]. Secondly, Goldbeck (2008) added that, trust depends on the context. For example in open source software community a newcomer can be trusted to submit small changes in existing code base than integrating his or her own new developed plugin or component. And finally, trust between people varies over time because the more people interact and know each other behaviour very well the level of trust between them can vary [20]. So, trust can be built or destroyed over time. For example a newcomer in software community can build trust by submitting small patches, helping others and do software testing [4].

2.2 Related Work

Developers in OSS project tend to put their effort voluntarily to participate in OSS development activities. However, these developers includes different beneficial assessment such as project usefulness, reputational benefits and psychological or emotional benefits so as they can remain involved in OSS project [2]. Additionally, low level of trust in these virtual communities may be associated with decrease in number of contributors participating in OSS development [2]. This study [2] shows that Affective trust support both team size and team effort, where they defined team size is the number of developers associated with the given project and team effort as the number of work week a contributor devoted to the project. On the other hand the study shows that cognitive trust support neither team size nor team effort. Another finding is the quality of communication between contributors will enhance the process of task completion with the community [2]. Furthermore, it was founded that; the most important factor influencing trust between developers are technical skills, their reputation and informal and formal practices within the community [10]. So, developers participation may be affected by how they are interacting and treating each other in OSS communities.

Developer determination to continue participating and contributing to an OSS project is really important for the survival of given project. However, there are different kinds of developers contributions which are committing lines of codes, forum participation, software documentation, writing project wiki and participating in communication media like mailing list and in instant chat [9]. But, one of the important contributions which adds OSS values are technical knowledge[10] [2] and communication quality between contributors [2]. Additionally, there were different proposed types of metrics used to measure contributors contributions. One of them is commits based on LoC done in this study [11] where they propose three different types of commits named as single commits which are 1 to 100 LoC, aggregate commits from 101 to 10000 LoC and finally repository refactoring more than 10000 LoC. Another metric used was number of work weeks a contributor devoted to the project and they found that it was directly support contributors task completion in a given project [2]. Nevertheless, De Laat in his study [4] pinpointed the problem of trusting contributors based on their contributions. This study also found the possibilities of using existing potential contributors who are already trusted so as to infer trust within the community. Additionally, the strong inference trust or weak inference of trust will depend on roles of contributors and past history or past performance [4].

Social reputation is one of the factors motivating a contributor to participate voluntarily in OSS communities [13]. For instance, positive evaluation from other community members. Factors that influence a developer to positively evaluate each other are like members number of positive evaluation he or she receipt before, shared affiliations shared between members and homophily factors like same location, programming language and community status [13]. Additionally, comparison between collaboration network and evaluation network was done using social network analysis (SNA) and the results shows that; number of positive evaluation contributor received is not related to number of collaboration he or she has [12] [10]. Moreover, the evaluation network in more connected than the collaboration network [12]. Finally, both evaluation network and collaboration network has a small world and scale free network properties. For example small average path length, high clustering coefficients and power-law degree distribution [12]. Furthermore, most of social netwroks have the small world network properties [8].

In todays modern web based social networks which connect different people around the world, trust in these social networks has been emerged as one of the important issues to consider [7] [8]. Furthermore, it is not easy to interact with all the members in such kind of networks because of number of members within the community. So, using of trust metric to infer trust relationship within the community between members who are not directly connected in the network [7] [8]. Additionally, both studies [7] [8] shows that algorithm used were able to predict trust scores in these web based social networks.

Chapter 3

Methodology

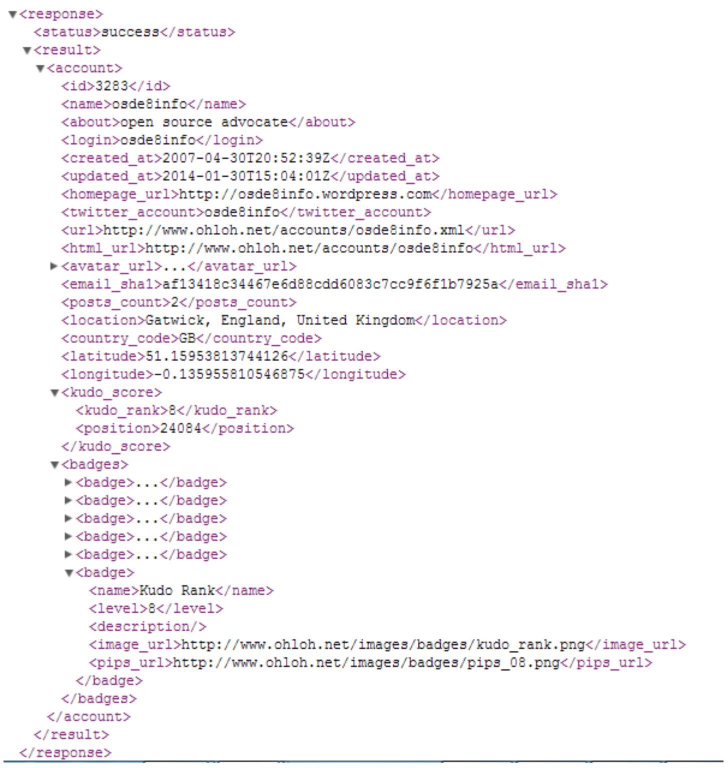
This study was conducted using qualitative data analysis method. This is because of the nature of the research goals, existing theory and data source available. This section describes the data source used and techniques used to perform data processing and data analysis. The following sub sections explains in details

3.1 Data Source

The study was conducted by using data collected from OpenHub data repository (http://www.openhub.net/) formerly known as Ohlol. This repository holds free information about open source projects and contains data of developers, code history, main programming languages, open source projects and organizations who manages those open source projects. OpenHub collects these data from different version control repositories holds open source projects like git, subversion, mercurial, CVS and bazaar. Additionally, this study choose OpenHub data repository because it contains information about evaluation between developers so it will be easy to construct evaluation network and it also contains some trust information like kudo rank which considered as global trust value in this study. Furthermore, OpenHub can be accessed using their API which is well documented at this link (https://github.com/blackducksw/ohloh_api). To access OpenHub data through API, you need to be a member and one needs to request for an API key [14]. Moreover, the data collected In this study was about members account information, contributors data, members history about the kudo they sent and kudo they receive and projects information.

3.2 Data Collection

Data collected using Ohloh API calls with the results was an xml file formats as shown in Figure 3.1 below. To conduct this study the following data were collected; members account data, kudo received data, kudo sent data, contributors data and project data. Additionally in ohloh a member and a contributor are two different kinds of information. A member is a person who registered as a user in ohloh website and a contributor is the person who contributes in open source project or projects [14]. A contributor can be a member and claims his or her contribution through his or her account.

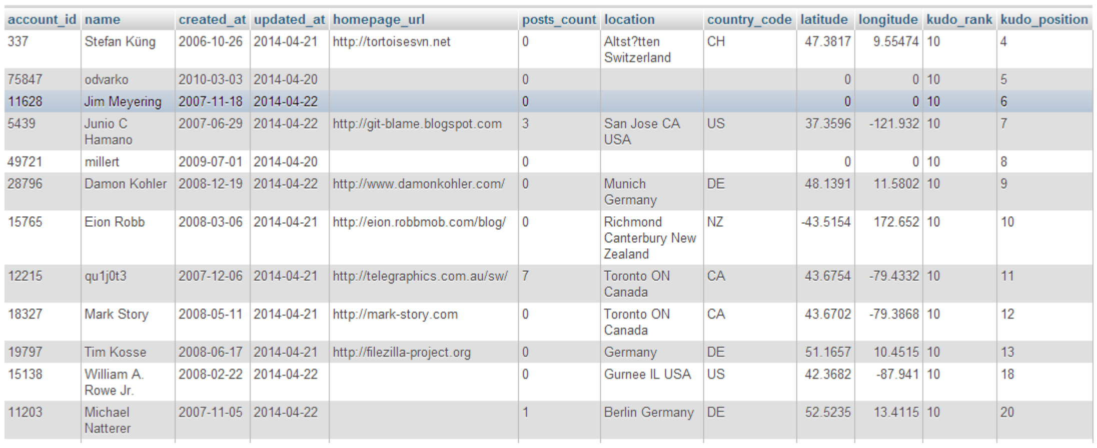


```
▼<response>
   <status>success</status>
  ▼<result>
    ▼<account>
       <id>3283</id>
       <name>osde8info</name>
       <about>open source advocate</about>
       <login>osde8info</login>
       <created_at>2007-04-30T20:52:39Z</created_at>
       <updated_at>2014-01-30T15:04:01Z</updated_at>
       <homepage_url>http://osde8info.wordpress.com</homepage_url>
       <twitter_account>osde8info</twitter_account>
       <url>http://www.ohloh.net/accounts/osde8info.xml</url>
       <html_url>http://www.ohloh.net/accounts/osde8info</html_url>
      ▶<avatar_url>...</avatar_url>
       <email_sha1>af13418c34467e6d88cdd6083c7cc9f6f1b7925a</email_sha1>
       <posts_count>2</posts_count>
       <location>Gatwick, England, United Kingdom</location>
       <country_code>GB</country_code>
       <latitude>51.15953813744126</latitude>
       <longitude>-0.135955810546875</longitude>
      ▼<kudo_score>
         <kudo_rank>8</kudo_rank>
         <position>24084</position>
       </kudo_score>
      ▼<badges>
        ▶<badge>...</badge>
        ▶<badge>...</badge>
        ▶<badge>...</badge>
        ▶<badge>...</badge>
        ▶<badge>...</badge>
        ▼<badge>
           <name>Kudo Rank</name>
           <level>8</level>
           <description/>
           <image_url>http://www.ohloh.net/images/badges/kudo_rank.png</image_url>
           <pips_url>http://www.ohloh.net/images/badges/pips_08.png</pips_url>
         </badge>
       </badges>
     </account>
   </result>
</response>
```

FIGURE 3.1: Xml file returned after calling Ohloh API.

3.3 Data processing

Java application was developed to call Ohloh API and store the results to a structure text file. Then, the text file was imported to database for easy processing as shown in database snapshot in Figure 3.2 below.



account_id	name	created_at	updated_at	homepage_url	posts_count	location	country_code	latitude	longitude	kudo_rank	kudo_position
337	Stefan Küng	2006-10-26	2014-04-21	http://tortoisesvn.net	0	Altst?tten Switzerland	CH	47.3817	9.55474	10	4
75847	odvarko	2010-03-03	2014-04-20		0			0	0	10	5
11628	Jim Meyering	2007-11-18	2014-04-22		0			0	0	10	6
5439	Junio C Hamano	2007-06-29	2014-04-22	http://git-blame.blogspot.com	3	San Jose CA USA	US	37.3596	-121.932	10	7
49721	millert	2009-07-01	2014-04-20		0			0	0	10	8
28796	Damon Kohler	2008-12-19	2014-04-22	http://www.damonkohler.com/	0	Munich Germany	DE	48.1391	11.5802	10	9
15765	Eion Robb	2008-03-06	2014-04-21	http://eion.robbmob.com/blog/	0	Richmond Canterbury New Zealand	NZ	-43.5154	172.652	10	10
12215	qu1j0t3	2007-12-06	2014-04-21	http://telegraphics.com.au/sw/	7	Toronto ON Canada	CA	43.6754	-79.4332	10	11
18327	Mark Story	2008-05-11	2014-04-21	http://mark-story.com	0	Toronto ON Canada	CA	43.6702	-79.3868	10	12
19797	Tim Kosse	2008-06-17	2014-04-21	http://filezilla-project.org	0	Germany	DE	51.1657	10.4515	10	13
15138	William A. Rowe Jr.	2008-02-22	2014-04-22		0	Gurnee IL USA	US	42.3682	-87.941	10	18
11203	Michael Natterer	2007-11-05	2014-04-22		1	Berlin Germany	DE	52.5235	13.4115	10	20

FIGURE 3.2: Database snapshot of the members account table .

3.4 Research Goals

3.4.1 RQ1: How likely a developer will become trusted in the community based on his or her contributions within the community?

3.4.1.1 Data Collection

To address this research goal, the study collects the following data about developers contributions. The data collected was members account information, kudo history data, contribution data and project data.

Member Account data An account data holds information about a member of ohloh website. This dataset holds several information about the member account like

- Account_id - the unique id of the registered member in ohloh website.
- Name - the name of the account holder [25].
- Created_at - Date where account was created [25].

- Updated_at - Date where account was updated [25].
- Homepage_url - Is the url of the member websites or blog [25].
- Post_count - This field shows number of posts made by a member in ohloh forum [25].
- Badges owned by account holder and one of the interesting points of this study is kudo score.as shown in figure 3.1 above.

Kudo score badge has two attributes which are kudo rank which is the number between 1 and 10 and kudo position. Kudo rank is the ranking scheme which is calculated based on number of kudo a specific member received from others and other factors like project stack and his or her contributions in the those projects [14]. The default kudo rank of newly account is kudo rank 1. Kudo position shows member position in the website based on his or her contributions.This study holds 563,427 information about registered members in ohloh data repository.

Project data A project dataset holds information about open source projects stored in ohloh website. The following data were collected about different open source projects;

- Project_id is the id of given open source software project [25].
- Project name is the name of the open source software project stored in ohloh website [25].
- Created_at is the date were project were added to ohloh website [25].
- Updated_at is the latest time the project were modified [25].
- Homepage_url is the homepage of the given open source software project [25].
- Project user count is the number of users who votes in ohloh website as are the users of this project [25].
- Average user rating is the number of rating a user votes to this project and these ratings are floating number from 1.0 to 5.0 where 1.0 is the lowest and 5.0 is the highest ratings [25].
- Number rating is the total number of users who have rated this project [25].

- Number of reviews is the total number of users who have write the review about the project [25].
- Analysis which shows the general analysis of the given project such as number of contributors, number of commits, project size, project age, project activities and past twelve month summaries in term of number of commits and number of contributors [25].

This study holds 662,439 data of open source software projects records.

Contributors data A contributor dataset holds information about peoples who contributes in different open source projects. This study contributor dataset holds 844,012 data of contribution of developers in different open source software projects and their activities are recorded in Ohloh website. The following data were collected about different contributors;

- Contributors id is the id of the specific contributor [25].
- Account id is the account id of the contributor if he or she registers an account in ohloh data set and claims specific contribution. This field will be null in this study database if the contributor does not have an account in ohloh website [25].
- Account name is the account name of the contributor if he or she is a member in ohloh website. This field will be null in this study database if the contributor does not have an account in ohloh website [25].
- Contributor name is the name used by a contributor when committing his or her codes to repositories [25].
- comment ration the fraction of new lines of code added by a contributors which are comments [25].
- First commit time is the first date a contributor commits his or her work [25].
- Last commit time is the last date a contributor commit his or her work [25].
- Man month total number of calendar months which a contributor made at least one commit [25].
- Commits are total number of commits made by specific contributor [25].

- project id is the id of the project where this contribution was made [25].

Kudo received history data Kudo received dataset holds information about history kudo received by a specific member. In this case the following data were collected about sender account id, sender account name, receiver account id, receiver account name, project id, project name, contributor id, contributor name and date where the the kudo was received was downloaded and stored in database. This study holds 46,926 information about kudo received by different ohlol members.

Kudo sent history data Kudo sent dataset holds information about history kudo sent by a specific member. In this case the following data about sender account id, sender account name, receiver account id, receiver account name, project id, project name, contributor id, contributor name and date where the the kudo was sent was downloaded and stored in database. This study holds 57,458 records about kudo sent by different account holders.

The process of sending kudo can be directly to a member account or to a member who contributes to a specific project. These two scenarios are shown in the appendix A and was recorded differently in this study as explain in the following data field description project_id, project_name, contributor_id and contributor_name

Kudo sent and kudo received dataset shares the same attributes definitions as explained below.

- Sender account id is the account id of the member who sends a kudo [25].
- Sender account id is the name of a member who sends a kudo [25].
- Receiver account id is the account id of the member who receives a kudo [25].
- Receiver account name is the name of a member who receives a kudo [25].
- Project id is the id of the project where contributor receives a kudo instead of his or her account [25]. This field will be null if the kudo sent to member account in this study database.
- Project is the name of the project where a contributor receives a kudo instead of his or her account [25]. This field will be null if the kudo sent to member account in this study database.

- Contributor id is the contributor id of the contributor if kudo was sent to a project contributor instead of member account [25]. This field will be null if the kudo sent to member account in this study database.

- Contributor name is the name of the project contributor if kudo was sent to a project contributor instead of the account [25]. This field will be null if the kudo sent to member account in this study database.

- Created at is the date were kudo was sent or received [25].

3.4.1.2 Data Analysis

This study analyses the possibility of a developer to become trusted based on his or her contributions in the OSS community. So, the data was grouped according to first commit date done by different members to any of the project he or she was contributed. The results shows that number of members falls in this group category ranges from 1 to 39. The given Table 3.1 below shows only top ten of the grouped members based on first commit date. Then the first three dates was selected and members contributions were analyzed as illustrated in Appendix B.

Developer contribution in OSS community is not only by committing lines of codes but also can be in different forms like being active in project forum, software documentation, writing project wiki and be active in mailing list [9]. Additionally, usually basic metric used to measure developers contribution in OSS is commuting the LoC [9] [11]. On the other hand this study data holds number of commits done by a specific contributor without specifying quantity of LoC included in those commits. So, one of the contribution criteria used in this study will be man month values. Man month is the number of month were a contributor did atleast a single commit [25]. The months were a contributor did not commit any code were not counted [25].

Table 3.1: Evaluation betwen developers

Number of contributors	First commit date
39	2012-03-26
39	2012-07-05
36	2012-05-10
35	2011-10-03
34	2012-09-04
34	2013-01-28
33	2012-06-12
33	2013-03-11
33	2012-05-15
33	2012-05-31
33	2012-03-21

3.4.2 RQ2: How likely that a developer will evaluate other developer of different trust value?

3.4.2.1 Data Collection

To address this research goal, the study collects kudo history data as explained in section 3.4.2 above.So, this study use kudo sent history and kudo received history data.

3.4.2.2 Data Analysis

The study examined evaluation between developers having different trust values. For instance, evaluation between members having different kudo ranking. To achieve this, the study categorise the members in clusters according to their kudo rank and study the transaction of kudo history between those clusters. Firstly, clusters were divided as follows based on members of kudo rank (9 and 10), kudo rank (7 and 8), kudo rank (5 and 6), kudo rank (3 and 4) and kudo rank (1 and 2). Moreover, this study categorizes the process of sending or receiving kudo between members into two groups. The first group is when a member sends or receives a kudo directly to his or her account. Secondly, is when a member sends or receives a kudo as a contributor of specific project. In this second scenario a member can receive kudo due to his or her contribution in different projects. For example, member A can receives a kudo due to his or her contribution in project X and at the same time member A can still receive a kudo due to his or her contribution to project Y. These two scenarios are different and are well explained in appendix A. Finally, the study continue to analyse the distribution of kudo history based on the members contributions in either the same projects or different projects.

3.4.3 RQ3: How to identify trust network from evaluation network in the open source software community?

3.4.3.1 Data Collection

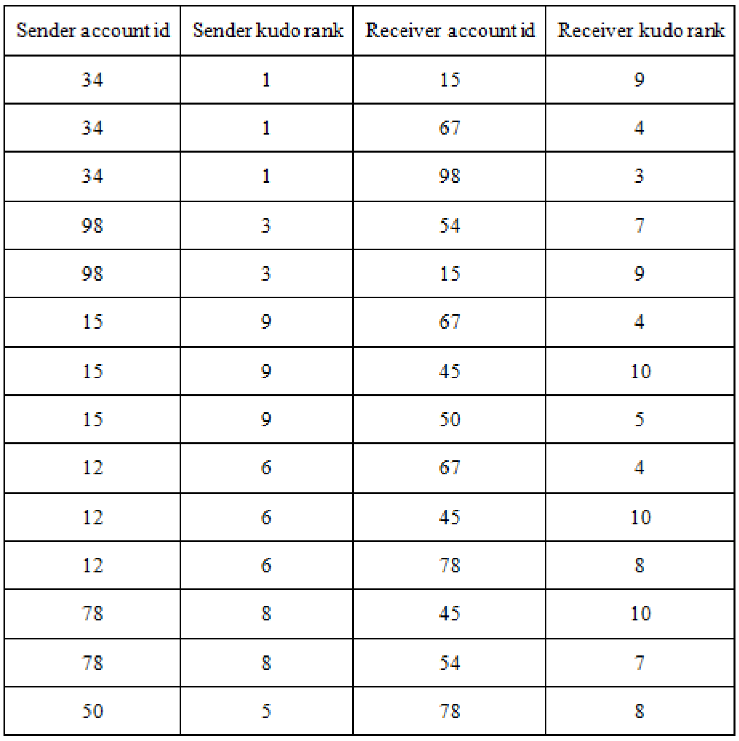
To address this research goal, the study collects the kudo history data and contributors data and project data as explained in section 3.4.2 above.

3.4.3.2 Data Analysis

Data from kudo sent history and kudo received history was used to track members history of evaluation activities between each other and capture the scenario who evaluates who?. Further more, an evaluation network was constructed where nodes are members id and evaluation between them as edges. Moreover, this study evaluation network have 15,664 nodes and 46,947 edges. The next step was to construct trust network out of evaluation network but this study faced one of the biggest challenge which is the missing of ground trust scores between members in the Openhub data repository but they have kudo rank as estimated global trust score in each nodes. For example, a member will just sent a kudo which is equal weighted evaluation without specifying how much he or she trust the member who receive that kudo. On the other hand, most of the previous studied web based social network like Epinions.com. There are trust statements between members. For example, member A will rate member B by 0.7 which means a trust statement can be modeled as $t_{AB} = 0.7$ meaning that member A trust member B by 0.7 trust score. These values will depend on the study data. However, having kudo rank as global trust for each member (node) in the this study evaluation network, this study uses kudo rank as base trust information and calculate what the study argue to be estimated local trust. So, this study develop an algorithm that uses kudo rank to estimate the local trust between members.

3.4.3.3 Algorithm

Algorithm was developed to approximate the local trust values between members who are not directly connected to each other. By using evaluation network data (who evaluates who?) and kudo rank assigned to each node.The algorithm developed inherits some of TidalTrust algorithm procedures explained in 2.1.5.1 above. To show development of this study algorithm, sample data of evaluation between developers was introduced as shown in Figure 3.3 below and its evaluation network of the sample data as shown in Figure 3.4 below.



Sender account id	Sender kudo rank	Receiver account id	Receiver kudo rank
34	1	15	9
34	1	67	4
34	1	98	3
98	3	54	7
98	3	15	9
15	9	67	4
15	9	45	10
15	9	50	5
12	6	67	4
12	6	45	10
12	6	78	8
78	8	45	10
78	8	54	7
50	5	78	8

FIGURE 3.3: Sample data of evalution between developer with their given kudo rank.

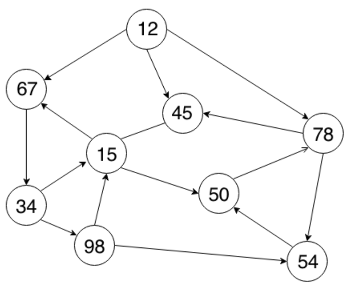


FIGURE 3.4: Evaluation network of the sample data. Nodes represents members and number inside the node is the member id. The arrows represents the kudo sent from source node to destination node

Steps used to develop the algorithm are as follows. At first, Adjacent matrix from evaluation network shown in Figure 3.5. was built to maintain the structure of our graph and form direcred graph shown in Figure 3.6. Secondly, in the adjacent matrix the field having 1 was replaced by a sender kudo rank then the algorithm was applied to a graph so as to approximate local trust of unknown nodes from the source node.

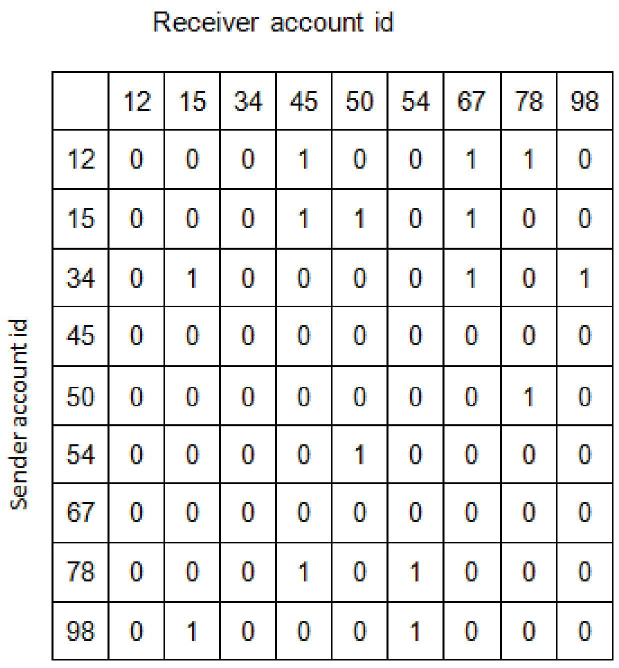
Steps of computing the approximates of local trusts

i. Source node is identified.

ii. Source node identifies sink nodes which are not directly connected to source node but can be reached through neighbours nodes.

iii. Source node neighbours reports back the approximated local trust of the sink which is the neighbour kudo rank if are directly connected to them. If not, the neighbours of neighbour nodes will report back the approximated value. This process is repeated until the sink approximated local trust is determined.

iv. The source node will take average of returned approximated local trust from its neighbours or neighbours of the neighbours.


Receiver account id
Sender account id

	12	15	34	45	50	54	67	78	98
12	0	0	0	1	0	0	1	1	0
15	0	0	0	1	1	0	1	0	0
34	0	1	0	0	0	0	1	0	1
45	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	1	0
54	0	0	0	0	1	0	0	0	0
67	0	0	0	0	0	0	0	0	0
78	0	0	0	1	0	1	0	0	0
98	0	1	0	0	0	1	0	0	0

FIGURE 3.5: Shows new adjacent of sample data.

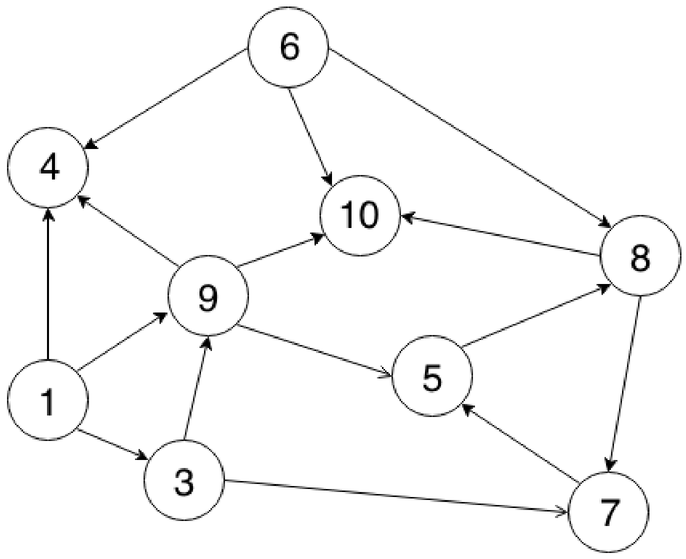


FIGURE 3.6: Shows directed graph of sample data. The nodes represents the members and number inside the nodes represents the kudo rank of the member

After the algorithm applied to our sample data, the third matrix will be generated to display approximated local trust calculated by the algorithm as shown in Figure 3.7 below. The values in red are the apploximated local trust obtained after running algorithm.

Receiver account id

Sender account id

	12	15	34	45	50	54	67	78	98
12	0	0	0	6	**7**	**8**	6	6	0
15	0	0	0	9	9	**8**	9	**5**	0
34	0	1	0	**8**	**8**	**5**	1	**5**	1
45	0	0	0	0	0	0	0	0	0
50	0	0	0	**8**	0	**8**	0	5	0
54	0	0	0	**8**	7	0	0	**5**	0
67	0	0	0	0	0	0	0	0	0
78	0	0	0	8	**7**	8	0	0	0
98	0	3	0	**8**	**8**	3	**9**	**5**	0
MLT	0	2	0	7.8	7.6	6.7	6.25	5.17	1
Kudo Rank	6	9	1	10	5	7	4	8	3

FIGURE 3.7: Shows new adjacent matrix with approximated local trust. The shaded cells are the ones generated after running the algorithm

To apply the developed algorithm to this study data, firstly the data was filtered to selecting members at least get evaluated 10 times. Data filtering was done because more than 70% of members data received only one kudo and their mean approximated local trust will be directly affected by the evaluator kudo rank. Another reason is size of the network. This study faces Java Virtual Machine memory errors when tried to apply big network data to the algorithm. So, filtering and reduce size of the network helps to overcome those challenges. Finally, the results of adjacent matrix of both weighted evaluation network and trust network with estimated local trust were stored in CVS files.

3.4.4 RQ4: To what extent can evaluation network approximate trust information in the open source software community?

3.4.4.1 Data Collection

To address this research goal, the study collects the kudo history data and contributors data and project data as explained in section 3.4.2 above.

3.4.4.2 Data Analysis

The study uses approximated local trust to studying to what extent evaluation network can approximate trust information in the open source software community. At first, the study find the mean estimated local trust of each node. then comparison between kudo rank of the node and mean of estimated local trust will be done for each node.

3.5 Data refinement process

The first version of data collected was refined to remove some of the data with missing members information. Those data removed are kudo transactions to un registered member which their receiver_account_id and receiver_account_name fields were represented as null and it will be difficult to analyze their contribution or kudo ranking since they miss members information.

Chapter 4

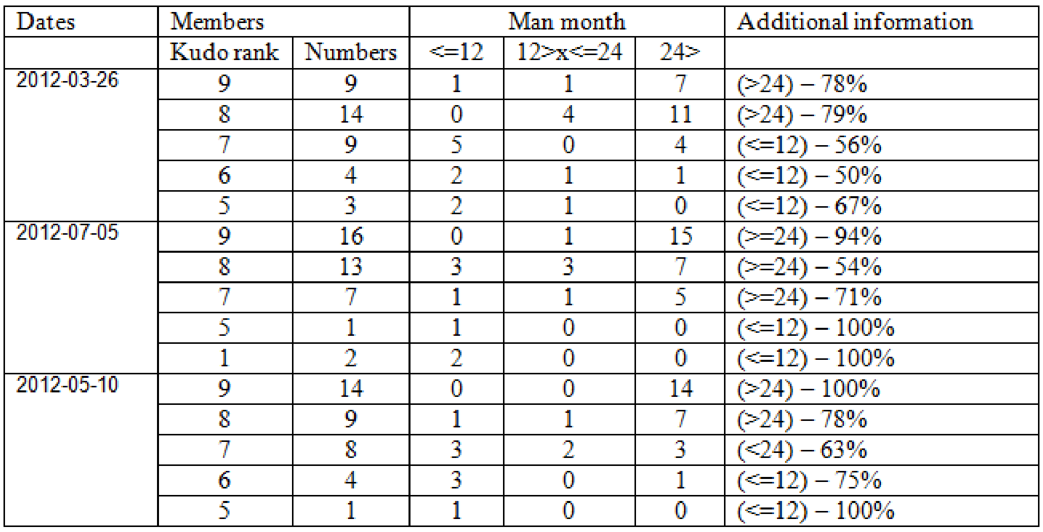
Result Analysis

4.1 Results Analysis

In this section the results of data analysis discussed in previous section are presented with the aim to answer research goals mentioned in Chapter 1 above.

RQ1: How likely a developer will become trusted in the community based on his or her contributions within the community?

To answer this question the assumption was made that at the first committing date all the members were having kudo rank 1 and this will evolve to any kudo rank according to the member contribution. The Figure 4.1 below was the summary of members contribution in three different first commit dates from this study database which are 2012-03-26, 2012-07-05 and 2012-05-10. The results shows that members having kudo rank 9 have contributed in different project by more than 24 man month from 78% to 100%. Members having kudo rank 8 have contributed more than 24 man month from 54% to 79% and kudo rank 7 from 56% to 71%. Moreover, members with kudo rank 5 have contributed to projects by less than 12 man month by 50% to 100% and those having kudo rank 1 they contributed to projects with less than 12 man month by 100%. So, the results shows that, trust values of given members are correlated to the amount of members contributions in the OSS community.

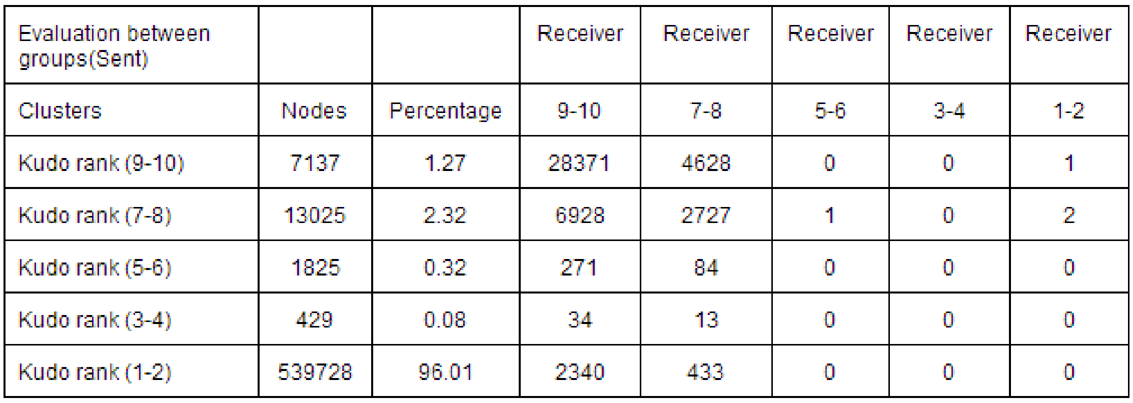


Dates	Members		Man month			Additional information
	Kudo rank	Numbers	<=12	12>x<=24	24>	
2012-03-26	9	9	1	1	7	(>24) – 78%
	8	14	0	4	11	(>24) – 79%
	7	9	5	0	4	(<=12) – 56%
	6	4	2	1	1	(<=12) – 50%
	5	3	2	1	0	(<=12) – 67%
2012-07-05	9	16	0	1	15	(>=24) – 94%
	8	13	3	3	7	(>=24) – 54%
	7	7	1	1	5	(>=24) – 71%
	5	1	1	0	0	(<=12) – 100%
	1	2	2	0	0	(<=12) – 100%
2012-05-10	9	14	0	0	14	(>24) – 100%
	8	9	1	1	7	(>24) – 78%
	7	8	3	2	3	(<24) – 63%
	6	4	3	0	1	(<=12) – 75%
	5	1	1	0	0	(<=12) – 100%

FIGURE 4.1: Shows summary of members contributribution based on man month criteria.

RQ2: How likely that a developer will evaluate other developer of different trust value?

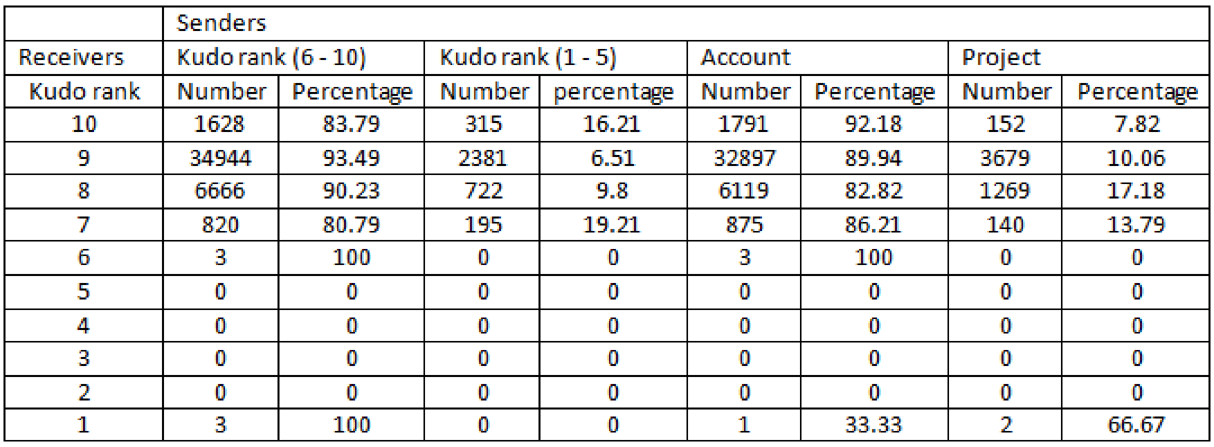
This study continues to analyzing the possibility of evaluation between members having different community status (Kudo Rank). The Figure 4.2 below shows that, there are more evaluation from community members having low kudo rank to members having high kudo rank.



Evaluation between groups(Sent)			Receiver	Receiver	Receiver	Receiver	Receiver
Clusters	Nodes	Percentage	9-10	7-8	5-6	3-4	1-2
Kudo rank (9-10)	7137	1.27	28371	4628	0	0	1
Kudo rank (7-8)	13025	2.32	6928	2727	1	0	2
Kudo rank (5-6)	1825	0.32	271	84	0	0	0
Kudo rank (3-4)	429	0.08	34	13	0	0	0
Kudo rank (1-2)	539728	96.01	2340	433	0	0	0

FIGURE 4.2: Shows clusters according to kudo rank of members and the evaluation between those clusters.

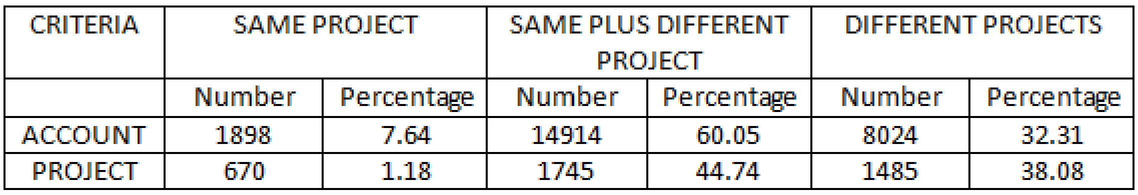
Additionally, this study point out two means of evaluation between members as explained in details in Appendix A and the summary of the results show that there were more evaluation through user personal accounts which is about 72.32% of total evaluation recorded in this study and 26.68% was sent to members as project contributors. Moreover, the results shows that the members having low kudo rank receives more kudo from their contribution than in members personal profiles as presented in Figure 4.3 below.



	Senders							
Receivers	Kudo rank (6 - 10)		Kudo rank (1 - 5)		Account		Project	
Kudo rank	Number	Percentage	Number	percentage	Number	Percentage	Number	Percentage
10	1628	83.79	315	16.21	1791	92.18	152	7.82
9	34944	93.49	2381	6.51	32897	89.94	3679	10.06
8	6666	90.23	722	9.8	6119	82.82	1269	17.18
7	820	80.79	195	19.21	875	86.21	140	13.79
6	3	100	0	0	3	100	0	0
5	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
1	3	100	0	0	1	33.33	2	66.67

Figure 4.3: Shows clusters kudo distribution based on receiving as account holder or to specific project as contributor.

Additionally, in analyzing the effect of the evaluation between members, they were either contributing in the same or different projects. The results show that, more evaluation occurs between members who works in the same and different projects at the same time which is about 60% of the kudo sent to members account. The next group is those that are contributing to completely different projects which is about 32% and the ones who contribute in the same project are 8%. On the other hand, in case of the kudo sent to members as contributors of specific project. The results shows that kudo sent to members contributing in the different and same project at the same time are 45% which are closely to those who works at different projects which are 38% and finally those who works in the same project are 17% as illustrated in Figure 4.4 and 4.5 below.



CRITERIA	SAME PROJECT		SAME PLUS DIFFERENT PROJECT		DIFFERENT PROJECTS	
	Number	Percentage	Number	Percentage	Number	Percentage
ACCOUNT	1898	7.64	14914	60.05	8024	32.31
PROJECT	670	1.18	1745	44.74	1485	38.08

Figure 4.4: Shows kudo history distribution sent to members working in the same projects or in different projects.

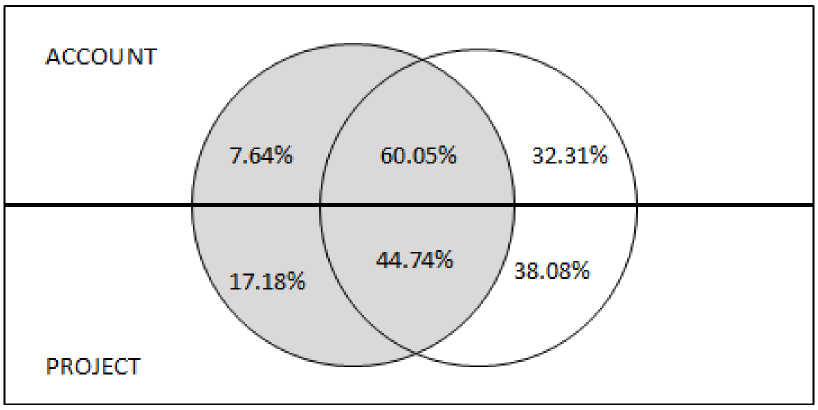
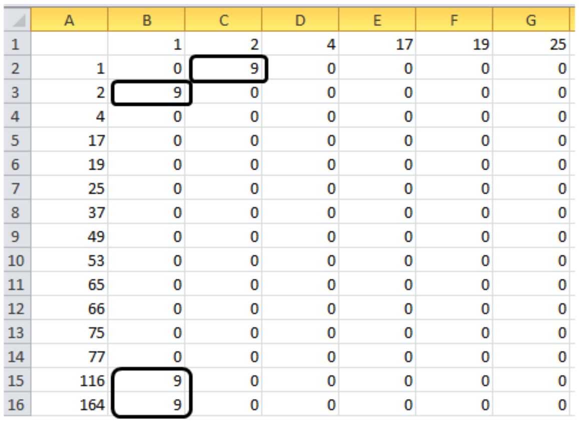


FIGURE 4.5: Shows kudo history distribution sent to members working in the same projects or in different projects.

RQ3: How to identify trust network from evaluation network in the open source software community.

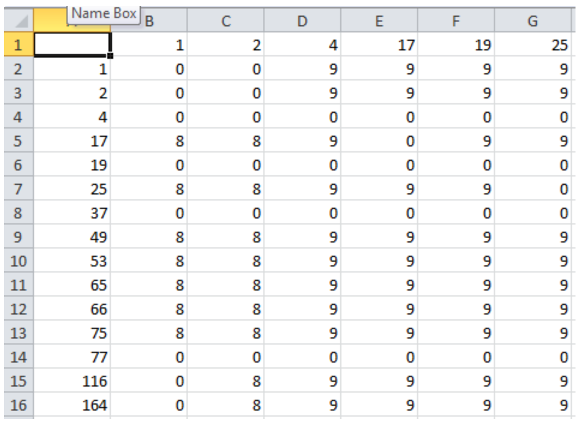
This study evaluation network composes of 15,664 nodes and 46,947 edges. To get trust network out of evaluation network, algorithm was developed to approximate local trust between members by using evaluation between members and their kudo rank as explained in section 3.4. This study was able to get approximated local trust values between members by using the algorithm explained in section 3.4. A partial snapshot of adjacent matrix of weighted evaluation network and results of approximated local matrix are shown in the following Figure 4.6 and Figure 4.7 respectively. In the Figure 4.6 the first row and column represents the member account id and the value on the adjacent row-column values represent kudo rank of a sender as weighted evaluation network scenario explained in section 3.4.3.3 above. Moreover, in Figure 4.7 the first row and column represents the member account id and the adjacent row-column values represents the approximated kudo rank as explained in section 3.4.4.2 above. The full adjacent matrix of the selected data which have 1340 rows and column can be found at the following link [1] and for the approximated local trust matrix of selected data which have 1340 rows and column at the following link [2].

[1]https://drive.google.com/file/d/0B7yISd0ndVt4NEJtUzJkdzZ6dnM/edit?usp=sharing
[2]https://drive.google.com/file/d/0B7yISd0ndVt4aGlIY1haQnNrOUO/edit?usp=sharing



	A	B	C	D	E	F	G
1		1	2	4	17	19	25
2	1	0	9	0	0	0	0
3	2	9	0	0	0	0	0
4	4	0	0	0	0	0	0
5	17	0	0	0	0	0	0
6	19	0	0	0	0	0	0
7	25	0	0	0	0	0	0
8	37	0	0	0	0	0	0
9	49	0	0	0	0	0	0
10	53	0	0	0	0	0	0
11	65	0	0	0	0	0	0
12	66	0	0	0	0	0	0
13	75	0	0	0	0	0	0
14	77	0	0	0	0	0	0
15	116	9	0	0	0	0	0
16	164	9	0	0	0	0	0

FIGURE 4.6: Shows partial snapshot of the adjacent matrix of the selected study data. The full matrix have 1340 rows and 1340 column.

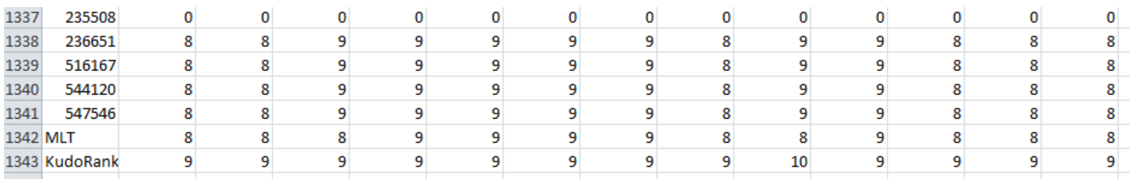


	A	B	C	D	E	F	G
1		1	2	4	17	19	25
2	1	0	0	9	9	9	9
3	2	0	0	9	9	9	9
4	4	0	0	0	0	0	0
5	17	8	8	9	0	9	9
6	19	0	0	0	0	0	0
7	25	8	8	9	9	9	0
8	37	0	0	0	0	0	0
9	49	8	8	9	9	9	9
10	53	8	8	9	9	9	9
11	65	8	8	9	9	9	9
12	66	8	8	9	9	9	9
13	75	8	8	9	9	9	9
14	77	0	0	0	0	0	0
15	116	0	8	9	9	9	9
16	164	0	8	9	9	9	9

FIGURE 4.7: Shows partial snapshot of the adjacent matrix with estimated local trust.The full matrix have 1340 rows and 1340 column.

RQ4: To what extent can evaluation network approximate trust information in the open source software community?

This study continue to analyses the possibility of using evaluation network to approximate trust information in the open source software community. This was done by using approximated local trust generated by the algorithm developed in previous section. Then, the mean approximated local trust of each member (node in the network) was calculated and the value was compared to the kudo rank of the given member as shown in snapshot of the approximated local trust illustrated in Figure 4.8 below. The results of this approach show that the value of approximated local trust was directly affected by the kudo rank of members who evaluate the given member. Furthermore, the stability of this value depends on evaluators kudo rank. For example, the number will stay high if most of evaluators have high kudo rank and will go low if most of evaluators have low kudo rank.



1337	235508	0	0	0	0	0	0	0	0	0	0	0	0	0
1338	236651	8	8	9	9	9	9	9	8	9	9	8	8	8
1339	516167	8	8	9	9	9	9	9	8	9	9	8	8	8
1340	544120	8	8	9	9	9	9	9	8	9	9	8	8	8
1341	547546	8	8	9	9	9	9	9	8	9	9	8	8	8
1342	MLT	8	8	8	9	9	9	9	8	8	9	8	8	8
1343	KudoRank	9	9	9	9	9	9	9	9	10	9	9	9	9

Figure 4.8: Shows partial snapshot of mean approximated local trust and kudo rank of every member among selcted data.

The results after comparison between kudo rank and mean approximated local trust are presented in the following Table 4.1.

Table 4.1: Evaluation betwen developers

Difference	-8	-7	-4	-3	-2	-1	0	1	2
Numbers	9	1	1	2	6	58	981	272	10
Percentage	0.6	0.07	0.07	0.14	0.42	4	68.6	19	0.7

4.2 Threats to Validity

This section identifies threats that may affect validity of this study.

4.2.1 Construct Validity

Construct validity threat is the extent which the studied operational measures reflects what the researcher intended to study according to research goals [26]. In this study construct validity can be assumptions made during conducting this study. Firstly, is the construction of evaluation network by considering binary evaluations weather 1 for evaluation between members and 0 for no evaluation between developers. Secondly, is considering kudo rank of the given member as the global trust value of the given member. This is because kudo rank has characteristics of global trust values as explained in section 2.4 above. Finally, is the using the kudo rank of evaluator as the local trust between evaluator and the one who is evaluated. To minimise this threat to the validity, this study was conducted based of different previous related researches works done in other web based social networks.

4.2.2 Internal Validity

Internal validity threat is the aspects where external factors may affect the study results and researcher can be aware with some of these factors and others may not be aware of them [26]. Openhub pulls data directly from version control system like Git and SVN. So, some of developers are not fully registered with their full details in open hub data repository. Additionally, some of the contributors registered with different names in different projects they contributes and some contributors have not updated their personal information. To minimize this threat to the validity, this study omitted some of contributors data missing contributors personal information like kudo rank which is one of the basic data of this study. However, these data omitted 100 %of them were kudo sent to contributors of specific project. So, may affect research goal number 2.

4.2.3 External Validity

External validity threats reflects to what extent the results of this study can be generalizable [26]. Generalizability of this study results is one of the validity threats of this study. This is because this study used single data source which is openhub data repository. To minimize this threat to validity, this study collects large volume of data and analysis of data was done in different alternative so as to improve the study results.

4.2.4 Reliability

Reliability is the aspect concerning with how the study data and data analysis are dependent to the researcher. This means that, how the results will be if the same study will be conducted by other researcher? [26]. This study results may varies depending on time because process of evaluation between contributors is the continuous process and may change over time. Another reason is assumption and modeling of trust made by researcher.

Chapter 5

Discussion

5.1 Discussion

This thesis focuses on the studying of trust in OSS communities. However, there are other researches were studied trust in different social networks. In this study, one of the research goal was to explore; How a community member will be trusted based on his or her contribution within OSS community? This study used man month values as a metric to measure members contribution. This was similarly to the team effort metric used in this study [2] which uses number of weeks a contributor devoted to the project. So one of their findings was affective trust is directly support team effort. On other hand this study found that the more contributors efforts to technically contribute to OSS projects, the more he or she is becoming trusted in the community. This finding is relating to one of the challenges OSS community faced and other web based social network, which is how to trust other members in the community based on their contribution. This challenge was also found in this study [4] where a member trust inference can be strong or weak based on his or her role and contributions.

Evaluation between members in OSS communities is another point of interest in this study. Firstly, clusters between members of different community status were formed and evaluation between members belonging in these clusters was studied. The results shows the richer get richer phenomenon. This is the same as one of the results observed in this study [13] where accumulation factor is influencing a community member to evaluate others who already have been evaluated most. Then, this thesis studied the kudo sent to members in their personal accounts and to their accounts as contributors. The results shows that, the more evaluations were sent to members personal accounts than in their contributors accounts. Moreover, the evaluation is happening more between members contributing at the same time in the same and different projects. So, this thesis can conclude that evaluation between members is not influenced by weather they are contributing to the same projects. On the other hand, homophily factors like same community status, same programming language and same location influencing evaluation between members [13].

Evaluation between developers will form an evaluation network which nodes are members and link between them is evaluation from one member to another [12]. Additionally, this OSS community used in this study shows a small world and scale free network properties [12]. However, the target of this study was to extract a trust network from evaluation network. To achieve that goal, this study develope an algorithm which uses evaluation network and community status values which in this case was kudo rank to approximate local trust values between members within the community. The algorithm uses existing network connection to infer trust between members who are not directly connected as did in these studies [7] [8]. Additionally, the study uses mean estimated local trust to the possibility of using evaluation network to extract trust information within the community. And the results showed that the approach was successfully able to extract trust information.

Chapter 6

Conclusion and Future Work

6.1 Summary

Trust has been one of the important issues to be considered in OSS development communities. For example interpersonal trust is important for team development effectiveness. This study leads to development of algorithm which uses evaluation network and kudo rank of members to approximate local trust. Then it uses mean estimated local trust to estimate trust information within open source community. Clustering of members group based on kudo rank was made and evaluation between groups was studied. Finally is the evolution of kudo rank of members with time based on their contributions.

6.2 Conclusion

Based on result analysis results the following conclusions were made by this study.

RQ1: How likely a developer will become trusted in the community based on his or her contributions within the community?

This study groups members based on their first commit time (date) as an assumption in that date all the members have kudo rank 1. However, the kudo rank of members will vary against time based on members contribution. The results shows that members having high kudo rank tends to put more effort on developing OSS and this made them to become trusted in the community. On the other hand, community members with low kudo rank put less effort on OSS development thats why they are less trusted.

RQ2: How likely that a developer will evaluate other developer of different trust value?

Firstly, this study formulate cluster of members based on kudo rank and analyse the evaluation between those clusters. The results show that the richer get richer phenomena where the members with high kudo rank are evaluated more. Additionally, this study goes deeper to analyse the kudo history sent to a members account and to members as contributor. The results shows that the members evaluation is based more by having more knowledge about personal information of a member he or she evaluated than technical contributions. Then the studies continue to check effect similarity of members contributing in the projects. And, the results shows that for the kudo sent to user accounts, more evaluations will happened between members working weather in the same project or in different projects. Moreover, in the case of kudo sent to the contributors nearly most of the kudo was sent among members working either in the same project and different project and completely contributing in different project. How ever, in both cases it shows there is less number of kudo sent between members contributing to the same project only.

RQ3: How to identify trust network from evaluation network in the open source software community?

The algorithm developed manages to approximate local trust between members by using evaluation network between them and their kudo rank values. So out of those approximated local trusts between members we can generate trust network between them.

RQ4: To what extent can evaluation network approximate trust information in the open source software community? This study uses approximated local trust between members developed by the algorithm to calculate mean local trust value for each individual member s. Then the mean local trust value was compared with kudo rank value of that given member. The results show that the value of mean local trust is affected directly with the kudo rank of the members evaluating this given member. For example if the member was evaluated more by the member having higher kudo rank which are considered to be trusted in this study. However, the evaluated member will have high kudo rank and be trusted as well and vice versa is true in case you are evaluated by members having low kudo rank. Also the study result shows that about 69% of the mean approximated local trust was the same as the kudo rank of the given member. Moreover, the following difference of weather 1 or -1 are about 23%. SO, this study concluded that evaluation network can approximate the trust information within a given social network.

6.3 Future Work

This study proposes the future possible researches can be done in the case of tracking the members who decided to take their kudo back. Currently Ohloh does not track this process in their data repository as shown in Appendix A.3. So currently require manual tracking but is one of the possible future study. Another possible study is to analyse the action taken within a community in case of the code submission or feature suggestion between members of the different kudo rank. For example, how long it takes if member A with kudo rank 10 commits changes to be integrated in the main software code base. On other hand member B with kudo rank 1 commit changes, how long it will take for the commit to be integrated with main software code base. Moreover, this can be done in other forms of contributions as mentioned in chapter 3 above such as forum posting, maintenance of the project wiki, software documentation and participating on mailing list and instant chat application conversations.
Furthermore, is to add more reference data to the research goal number 4 and proposing standardization of the metric used in this study man month. Some previous studies shows a standard size of the project can be determined using LoC [27]. Additionally, the other studies propose the standard size of commit based on the LoC committed by a developer [11]. So I think it will be a good study to try to standardize the metric like man month as shown in Ohloh data repository. Because this study uses less or equal to 12 man month with the assumption that may be a contributor is the student at a certain university uses open source software as a way learning and may be after finishing the studies will no longer involve that of software development. On the other hand uses greater than 24 man month as assumption that a member who will contribute for that

amount of time he or she must be well committed in contributing in OSS communities. Another proposed future research is to examine the effect of difference between mean approximated local trust obtained after using algorithm and kudo rank of the member against given time. With the assumption of having positive value means the member will be promoted to higher kudo rank and negative values means the member will remain or is kudo rank will be lowered in the near future. This is because the algorithm results shows that the evaluators kudo rank may affect the value of mean approximated local trust.

Appendix A

Process of sending kudo to other member

Ohloh members can send a kudo to another ohloh members. And, this process of sending kudo can be done in two different ways. Firstly, is to a directly member account and secondly to a member who contributing to a specific project.

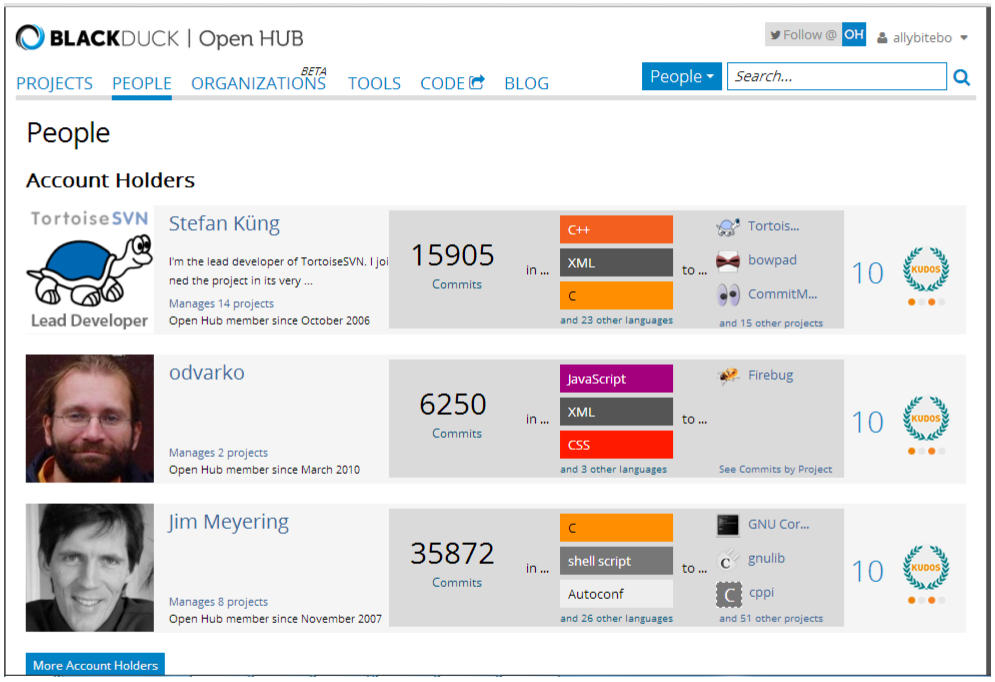
A.1 The following screen captures shows how a ohloh member can send a kudo to another member

.

Firstly a member login to the site. Then, a user can select people menu to browse list of all members in the ohloh dataset as shown in the following Figure A.1 below.

Then a member can search a specific user by using search field or browser for more account by clicking more account holder button. For this demonstration user Stefan Kng was selected and the following page open for specific member. The page shows account summary, badges the user acquired and different activities performed by a specific user as shown in the Figure A.2 below.

A member can decide to give a kudo to Stefan Kng by clicking a Give Kudo beside the member profile picture. The window as shown in Figure A.3 will appear and the member can add a message associated to the kudo he or she sent to Stefan Kng.

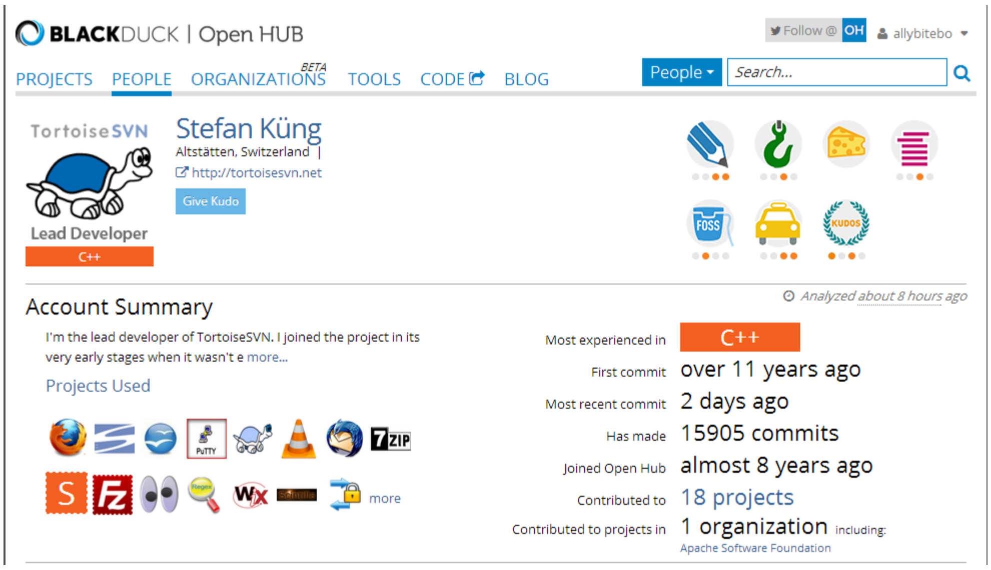


People

Account Holders

TortoiseSVN Lead Developer	Stefan Küng I'm the lead developer of TortoiseSVN. I joined the project in its very ... Manages 14 projects Open Hub member since October 2006	15905 Commits	in ... C++, XML, C and 23 other languages	to ... Tortois..., bowpad, CommitM... and 15 other projects	10	KUDOS
	odvarko Manages 2 projects Open Hub member since March 2010	6250 Commits	in ... JavaScript, XML, CSS and 3 other languages	to ... Firebug See Commits by Project	10	KUDOS
	Jim Meyering Manages 8 projects Open Hub member since November 2007	35872 Commits	in ... C, shell script, Autoconf and 26 other languages	to ... GNU Cor..., gnulib, cppi and 51 other projects	10	KUDOS

FIGURE A.1: Shows members lists.



Stefan Küng
Altstätten, Switzerland |
http://tortoisesvn.net

Account Summary

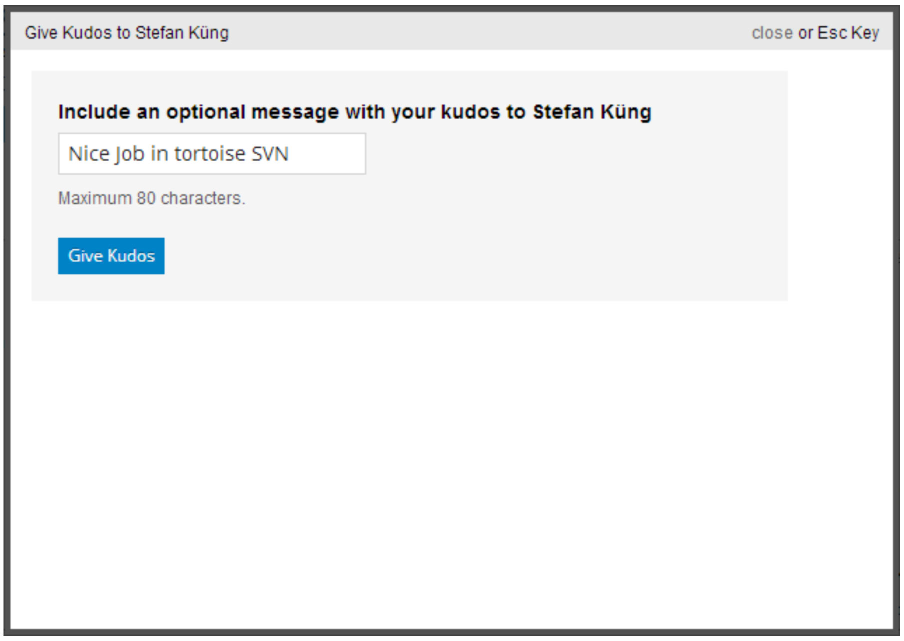
Analyzed about 8 hours ago

I'm the lead developer of TortoiseSVN. I joined the project in its very early stages when it wasn't e more...

Projects Used

Most experienced in	C++
First commit	over 11 years ago
Most recent commit	2 days ago
Has made	15905 commits
Joined Open Hub	almost 8 years ago
Contributed to	18 projects
Contributed to projects in	1 organization including: Apache Software Foundation

FIGURE A.2: Shows member's pages.



Give Kudos to Stefan Küng

close or Esc Key

Include an optional message with your kudos to Stefan Küng

Nice Job in tortoise SVN

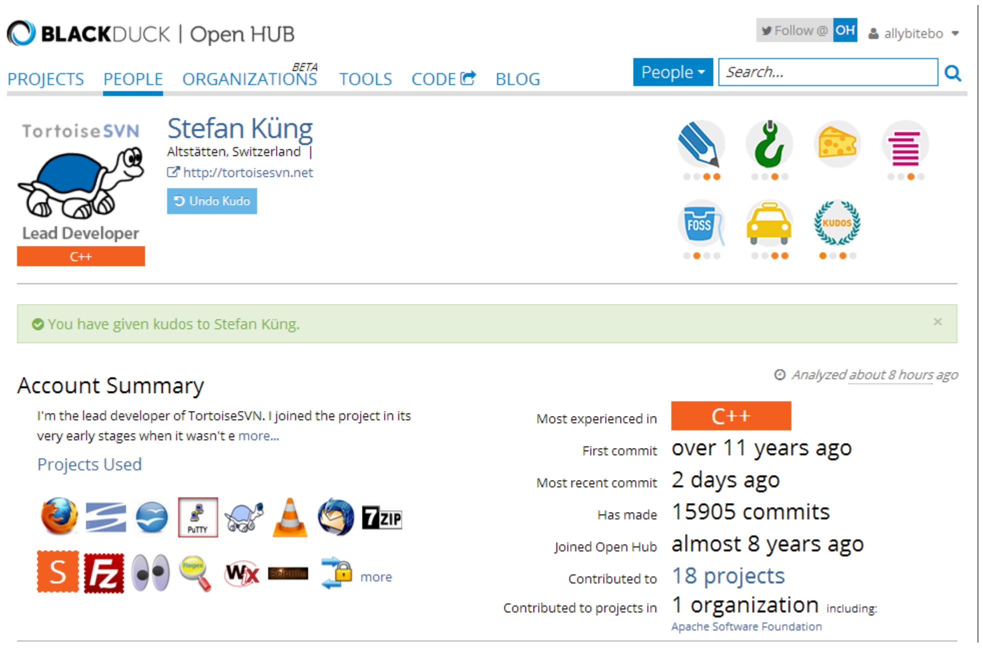
Maximum 80 characters.

Give Kudos

Figure A.3: Shows a pages where a member can write a message to attach to the kudo he or she want to send.

Then a user can click Give Kudo button to send a kudo a user or he or she can cancel the process by clicking close or use escape key on keyboard.

When a Give kudo button pressed the process of sending kudo to a member will be successful and the following page will be displayed to confirm the process.



Stefan Küng
Altstätten, Switzerland
http://tortoisesvn.net

You have given kudos to Stefan Küng.

Analyzed about 8 hours ago

Account Summary

I'm the lead developer of TortoiseSVN. I joined the project in its very early stages when it wasn't e more...

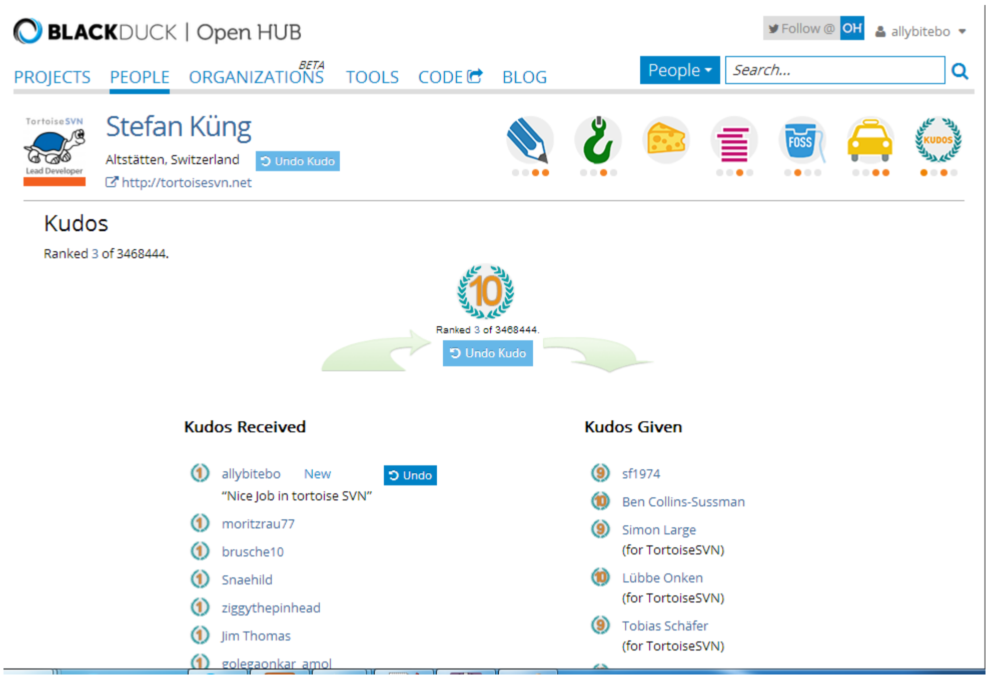
Projects Used

more

Most experienced in	C++
First commit	over 11 years ago
Most recent commit	2 days ago
Has made	15905 commits
Joined Open Hub	almost 8 years ago
Contributed to	18 projects
Contributed to projects in	1 organization including: Apache Software Foundation

Figure A.4: Shows a confirmation page to a user sent a kudo.

And, the sent kudo can be observed in members page as shown in Figure 3.5 and kudo history of the specific user during API call as shown in Figure 3.6.



Stefan Küng

Altstätten, Switzerland Undo Kudo

http://tortoisesvn.net

Kudos

Ranked 3 of 3468444.

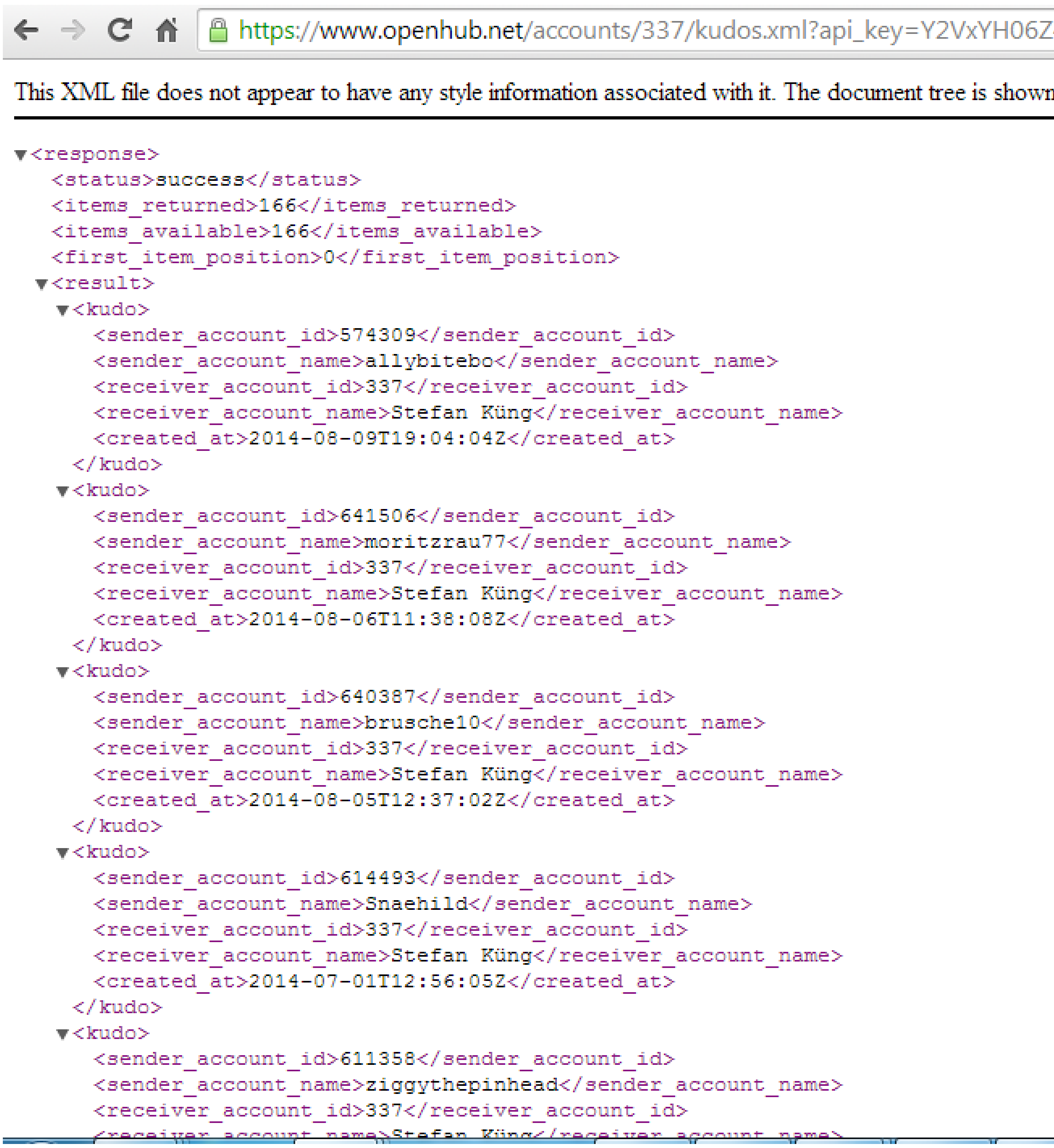
10

Ranked 3 of 3468444.

Undo Kudo

Kudos Received	Kudos Given
allybitebo New Undo "Nice Job in tortoise SVN"	sf1974
moritzrau77	Ben Collins-Sussman
brusche10	Simon Large (for TortoiseSVN)
Snaehild	Lübbe Onken (for TortoiseSVN)
ziggythepinhead	Tobias Schäfer (for TortoiseSVN)
Jim Thomas	
golegaonkar amol	

Figure A.5: Shows a member's page with number of kudo he received.


https://www.openhub.net/accounts/337/kudos.xml?api_key=Y2VxYH06Z

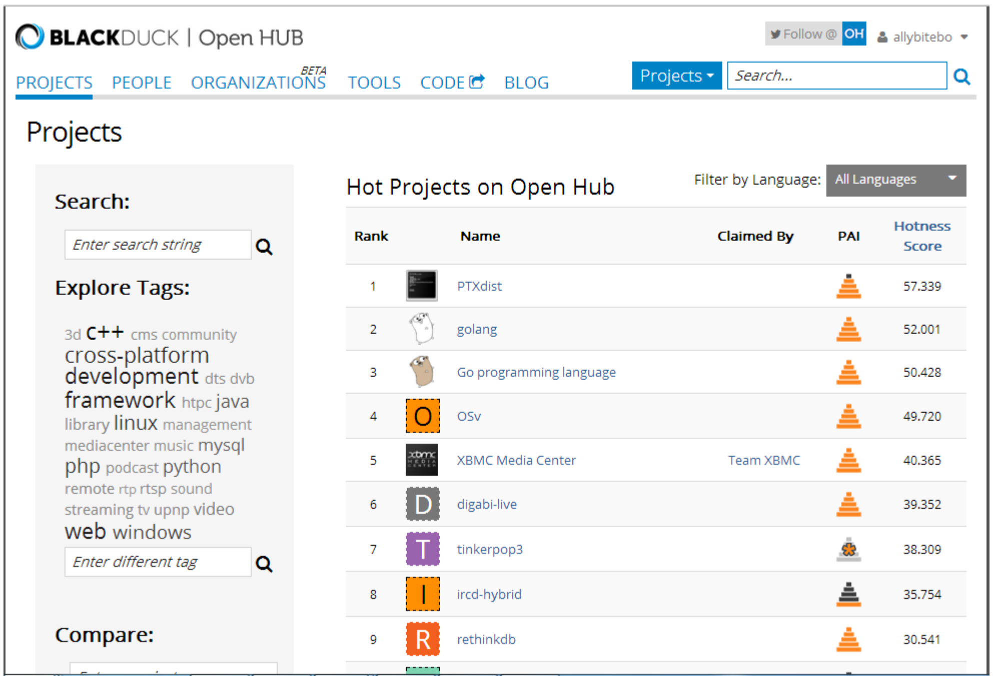
This XML file does not appear to have any style information associated with it. The document tree is shown

```
▼<response>
   <status>success</status>
   <items_returned>166</items_returned>
   <items_available>166</items_available>
   <first_item_position>0</first_item_position>
 ▼<result>
   ▼<kudo>
      <sender_account_id>574309</sender_account_id>
      <sender_account_name>allybitebo</sender_account_name>
      <receiver_account_id>337</receiver_account_id>
      <receiver_account_name>Stefan Küng</receiver_account_name>
      <created_at>2014-08-09T19:04:04Z</created_at>
    </kudo>
   ▼<kudo>
      <sender_account_id>641506</sender_account_id>
      <sender_account_name>moritzrau77</sender_account_name>
      <receiver_account_id>337</receiver_account_id>
      <receiver_account_name>Stefan Küng</receiver_account_name>
      <created_at>2014-08-06T11:38:08Z</created_at>
    </kudo>
   ▼<kudo>
      <sender_account_id>640387</sender_account_id>
      <sender_account_name>brusche10</sender_account_name>
      <receiver_account_id>337</receiver_account_id>
      <receiver_account_name>Stefan Küng</receiver_account_name>
      <created_at>2014-08-05T12:37:02Z</created_at>
    </kudo>
   ▼<kudo>
      <sender_account_id>614493</sender_account_id>
      <sender_account_name>Snaehild</sender_account_name>
      <receiver_account_id>337</receiver_account_id>
      <receiver_account_name>Stefan Küng</receiver_account_name>
      <created_at>2014-07-01T12:56:05Z</created_at>
    </kudo>
   ▼<kudo>
      <sender_account_id>611358</sender_account_id>
      <sender_account_name>ziggythepinhead</sender_account_name>
      <receiver_account_id>337</receiver_account_id>
      <receiver_account_name>Stefan Küng</receiver_account_name>
```

Figure A.6: Shows a xml file returned with the list of kudo a member receives.

A.2 The following screen captures show how a member can send a kudo to a specific project contributor.

Firstly a member must login and after successfully login a user can browse project list as shown in Figure A.7 or search a specific project as shown in Figure A.8 below.For the demonstration purpose this study search git project and result is shown in Figure A.8 below.


BLACKDUCK | Open HUB
Follow @ OH
allybitebo
PROJECTS
PEOPLE
ORGANIZATIONS BETA
TOOLS
CODE
BLOG
Projects
Search...
Search:
Enter search string
Explore Tags:
3d C++ cms community cross-platform development dts dvb framework htpc java library linux management mediacenter music mysql php podcast python remote rtp rtsp sound streaming tv upnp video web windows
Enter different tag
Compare:
Filter by Language:
All Languages

Projects

Hot Projects on Open Hub

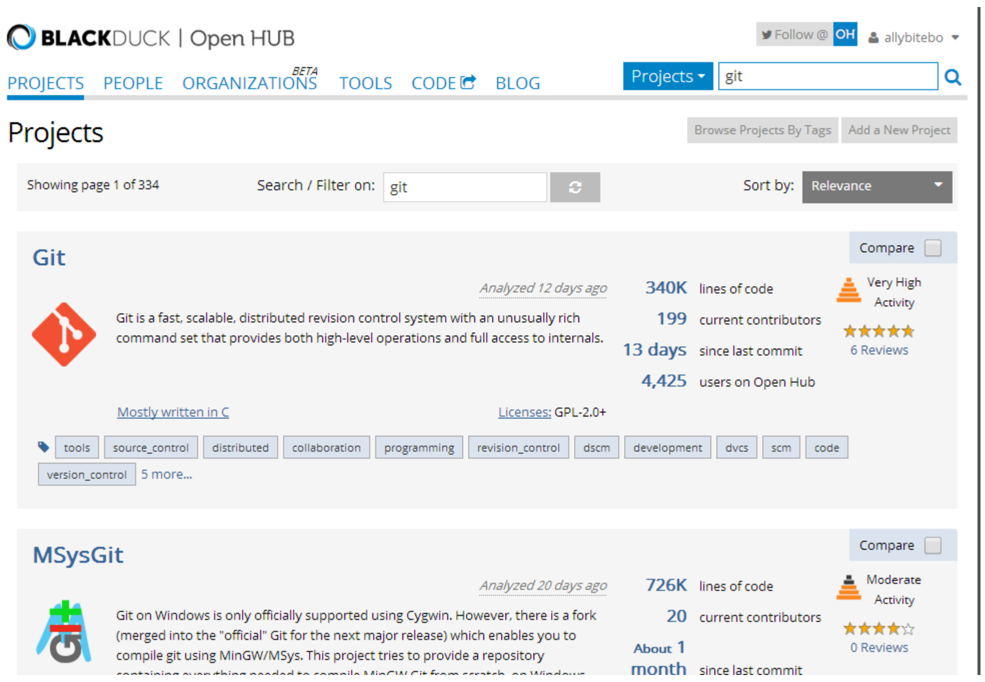
Rank	Name	Claimed By	PAI	Hotness Score
1	PTXdist			57.339
2	golang			52.001
3	Go programming language			50.428
4	OSv			49.720
5	XBMC Media Center	Team XBMC		40.365
6	digabi-live			39.352
7	tinkerpop3			38.309
8	ircd-hybrid			35.754
9	rethinkdb			30.541

Figure A.7: Shows project lists results when a user click browse project link.

Then from the results, the git project was selected so as to observe different project activities and contributors of that specific project as shown in Figure A.9.

A member can browse the list of contributors who they contribute in git project or can use search field to search for a specific contributor. For the demonstration purpose this study search a specific user with the name linus and the search results appear as shown in Figure A.10.

By browsing to a specific contributor page as shown in Figure A.11, a member can decide to give a kudo to Linus Torvalds by clicking the Give Kudo button and fill the message if a member want to associate with the kudo he or she sent as shown in Figure



Projects

Showing page 1 of 334 Search / Filter on: git Sort by: Relevance

Git

Compare

Analyzed 12 days ago

Git is a fast, scalable, distributed revision control system with an unusually rich command set that provides both high-level operations and full access to internals.

340K	lines of code
199	current contributors
13 days	since last commit
4,425	users on Open Hub

Very High Activity

★★★★★ 6 Reviews

Mostly written in C Licenses: GPL-2.0+

tools source_control distributed collaboration programming revision_control dscm development dvcs scm code version_control 5 more...

MSysGit

Compare

Analyzed 20 days ago

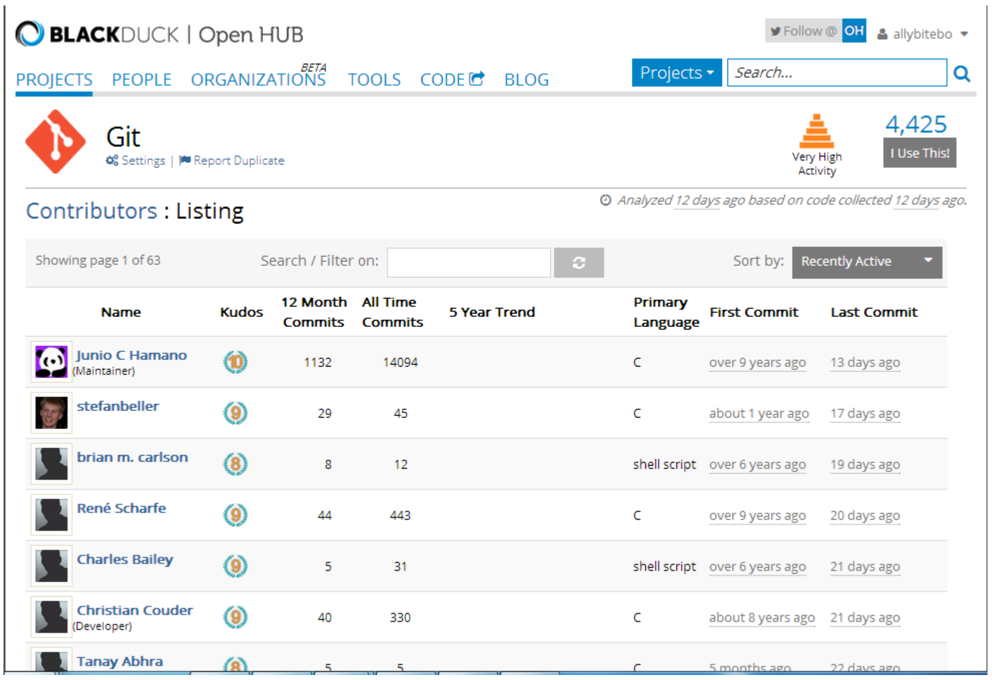
Git on Windows is only officially supported using Cygwin. However, there is a fork (merged into the "official" Git for the next major release) which enables you to compile git using MinGW/MSys. This project tries to provide a repository

726K	lines of code
20	current contributors
About 1 month	since last commit

Moderate Activity

★★★★☆ 0 Reviews

FIGURE A.8: Shows a search results when a user search for a specific project name, this study use "git" as a search word.



Git

Settings | Report Duplicate

Very High Activity 4,425 I Use This!

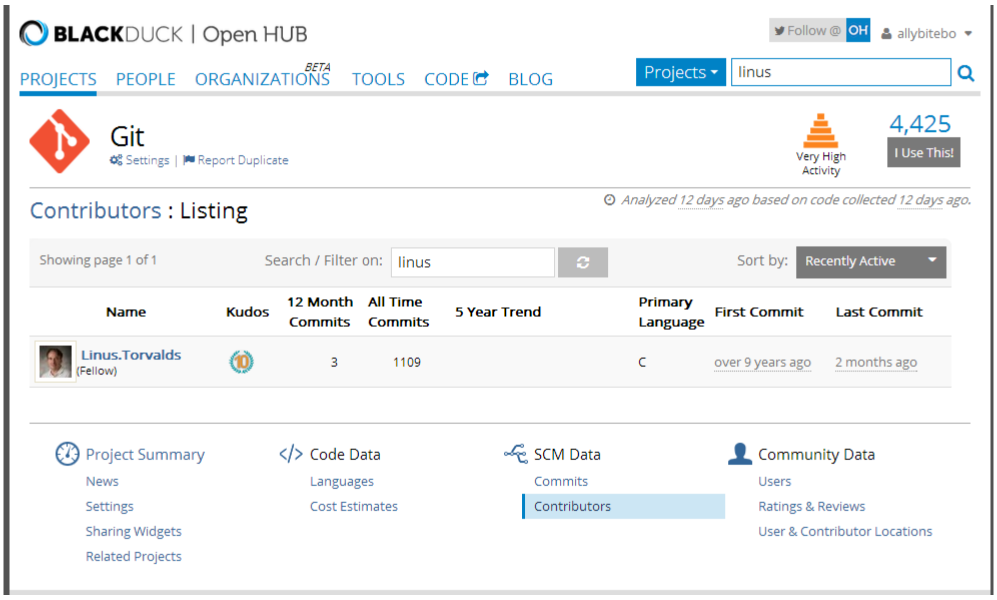
Contributors : Listing

Analyzed 12 days ago based on code collected 12 days ago.

Showing page 1 of 63 Search / Filter on: Sort by: Recently Active

Name	Kudos	12 Month Commits	All Time Commits	5 Year Trend	Primary Language	First Commit	Last Commit
Junio C Hamano (Maintainer)	10	1132	14094		C	over 9 years ago	13 days ago
stefanbeller	9	29	45		C	about 1 year ago	17 days ago
brian m. carlson	8	8	12		shell script	over 6 years ago	19 days ago
René Scharfe	9	44	443		C	over 9 years ago	20 days ago
Charles Bailey	9	5	31		shell script	over 6 years ago	21 days ago
Christian Couder (Developer)	9	40	330		C	about 8 years ago	21 days ago
Tanay Abhra	8	5	5		C	5 months ago	22 days ago

FIGURE A.9: Shows a list of git project contributors.



Analyzed 12 days ago based on code collected 12 days ago.

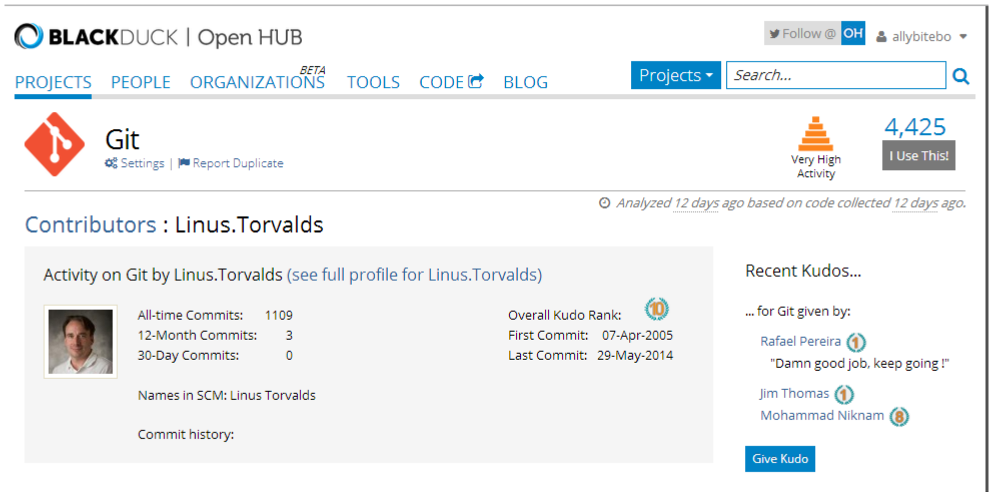
Contributors : Listing

Showing page 1 of 1 Search / Filter on: linus Sort by: Recently Active

Name	Kudos	12 Month Commits	All Time Commits	5 Year Trend	Primary Language	First Commit	Last Commit
Linus.Torvalds (Fellow)	10	3	1109		C	over 9 years ago	2 months ago

FIGURE A.10: Shows a search results when a user search for a specific project contributor name, this study use ”linus” as a search word.

A.3 above. More over the successfully kudo sent message will be displayed as shown in Figure A.12.



Analyzed 12 days ago based on code collected 12 days ago.

Contributors : Linus.Torvalds

Activity on Git by Linus.Torvalds (see full profile for Linus.Torvalds)

All-time Commits: 1109
12-Month Commits: 3
30-Day Commits: 0

Overall Kudo Rank: 10
First Commit: 07-Apr-2005
Last Commit: 29-May-2014

Names in SCM: Linus Torvalds

Commit history:

Recent Kudos...

... for Git given by:

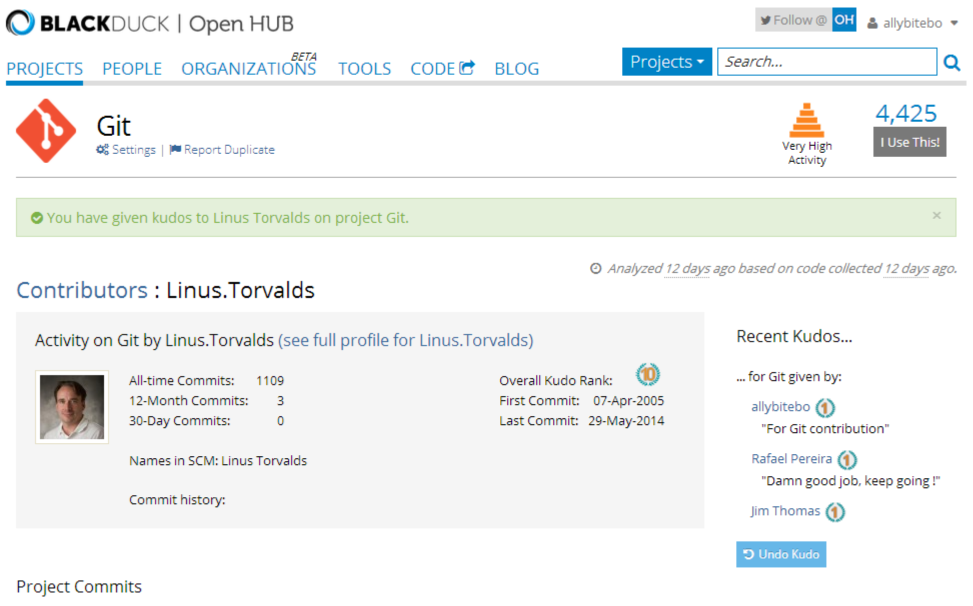
Rafael Pereira 1
"Damn good job, keep going !"

Jim Thomas 1
Mohammad Niknam 8

FIGURE A.11: Shows a project contributor page.

And, this kudo will be recorded as a kudo to a contributor in particular project. The record of this kudo can be observed in specific member kudo history of the given member as shown in Figure A.13 below.

And, these records can be observed also in sender kudo history as shown in Figure A.14 below.



You have given kudos to Linus Torvalds on project Git.

Analyzed 12 days ago based on code collected 12 days ago.

Contributors : Linus.Torvalds

Activity on Git by Linus.Torvalds (see full profile for Linus.Torvalds)

All-time Commits:	1109
12-Month Commits:	3
30-Day Commits:	0
Overall Kudo Rank:	
First Commit:	07-Apr-2005
Last Commit:	29-May-2014

Names in SCM: Linus Torvalds

Commit history:

Recent Kudos...

... for Git given by:

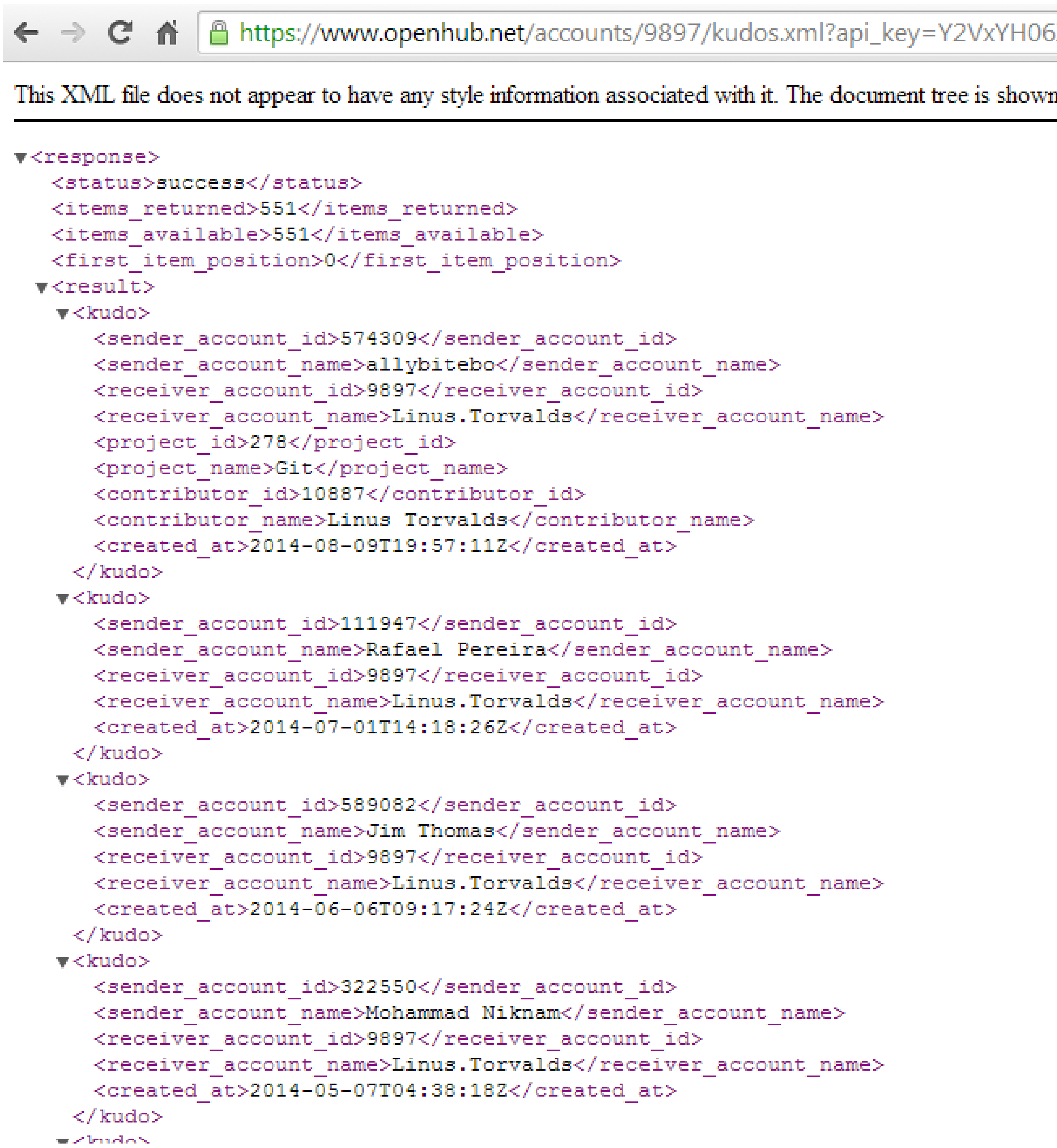
allybitebo
"For Git contribution"

Rafael Pereira
"Damn good job, keep going !"

Jim Thomas

Project Commits

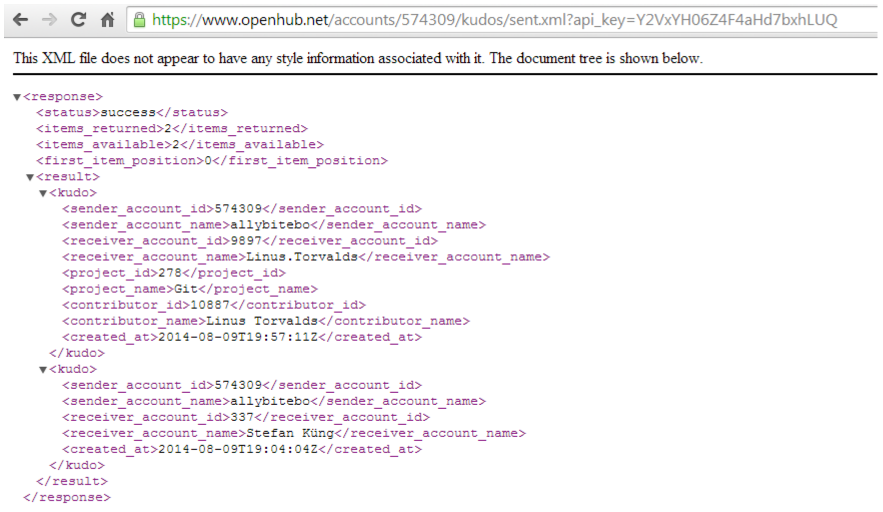
Figure A.12: Shows a kudo sent confirmation page.


https://www.openhub.net/accounts/9897/kudos.xml?api_key=Y2VxYH06

This XML file does not appear to have any style information associated with it. The document tree is shown

```
▼<response>
   <status>success</status>
   <items_returned>551</items_returned>
   <items_available>551</items_available>
   <first_item_position>0</first_item_position>
  ▼<result>
    ▼<kudo>
       <sender_account_id>574309</sender_account_id>
       <sender_account_name>allybitebo</sender_account_name>
       <receiver_account_id>9897</receiver_account_id>
       <receiver_account_name>Linus.Torvalds</receiver_account_name>
       <project_id>278</project_id>
       <project_name>Git</project_name>
       <contributor_id>10887</contributor_id>
       <contributor_name>Linus Torvalds</contributor_name>
       <created_at>2014-08-09T19:57:11Z</created_at>
     </kudo>
    ▼<kudo>
       <sender_account_id>111947</sender_account_id>
       <sender_account_name>Rafael Pereira</sender_account_name>
       <receiver_account_id>9897</receiver_account_id>
       <receiver_account_name>Linus.Torvalds</receiver_account_name>
       <created_at>2014-07-01T14:18:26Z</created_at>
     </kudo>
    ▼<kudo>
       <sender_account_id>589082</sender_account_id>
       <sender_account_name>Jim Thomas</sender_account_name>
       <receiver_account_id>9897</receiver_account_id>
       <receiver_account_name>Linus.Torvalds</receiver_account_name>
       <created_at>2014-06-06T09:17:24Z</created_at>
     </kudo>
    ▼<kudo>
       <sender_account_id>322550</sender_account_id>
       <sender_account_name>Mohammad Niknam</sender_account_name>
       <receiver_account_id>9897</receiver_account_id>
       <receiver_account_name>Linus.Torvalds</receiver_account_name>
       <created_at>2014-05-07T04:38:18Z</created_at>
     </kudo>
    ▼<kudo>
```

Figure A.13: Shows a kudo sent history after API call.



https://www.openhub.net/accounts/574309/kudos/sent.xml?api_key=Y2VxYH06Z4F4aHd7bxhLUQ

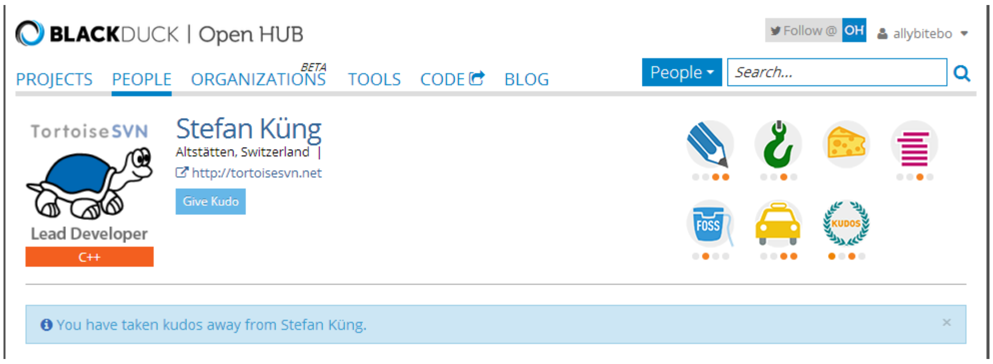
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<response>
   <status>success</status>
   <items_returned>2</items_returned>
   <items_available>2</items_available>
   <first_item_position>0</first_item_position>
 ▼<result>
   ▼<kudo>
      <sender_account_id>574309</sender_account_id>
      <sender_account_name>allybitebo</sender_account_name>
      <receiver_account_id>9897</receiver_account_id>
      <receiver_account_name>Linus.Torvalds</receiver_account_name>
      <project_id>278</project_id>
      <project_name>Git</project_name>
      <contributor_id>10887</contributor_id>
      <contributor_name>Linus Torvalds</contributor_name>
      <created_at>2014-08-09T19:57:11Z</created_at>
    </kudo>
   ▼<kudo>
      <sender_account_id>574309</sender_account_id>
      <sender_account_name>allybitebo</sender_account_name>
      <receiver_account_id>337</receiver_account_id>
      <receiver_account_name>Stefan Küng</receiver_account_name>
      <created_at>2014-08-09T19:04:04Z</created_at>
    </kudo>
  </result>
</response>
```

Figure A.14: Shows a sender xml file retrurned after API call.

A.3 The following screen captures show how a member can take back kudo he or she sent before

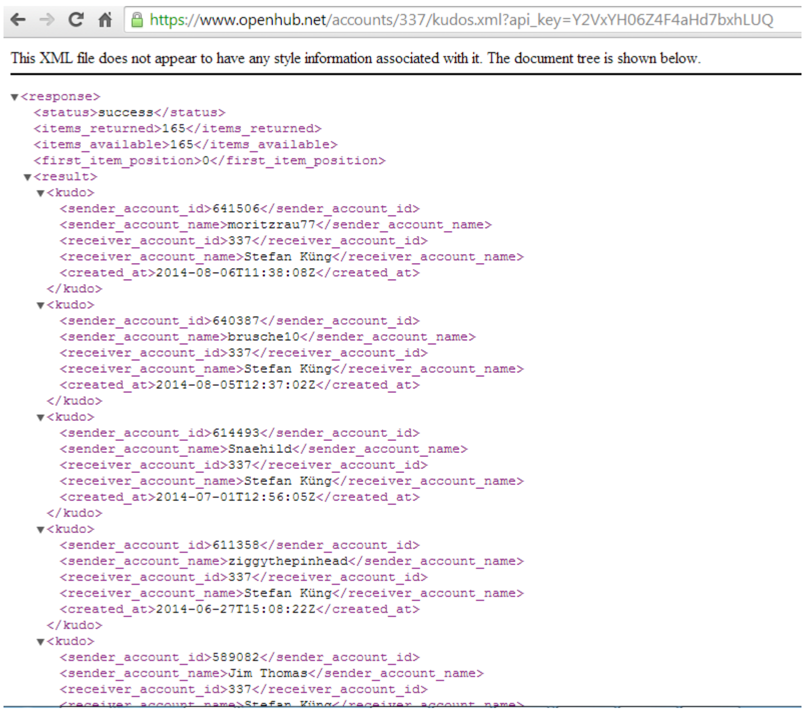
A member can decide to take the kudo he or she sent before. In this case, it means a member decided to stop evaluating the member. For the demonstration purpose this study take away kudo sent to Stefan Kung and the following confirmation screen appears in Figure A.15.



You have taken kudos away from Stefan Küng.

Figure A.15: Shows a kudo taken back confirmation page.

And, the process can be shown in xml file after API call as shown in Figure A.16.


https://www.openhub.net/accounts/337/kudos.xml?api_key=Y2VxYH06Z4F4aHd7bxhLUQ

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<response>
   <status>success</status>
   <items_returned>165</items_returned>
   <items_available>165</items_available>
   <first_item_position>0</first_item_position>
 ▼<result>
  ▼<kudo>
     <sender_account_id>641506</sender_account_id>
     <sender_account_name>moritzrau77</sender_account_name>
     <receiver_account_id>337</receiver_account_id>
     <receiver_account_name>Stefan Küng</receiver_account_name>
     <created_at>2014-08-06T11:38:08Z</created_at>
   </kudo>
  ▼<kudo>
     <sender_account_id>640387</sender_account_id>
     <sender_account_name>brusche10</sender_account_name>
     <receiver_account_id>337</receiver_account_id>
     <receiver_account_name>Stefan Küng</receiver_account_name>
     <created_at>2014-08-05T12:37:02Z</created_at>
   </kudo>
  ▼<kudo>
     <sender_account_id>614493</sender_account_id>
     <sender_account_name>Snaehild</sender_account_name>
     <receiver_account_id>337</receiver_account_id>
     <receiver_account_name>Stefan Küng</receiver_account_name>
     <created_at>2014-07-01T12:56:05Z</created_at>
   </kudo>
  ▼<kudo>
     <sender_account_id>611358</sender_account_id>
     <sender_account_name>ziggythepinhead</sender_account_name>
     <receiver_account_id>337</receiver_account_id>
     <receiver_account_name>Stefan Küng</receiver_account_name>
     <created_at>2014-06-27T15:08:22Z</created_at>
   </kudo>
  ▼<kudo>
     <sender_account_id>589082</sender_account_id>
     <sender_account_name>Jim Thomas</sender_account_name>
     <receiver_account_id>337</receiver_account_id>
     <receiver_account_name>Stefan Küng</receiver_account_name>
```

Figure A.16: Shows a kudo taken back xml file after API call.

Appendix B

Table showing summary of developers contributions based on first commit dates

	GROUP ONE :FIRST COMMIT DATE - 2012-03-26										
Members	Member _id	KR	KP	NKR	Commits	Projects	Man month	Age		size	
39	15097	8	14821	1	2263	Number = 36	285	MWECB	33	VLDT	29
						TCBR = 26024		YECB	0	ASDT	4
						TC= 3,137,230		Unknown	3	SD / SDT	1
						TLC = 66,738,948				Unknown	2
	152718	9	10914	3	1461	Number = 28	89	MWECB	25	VLDT	23
						TCBR = 16,828		YECB	2	ASDT	4
						TC= 1,440,950		Unknown	1	SD / SDT	1
						TLC = 28,803,169				Unknown	0
	195069	9	4740	2	188	Number = 2	35	MWECB	2	VLDT	2
						TCBR = 644		YECB	0	ASDT	0
						TC= 41781		Unknown	0	SD / SDT	0
						TLC = 881,704				Unknown	0
	24597	9	1834	5	3171	Number = 10	139	MWECB	10	VLDT	8
						TCBR = 4593		YECB	0	ASDT	1
						TC= 326,831		Unknown	0	SD / SDT	1
						TLC = 4,548,506				Unknown	0
	27129	9	7430	4	5318	Number = 40	1926	MWECB	39	VLDT	19
						TCBR = 775,630		YECB	0	ASDT	10
						TC= 26,405,957		Unknown	1	SD / SDT	10
						TLC = 1,154,193,707				Unknown	1
	28241	9	2789	3	2287	Number = 7	179	MWECB	7	VLDT	3
						TCBR = 6865		YECB	0	ASDT	1
						TC= 1,145,903		Unknown	0	SD / SDT	3
						TLC = 20,840,893				Unknown	0
	32419	9	6212	4	11	Number = 2	3	MWECB	2	VLDT	1
						TCBR = 186		YECB	0	ASDT	1
						TC= 14,183		Unknown	0	SD / SDT	0
						TLC = 1,175,802				Unknown	0
	50300	9	9560	1	1074	Number = 7	88	MWECB	4	VLDT	4
						TCBR = 536		YECB	3	ASDT	3
						TC= 18995		Unknown	0	SD / SDT	0
						TLC = 292,033				Unknown	0

	GROUP ONE :FIRST COMMIT DATE - 2012-03-26										
Members	Member _id	KR	KP	NKR	Commits	Projects	Man month	Age		size	
	214087	8	18507	1	7310	Number = 20	39	MWECB	16	VLDT	14
						TCBR = 111,232		YECB	1	ASDT	3
						TC= 4,939,597		Unknown	3	SD / SDT	0
						TLC = 185,261,438				Unknown	3
	151250	8	39032	0	1107	Number = 10	77	MWECB	8	VLDT	2
						TCBR = 9963		YECB	2	ASDT	6
						TC= 94486		Unknown	0	SD / SDT	2
						TLC = 2,285,235				Unknown	
	522676	7	73676	0	848	Number = 2	49	MWECB	2	VLDT	2
						TCBR = 236		YECB	0	ASDT	0
						TC= 31117		Unknown	0	SD / SDT	0
						TLC = 1,639,804				Unknown	0
	581948	6	213138	0	337	Number = 6	27	MWECB	3	VLDT	6
						TCBR = 1610		YECB	3	ASDT	0
						TC= 19063		Unknown	0	SD / SDT	0
						TLC = 262644				Unknown	0
	229259	8	38890	0	154	Number = 4	37	MWECB	4	VLDT	4
						TCBR = 1365		YECB	0	ASDT	0
						TC= 157190		Unknown	0	SD / SDT	0
						TLC = 670820				Unknown	0
	557651	8	49845	0	108	Number = 1	15	MWECB	1	VLDT	1
						TCBR = 293		YECB	0	ASDT	0
						TC= 23785		Unknown	0	SD / SDT	0
						TLC = 1,535,979			0	Unknown	0
	143869	7	123499	0	2373	Number = 17	144	MWECB	7	VLDT	11
						TCBR = 599		YECB	9	ASDT	3
						TC= 14,963		Unknown	1	SD / SDT	2
						TLC = 215,048				Unknown	1
	261491	8	28321	0	1991	Number = 15	253	MWECB	5	VLDT	13
						TCBR = 905		YECB	10	ASDT	2
						TC= 7383		Unknown	0	SD / SDT	0
						TLC = 216,532				Unknown	0

	GROUP ONE :FIRST COMMIT DATE - 2012-03-26										
Members	Member _id	KR	KP	NKR	Commits	Projects	Man month	Age		size	
	504048	7	89237	0	61	Number = 1	12	MWECB	0	VLDT	0
						TCBR = 11		YECB	1	ASDT	0
						TC= 533		Unknown	0	SD / SDT	1
						TLC = 190,398				Unknown	0
	51745	6	198295	0	230	Number = 3	22	MWECB	0	VLDT	3
						TCBR = 82		YECB	3	ASDT	0
						TC= 1774		Unknown	0	SD / SDT	0
						TLC = 7284				Unknown	0
	10154	9	12820	0	104	Number = 5	14	MWECB	5	VLDT	5
						TCBR = 1184		YECB	0	ASDT	0
						TC= 141605		Unknown	0	SD / SDT	0
						TLC = 461947				Unknown	0
	88863	8	37861	0	1160	Number = 26	141	MWECB	25	VLDT	23
						TCBR = 17184		YECB	0	ASDT	2
						TC= 1,363,651		Unknown	1	SD / SDT	0
						TLC = 108,686,040				Unknown	1
	98861	5	319957	0	27	Number = 2	13	MWECB	1	VLDT	2
						TCBR = 802		YECB	1	ASDT	0
						TC= 28,619		Unknown	0	SD / SDT	0
						TLC = 277,380				Unknown	0
	155678	7	141151	0	22	Number = 1	4	MWECB	1	VLDT	1
						TCBR = 39		YECB	0	ASDT	0
						TC= 8534		Unknown	0	SD / SDT	0
						TLC = 194834				Unknown	0
	240600	7	138452	0	20	Number = 2	6	MWECB	2	VLDT	2
						TCBR = 13905		YECB	0	ASDT	0
						TC= 544,083		Unknown	0	SD / SDT	0
						TLC = 18,230,647				Unknown	0
	29880	8	17996	0	108	Number = 6	16	MWECB	5	VLDT	5
						TCBR = 1095		YECB	1	ASDT	0
						TC= 161,545		Unknown	0	SD / SDT	1
						TLC = 11,383,587				Unknown	0

	GROUP ONE :FIRST COMMIT DATE - 2012-03-26										
Members	Member _id	KR	KP	NKR	Commits	Projects	Man month	Age		size	
	121485	6	208724	0	1	Number = 1	1	MWECB	1	VLDT	0
						TCBR = 131		YECB	0	ASDT	1
						TC= 3829		Unknown	0	SD / SDT	0
						TLC = 21,628				Unknown	0
	136317	8	18967	0	5087	Number = 64	263	MWECB	58	VLDT	38
						TCBR = 587,404		YECB	1	ASDT	12
						TC= 21,253,803		Unknown	5	SD / SDT	6
						TLC = 719,288,990				Unknown	5
	143475	6	185906	0	6	Number = 2	5	MWECB	2	VLDT	2
						TCBR =192		YECB	0	ASDT	0
						TC= 16259		Unknown	0	SD / SDT	0
						TLC = 237,031				Unknown	0
	152187	8	19536	0	1695	Number = 6	81	MWECB	6	VLDT	2
						TCBR = 511		YECB	0	ASDT	3
						TC= 26158		Unknown	0	SD / SDT	1
						TLC = 548,214				Unknown	0
	154218	5	378547	0	54	Number = 2	3	MWECB	2	VLDT	1
						TCBR = 41		YECB	0	ASDT	0
						TC= 3016		Unknown	0	SD / SDT	1
						TLC = 82,501				Unknown	0
	154707	5	339413	0	5	Number = 2	2	MWECB	1	VLDT	1
						TCBR = 29		YECB	1	ASDT	1
						TC= 976		Unknown	0	SD / SDT	0
						TLC = 6468				Unknown	0
	213675	7	97104	0	22	Number = 2	11	MWECB	2	VLDT	2
						TCBR =2404		YECB	0	ASDT	0
						TC= 147,998		Unknown	0	SD / SDT	0
						TLC = 2,059,409				Unknown	0
	223553	7	87125	0	87	Number = 10	33	MWECB	10	VLDT	9
						TCBR = 10169		YECB	0	ASDT	1
						TC= 659,131		Unknown	0	SD / SDT	0
						TLC = 9,687,251				Unknown	0

	GROUP ONE :FIRST COMMIT DATE - 2012-03-26										
Members	Member _id	KR	KP	NKR	Commits	Projects	Man month	Age		size	
	320750	8	35729	0	1981	Number = 24	167	MWECB	21	VLDT	19
						TCBR = 49,563		YECB	1	ASDT	2
						TC= 2,734,628		Unknown	2	SD / SDT	1
						TLC = 193,414,538				Unknown	2
	555925	8	56365	0	50	Number = 7	18	MWECB	5	VLDT	6
						TCBR = 1660		YECB	1	ASDT	0
						TC= 80204		Unknown	1	SD / SDT	1
						TLC = 2,060,579				Unknown	0
	562870	9	8004	0	335	Number = 13	34	MWECB	13	VLDT	7
						TCBR = 4733		YECB	0	ASDT	6
						TC= 649,598		Unknown	0	SD / SDT	0
						TLC = 3,355,756				Unknown	0
	577651	8	49845	0	108	Number = 1	15	MWECB	1	VLDT	1
						TCBR = 293		YECB	0	ASDT	0
						TC= 23,785		Unknown	0	SD / SDT	0
						TLC = 1,535,979				Unknown	0
	61356	7	82643	0	321	Number = 14	38	MWECB	12	VLDT	13
						TCBR = 28,141		YECB	2	ASDT	1
						TC= 1,099,466		Unknown	0	SD / SDT	0
						TLC = 36,107,274				Unknown	0
	6159	7	127839	0	44	Number = 11	11	MWECB	7	VLDT	7
						TCBR = 10148		YECB	2	ASDT	1
						TC= 190,700		Unknown	2	SD / SDT	1
						TLC = 1,152,864				Unknown	2
	83544	8	35535	0	46988	Number = 10	752	MWECB	8	VLDT	7
						TCBR =8390		YECB	0	ASDT	1
						TC= 1,314,196		Unknown	2	SD / SDT	1
						TLC = 43,371,689				Unknown	1
	8608	8	14486	0	3775	Number = 77	528	MWECB	61	VLDT	49
						TCBR = 211,610		YECB	0	ASDT	5
						TC= 10,153,132		Unknown	16	SD / SDT	7
						TLC = 332,576,589				Unknown	16

	GROUP TWO : FIRST COMMIT DATE : 2012-07.05										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
39	11217	8	47687	0	487	Number = 34	70	MWECB	29	VLDT	32
						TCBR = 30,167		YECB	4	ASDT	2
						TC= 487		Unknown	1	SD /	0
						TLC = 7,754,625				Unknown	0
	112328	7	88756	0	556	Number = 3	36	MWECB	1	VLDT	2
						TCBR = 188		YECB	2	ASDT	1
						TC= 16293		Unknown	0	SD / SDT	0
						TLC = 230,387				Unknown	0
	112162	7	119769	0	177	Number = 1	16	MWECB	1	VLDT	1
						TCBR = 42		YECB	0	ASDT	0
						TC= 10,484		Unknown	0	SD / SDT	0
						TLC = 175,639				Unknown	0
	125335	8	18823	1	6680	Number = 31	293	MWECB	21	VLDT	19
						TCBR = 119,158		YECB	4	ASDT	4
						TC= 7,842,385		Unknown	6	SD / SDT	2
						TLC = 285,460,267				Unknown	6
	133527	8	34654	1	656	Number = 1	21	MWECB	0	VLDT	1
						TCBR = 11		YECB	1	ASDT	0
						TC= 878		Unknown	0	SD / SDT	0
						TLC = 42,722				Unknown	0
	135353	7	130377	0	90	Number = 19	35	MWECB	11	VLDT	10
						TCBR = 207,977		YECB	0	ASDT	1
						TC= 8,007,574		Unknown	8	SD / SDT	0
						TLC = 299,073,432				Unknown	8
	140222	9	10376	1	2153	Number = 6	80	MWECB	5	VLDT	4
						TCBR = 683		YECB	1	ASDT	2
						TC= 18,680		Unknown	0	SD / SDT	0
						TLC = 170,047				Unknown	0
	141853	8	20863	1	420	Number = 37	11	MWECB	33	VLDT	30
						TCBR = 22344		YECB	4	ASDT	4
						TC= 2,116,811		Unknown	0	SD / SDT	3
						TLC = 169,278,961				Unknown	1

	GROUP TWO : FIRST COMMIT DATE : 2012-07.05										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
39	185938	9	3918	0	54	Number = 4	13	MWECB	4	VLDT	4
						TCBR = 12,421		YECB	0	ASDT	0
						TC= 1,112,950		Unknown	0	SD /	0
						TLC = 17,239,755				Unknown	0
	2071	9	9310	2	949	Number = 12	157	MWECB	11	VLDT	12
						TCBR = 16,303		YECB	1	ASDT	0
						TC= 1,126,876		Unknown	0	SD / SDT	0
						TLC = 46,065,621				Unknown	0
	214881	7	157398	0	96	Number = 7	26	MWECB	6	VLDT	7
						TCBR = 213		YECB	1	ASDT	0
						TC= 29,220		Unknown	0	SD / SDT	0
						TLC = 1,849,998				Unknown	0
	216289	1	914740	0	143	Number = 1	2	MWECB	0	VLDT	1
						TCBR = 44		YECB	1	ASDT	0
						TC= 1,700		Unknown	0	SD / SDT	0
						TLC = 609,644				Unknown	0
	222160	5	360999	0	2	Number = 1	2	MWECB	1	VLDT	1
						TCBR = 784		YECB	0	ASDT	0
						TC= 28,565		Unknown	0	SD / SDT	0
						TLC = 277,138				Unknown	0
	222223	7	123278	0	26	Number = 7	9	MWECB	6	VLDT	6
						TCBR = 1152		YECB	1	ASDT	0
						TC= 84,075		Unknown	0	SD / SDT	1
						TLC =2,407,635				Unknown	0
	222971	7	109418	0	208	Number = 6	40	MWECB	5	VLDT	5
						TCBR = 2122		YECB	1	ASDT	1
						TC= 190,353		Unknown	0	SD / SDT	0
						TLC = 5,436,661				Unknown	0
	226187	8	28013	2	1183	Number = 6	112	MWECB	6	VLDT	6
						TCBR = 6526		YECB	0	ASDT	0
						TC= 354,187		Unknown	0	SD / SDT	0
						TLC = 5,753,689				Unknown	0

	GROUP TWO : FIRST COMMIT DATE : 2012-07.05										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
39	236638	8	22648	0	4347	Number = 1	22	MWECB	0	VLDT	1
						TCBR = 84		YECB	1	ASDT	0
						TC= 27,744		Unknown	0	SD /	0
						TLC = 2,301,488				Unknown	0
	2384	9	5270	2	9700	Number = 31	654	MWECB	29	VLDT	24
						TCBR = 14,873		YECB	1	ASDT	5
						TC= 748,193		Unknown	1	SD / SDT	2
						TLC = 21,552,383				Unknown	0
	27638	9	8037	4	766	Number = 41	118	MWECB	36	VLDT	33
						TCBR = 24121		YECB	5	ASDT	4
						TC= 1,740,134		Unknown	0	SD / SDT	4
						TLC = 99,614,456				Unknown	0
	290	9	925	6	6413	Number = 7	219	MWECB	6	VLDT	5
						TCBR = 3138		YECB	0	ASDT	1
						TC= 417,605		Unknown	1	SD / SDT	1
						TLC = 23,687,645				Unknown	0
	298247	8	29796	0	1053	Number = 6	66	MWECB	4	VLDT	6
						TCBR = 4606		YECB	2	ASDT	0
						TC= 177,965		Unknown	0	SD / SDT	0
						TLC = 2,736,644				Unknown	0
	300028	8	50738	2	771	Number = 17	13	MWECB	15	VLDT	16
						TCBR = 2788		YECB	2	ASDT	1
						TC= 30,958		Unknown	0	SD / SDT	0
						TLC = 322,018				Unknown	0
	305378	7	88544	0	985	Number = 3	39	MWECB	1	VLDT	2
						TCBR = 57		YECB	2	ASDT	0
						TC= 4638		Unknown	0	SD / SDT	1
						TLC = 760,887				Unknown	0
	323583	1	1019450	0	42	Number = 1	8	MWECB	1	VLDT	1
						TCBR = 200		YECB	0	ASDT	0
						TC= 5518		Unknown	0	SD / SDT	0
						TLC = 1,082,246				Unknown	0

	GROUP TWO : FIRST COMMIT DATE : 2012-07.05										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
39	34264	8	48075	0	155	Number = 7	18	MWECB	5	VLDT	6
						TCBR = 29,542		YECB	0	ASDT	1
						TC= 14,134,164		Unknown	2	SD /	0
						TLC = 45,632,165				Unknown	0
	3506	9	1703	15	6038	Number = 14	191	MWECB	12	VLDT	12
						TCBR = 15,785		YECB	0	ASDT	0
						TC= 2,987,314		Unknown	2	SD / SDT	0
						TLC = 86,496,475				Unknown	2
	359800	9	6933	2	6496	Number = 28	240	MWECB	22	VLDT	20
						TCBR = 142,367		YECB	1	ASDT	2
						TC= 5,148,639		Unknown	5	SD / SDT	1
						TLC = 243,544,448				Unknown	5
	43413	9	9795	1	6712	Number = 13	265	MWECB	11	VLDT	12
						TCBR = 14,157		YECB	2	ASDT	0
						TC= 1,505,158		Unknown	0	SD / SDT	1
						TLC = 170,673,108				Unknown	0
	47370	9	4745	0	193	Number = 8	30	MWECB	6	VLDT	7
						TCBR = 4112		YECB	2	ASDT	0
						TC= 228,247		Unknown	0	SD / SDT	1
						TLC = 3,614,911				Unknown	0
	52541	9	10327	1	2515	Number = 12	151	MWECB	10	VLDT	10
						TCBR = 27,980		YECB	1	ASDT	0
						TC= 2,000,964		Unknown	1	SD / SDT	1
						TLC =44,717,619				Unknown	1
	59874	8	23820	1	256	Number = 26	83	MWECB	26	VLDT	20
						TCBR = 206,503		YECB	0	ASDT	3
						TC= 8,842,676		Unknown	0	SD / SDT	2
						TLC = 355,492,080				Unknown	1
	60504	9	87	3	7583	Number = 75	384	MWECB	28	VLDT	51
						TCBR = 6142		YECB	38	ASDT	19
						TC= 236,512		Unknown	9	SD / SDT	5
						TLC = 77,029,665				Unknown	0

	GROUP TWO : FIRST COMMIT DATE : 2012-07.05										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
39	6098	9	1382	3	1416	Number = 6	123	MWECB	4	VLDT	1
						TCBR = 67		YECB	2	ASDT	3
						TC= 1724		Unknown	0	SD / SDT	2
						TLC = 21047				Unknown	0
	61575	8	28365	0	20	Number = 1	7	MWECB	0	VLDT	1
						TCBR = 12		YECB	1	ASDT	0
						TC= 626		Unknown	0	SD / SDT	0
						TLC = 101,222				Unknown	0
	6673	9	192	28	18,179	Number = 64		MWECB	54	VLDT	32
						TCBR = 6,476		YECB	7	ASDT	20
						TC= 465,022		Unknown	3	SD / SDT	10
						TLC = 25,243,499				Unknown	2
	70262	9	3291	3	2132	Number = 14	96	MWECB	8	VLDT	11
						TCBR = 6512		YECB	6	ASDT	2
						TC= 570,648		Unknown	0	SD / SDT	1
						TLC = 28,157,517				Unknown	0
	80400	8	24518	1	1041	Number = 9	101	MWECB	8	VLDT	4
						TCBR = 25,970		YECB	0	ASDT	3
						TC= 1,062,293		Unknown	1	SD / SDT	1
						TLC = 31,638,401				Unknown	0
	87087	8	15585	1	2387	Number = 33	231	MWECB	32	VLDT	30
						TCBR = 20,581		YECB	0	ASDT	2
						TC= 2,229,315		Unknown	1	SD / SDT	0
						TLC =48,894,330				Unknown	1
	9263	9	7418	4	639	Number = 14	41	MWECB	11	VLDT	11
						TCBR = 26,818		YECB	1	ASDT	2
						TC= 938,547		Unknown	2	SD / SDT	0
						TLC = 37,233,579				Unknown	1

	GROUP THREE : FIRST COMMIT DATE : 2012-05-10										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
36	39950	9	6367	9	301	Number = 10	47	MWECB	8	VLDT	8
						TCBR = 6013		YECB	2	ASDT	2
						TC= 101,346		Unknown	0	SD /	0
						TLC = 1,218,725				Unknown	0
	8511	9	3900	7	11490	Number = 36	268	MWECB	30	VLDT	28
						TCBR = 158,755		YECB	1	ASDT	1
						TC= 7,870,618		Unknown	5	SD / SDT	2
						TLC = 348,083,776				Unknown	5
	35110	7	79240	0	386	Number =23	59	MWECB	17	VLDT	18
						TCBR = 14,360		YECB	5	ASDT	3
						TC= 225,820		Unknown	1	SD / SDT	1
						TLC = 41,004,582				Unknown	1
	100723	9	4024	3	3839	Number = 70	212	MWECB	64	VLDT	65
						TCBR = 48,996		YECB	5	ASDT	2
						TC= 2,597,515		Unknown	1	SD / SDT	2
						TLC = 54,229,705				Unknown	1
	100764	9	14,206	4	483	Number = 5	68	MWECB	5	VLDT	5
						TCBR = 1811		YECB	0	ASDT	0
						TC= 273,973		Unknown	0	SD / SDT	0
						TLC = 21,715,869				Unknown	0
	101518	8	37197	1	422	Number = 13	84	MWECB	12	VLDT	11
						TCBR = 9032		YECB	1	ASDT	1
						TC= 622,515		Unknown	0	SD / SDT	1
						TLC = 11,103,521				Unknown	0
	109973	8	19980	0	976	Number = 6	61	MWECB	6	VLDT	6
						TCBR = 2132		YECB	0	ASDT	0
						TC= 238,921		Unknown	0	SD / SDT	0
						TLC = 6,922,292				Unknown	0
	122800	8	19717	1	1362	Number = 17	99	MWECB	15	VLDT	12
						TCBR = 5330		YECB	1	ASDT	4
						TC= 208,965		Unknown	1	SD / SDT	1
						TLC = 38,726,919				Unknown	0

	GROUP THREE : FIRST COMMIT DATE : 2012-05-10										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
36	12291	9	857	15	50799	Number = 65	1354	MWECB	44	VLDT	42
						TCBR = 6053		YECB	17	ASDT	11
						TC= 827,247		Unknown	4	SD /	8
						TLC = 19,439,142				Unknown	4
	12568	9	1762	15	4030	Number = 20	161	MWECB	19	VLDT	15
						TCBR = 31,186		YECB	0	ASDT	0
						TC= 6,416,496		Unknown	1	SD / SDT	4
						TLC = 149,436,881				Unknown	1
	15124	9	1362	14	4688	Number = 28	224	MWECB	16	VLDT	25
						TCBR = 9120		YECB	12	ASDT	1
						TC= 710,557		Unknown	0	SD / SDT	2
						TLC = 4,454,068				Unknown	0
	153366	6	173179	0	3	Number = 1	0	MWECB	1	VLDT	1
						TCBR = 55		YECB	0	ASDT	0
						TC= 9938		Unknown	0	SD / SDT	0
						TLC = 299,812				Unknown	0
	154704	5	339413	0	5	Number = 2	2	MWECB	1	VLDT	1
						TCBR = 29		YECB	1	ASDT	1
						TC= 976		Unknown	0	SD / SDT	0
						TLC = 6468				Unknown	0
	155035	9	3286	1	3582	Number = 31	103	MWECB	10	VLDT	19
						TCBR = 7555		YECB	19	ASDT	6
						TC= 223,455		Unknown	1	SD / SDT	4
						TLC =4,984,287				Unknown	1
	190776	6	181169	0	370	Number = 4	33	MWECB	3	VLDT	4
						TCBR = 108		YECB	1	ASDT	0
						TC= 3562		Unknown	0	SD / SDT	0
						TLC = 133,692				Unknown	0
	19137	9	8022	2	5687	Number = 5	309	MWECB	4	VLDT	5
						TCBR = 1000		YECB	1	ASDT	0
						TC= 216,295		Unknown	0	SD / SDT	0
						TLC = 6,479,028				Unknown	0

	GROUP THREE : FIRST COMMIT DATE : 2012-05-10										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
36	204483	9	8022	0	1507	Number = 20	61	MWECB	19	VLDT	16
						TCBR = 16651		YECB	1	ASDT	1
						TC= 1,777,110		Unknown	0	SD / SDT	2´3
						TLC = 32,314,674				Unknown	0
	20861	8	49834	0	1020	Number = 12	57	MWECB	12	VLDT	10
						TCBR = 2785		YECB	0	ASDT	2
						TC= 640,524		Unknown	0	SD / SDT	0
						TLC = 4,174,830				Unknown	0
	215731	7	120001	0	16	Number = 1	6	MWECB	0	VLDT	0
						TCBR = 5		YECB	1	ASDT	1
						TC= 295		Unknown	0	SD / SDT	0
						TLC = 40,447				Unknown	0
	2199	9	4644	9	1096	Number = 7	100	MWECB	7	VLDT	6
						TCBR = 28968		YECB	0	ASDT	1
						TC= 1,246.680		Unknown	0	SD / SDT	0
						TLC =				Unknown	0
	224068	7	81977	0	282	Number = 1	18	MWECB	1	VLDT	1
						TCBR = 229		YECB	0	ASDT	0
						TC= 10,844		Unknown	0	SD / SDT	0
						TLC = 1,022,846				Unknown	0
	230044	7	134392	9	36	Number = 4	10	MWECB	2	VLDT	2
						TCBR = 2564		YECB	0	ASDT	2
						TC= 77,912		Unknown	2	SD / SDT	0
						TLC = 731,586				Unknown	0
	232377	8	42096	0	1216	Number = 14	97	MWECB	11	VLDT	9
						TCBR = 2144		YECB	3	ASDT	3
						TC= 71,296		Unknown	0	SD / SDT	1
						TLC = 2,097,284				Unknown	1
	234924	8	28845	0	1479	Number = 2	45	MWECB	1	VLDT	0
						TCBR = 29		YECB	1	ASDT	2
						TC= 3279		Unknown	0	SD / SDT	0
						TLC = 178,909				Unknown	0

	GROUP THREE : FIRST COMMIT DATE : 2012-05-10										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
36	239697	7	145984	0	15	Number = 1	7	MWECB	1	VLDT	1
						TCBR = 305		YECB	0	ASDT	0
						TC= 18,848		Unknown	0	SD /	0
						TLC = 135,636				Unknown	0
	250673	7	122113	0	39	Number = 6	18	MWECB	5	VLDT	6
						TCBR = 2035		YECB	1	ASDT	0
						TC= 47,357		Unknown	0	SD / SDT	0
						TLC = 613,793				Unknown	0
	307776	6	251620	0	1	Number = 1	0	MWECB	1	VLDT	1
						TCBR = 331		YECB	0	ASDT	0
						TC= 4526		Unknown	0	SD / SDT	0
						TLC = 103,131				Unknown	0
	356564	8	26299	1	301	Number = 5	18	MWECB	3	VLDT	2
						TCBR = 172		YECB	2	ASDT	3
						TC= 5817		Unknown	0	SD / SDT	0
						TLC = 61254				Unknown	0
	40423	8	19615	2	2637	Number = 8	74	MWECB	4	VLDT	7
						TCBR = 2273		YECB	3	ASDT	0
						TC= 48,532		Unknown	1	SD / SDT	0
						TLC = 6,286,013				Unknown	1
	49302	7	106019	0	1138	Number = 12	77	MWECB	6	VLDT	7
						TCBR = 27,537		YECB	5	ASDT	4
						TC= 1,199,175		Unknown	1	SD / SDT	1
						TLC = 37,440,335				Unknown	0
	569819	7	92010	0	56	Number = 19	25	MWECB	13	VLDT	11
						TCBR = 2171		YECB	5	ASDT	2
						TC= 79,196		Unknown	2	SD / SDT	5
						TLC = 1,501,118				Unknown	1
	59723	8	21649	2	128	Number = 4	8	MWECB	4	VLDT	2
						TCBR = 100		YECB	0	ASDT	1
						TC= 20,089		Unknown	0	SD / SDT	1
						TLC = 441,402				Unknown	0

	GROUP THREE : FIRST COMMIT DATE : 2012-05-10										
Member	Member_id	KR	KP	NKR	Commits	Project	Man month	Age		Size	
36	69072	9	5792	3	1799	Number = 38	229	MWECB	28	VLDT	25
						TCBR = 17,100		YECB	10	ASDT	9
						TC= 1,576,627		Unknown	0	SD / SDT	3
						TLC = 160,918,610				Unknown	1
	7277	9	5353	0	4094	Number = 15	113	MWECB	15	VLDT	12
						TCBR = 12,533		YECB	0	ASDT	2
						TC= 987,370		Unknown	0	SD / SDT	1
						TLC = 19,445,011				Unknown	0
	76833	6	232033	0	20	Number = 5	3	MWECB	4	VLDT	5
						TCBR = 1965		YECB	1	ASDT	0
						TC= 15,480		Unknown	0	SD / SDT	0
						TLC = 196,650				Unknown	0
	8398	9	3103	3	2511	Number = 53	441	MWECB	50	VLDT	48
						TCBR = 60,453		YECB	2	ASDT	3
						TC= 2,337,538		Unknown	1	SD / SDT	1
						TLC = 50,997,180				Unknown	1

Bibliography

[1] Petri Sirkkala, Imed Hammouda, and Timo Aaltonen. From proprietary to open source: Building a network of trust. 2010.

[2] Katherine J Stewart and Sanjay Gosain. The impact of ideology on effectiveness in open source software development teams. *Mis Quarterly*, pages 291–314, 2006.

[3] Michael S Lane, Glen van der Vyver, Prajwal Basnet, and Srecko Howard. Interpretative insights into interpersonal trust and effectiveness of virtual communities of open source software (oss) developers. 2004.

[4] Paul B De Laat. How can contributors to open-source communities be trusted? on the assumption, inference, and substitution of trust. *Ethics and Information Technology*, 12(4):327–341, 2010.

[5] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. Social coding in github: transparency and collaboration in an open software repository. pages 1277–1286, 2012.

[6] Haibin Zhang, Yan Wang, and Xiuzhen Zhang. A trust vector approach to transaction context-aware trust evaluation in e-commerce and e-service environments. pages 1–8, 2012.

[7] Paolo Massa and Paolo Avesani. Controversial users demand local trust metrics: An experimental study on epinions. com community. 20(1):121, 2005.

[8] Jennifer Golbeck and James Hendler. Inferring binary trust relationships in web-based social networks. *ACM Transactions on Internet Technology (TOIT)*, 6(4): 497–529, 2006.

[9] Georgios Gousios, Eirini Kalliamvakou, and Diomidis Spinellis. Measuring developer contribution from software repository data. pages 129–132, 2008.

[10] Maria Antikainen, Timo Aaltonen, and Jaani Väisänen. The role of trust in oss communitiescase linux kernel community. pages 223–228, 2007.

[11] Oliver Arafat and Dirk Riehle. The commit size distribution of open source software. pages 1–8, 2009.

[12] Daning Hu and J Leon Zhao. A comparison of evaluation networks and collaboration networks in open source software communities. *AMCIS 2008 Proceedings*, page 277, 2008.

[13] Daning Hu, J Leon Zhao, and Jiesi Cheng. Reputation management in an open source developer social network: An empirical study on determinants of positive evaluations. *Decision Support Systems*, 53(3):526–533, 2012.

[14] Rosalva E Gallardo-Valencia, Phitchayaphong Tantikul, and Susan Elliott Sim. Searching for reputable source code on the web. pages 183–186, 2010.

[15] Dick Stenmark. Distrust in information systems research. *Proceedings of HICSS-46, Maui, Hawaii, January 7-10, 2013*, 2013.

[16] Heikki Orsila, Jaco Geldenhuys, Anna Ruokonen, and Imed Hammouda. Trust issues in open source software development. pages 9–12, 2009.

[17] Florian S Gysin and Adrian Kuhn. A trustability metric for code search based on developer karma. pages 41–44, 2010.

[18] Ramanthan Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Propagation of trust and distrust. pages 403–412, 2004.

[19] Jennifer Golbeck. Analyzing the social web. 2013.

[20] Jennifer Golbeck. Computing with social trust. 2008.

[21] Thomas DuBois, Jennifer Golbeck, and Aravind Srinivasan. Predicting trust and distrust in social networks. pages 418–424, 2011.

[22] Audun Jøsang. Trust and reputation systems. pages 209–245, 2007.

[23] Jennifer Golbeck and Ugur Kuter. The ripple effect: change in trust and its impact over a social network. pages 169–181, 2009.

[24] Jennifer Ann Golbeck. Computing and applying trust in web-based social networks. 2005.

[25] Ohloh api. https://github.com/blackducksw/ohloh_api/. Last Accessed: 2014-09-01.

[26] Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*, 14(2):131–164, 2009.

[27] M.M.Mahbubul Syeed and Imed Hammouda. Who contributes to what? exploring hidden relationships between floss projects. 427:21–30, 2014. doi: 10.1007/978-3-642-55128-4_3. URL http://dx.doi.org/10.1007/978-3-642-55128-4_3.