UNIVERSITY OF TARTU
Institute of Computer Science
Software Engineering Curriculum

Hariti Atulkumar Sanchaniya

Communication Overhead In Open Source Software Projects

Master's Thesis (30 ECTS)

Supervisor(s): Ezequiel Scott

Tartu 2021

Communication Overhead In Open Source Software Projects

Abstract:

Open source software development is gaining more popularity in recent times. It is predominantly because of the flexible nature of open source software projects. Countless famous organizations also have started investing their resources in open source software development teams. In addition to this, agile software development has been dominating the software development domain for several years. Agile software development focuses on interactions, collaboration, and requirement changes therefore communication between contributors in open source software projects following agile development has notable importance. In open-source software projects participation of contributors solely depends on their motivation, due to this contributor's longevity in the project is also significant. Since productivity is of interest in many research fields, therefore, this study focuses on analyzing the effect of communication overhead and team member longevity on productivity for open source software projects that are following agile software development. Two projects, MESOS and FABRIC have been analyzed in this thesis. The paper starts with defining hypotheses and afterward analyzing them using statistical tests. The result of the analysis shows that there is no correlation between communication overhead, team member longevity, and productivity which was studied based on velocity for the considered open-source software projects. This paper also presents limitations of the thesis and work that can be considered as further research on the topic.

Keywords:

Communication overhead, open-source software projects, team member longevity, productivity

CERCS: P170 - Computer science, numerical analysis, systems, control

Kommunikatsiooni üleküllus avatud lähtekoodiga tarkvaraarendusprojektides

Lühikokkuvõte:

Avatud lähtekoodiga tarkvaraarenduse populaarus kasvab üha enam. Enamjaolt on selle põhjuseks avatud lähtekoodika arenduste paindlik olemus. Lugematu arv tuntuid ettevõtteid on alustanud ressursside investeerimist avatud lähtekoodiga tarkvaraarendusmeeskondadesse. Lisaks sellele on mitu aastat tarkvaraarenduse keskkonda domineerinud agiilne tarkvaraarenduse metoodika. Agiilne arendus keskendub panustajate vahelisse suhtlusesse, koostöösse ja nõuete muutustesse, sellest tulenevalt on märkimisväärselt tähtsal kohal panustajate vaheline suhtlus avatud lähtekoodiga projektides, kus jälgitakse agiilset arenduskäiku. Avatud lähtekoodiga projektist osavõtmine on suuresti seotud panustaja motivatsiooniga, sellest tulenevalt on nende pikaealisus projektis samuti tähtsal kohal. Kuna produktiivsus on huvialaks paljudes uuringutes, siis see uurimistöö analüüsib kommunikatsiooni üleküllust ja liikmete pikaajalisuse suhet avatud lähtekoodiga tarkvaraarendusmeeskonnas, kus jälgitakse agiilset metoodikat. MESOS ja FABRIC olid kaks projekti mida analüüsiti. Uurimistöö algab hüpoteeside kirjeldusest, millele järgneb analüüs koos statistiliste katsetega.
Analüüsi tulemus näitab, et vaadeldud avatud lähtekoodiga projektides puudub kiirusmäära põhjal uuritud korrelatsioon kommunikatsiooni ülekülluse, panustajate pikaajalisuse ja produktiivsuse vahel. Samuti kirjeldab uurimistöös ette tulnud piiranguid ja punkte mida edasi uurida.

Võtmesõnad:
Kommunikatsiooni üleküllus, avatud lähtekoodiga tarkvaraendusprojektid, meeskonnaliikmete pikaajalisus, produktiivsus.

CERCS: P170 Arvutiteadus, arvanalüüs, süsteemid, kontroll

Table of Contents

1 Introduction

Open-source software projects (OSSPs) are getting more popular because of their flexible nature and ability to give freedom to their users. OSSP users can run or use these projects in any manner that is useful to them, as well as change source code to meet their feature requirements and have their own version of the code. There are developers from all over the world who contribute to the OSSPs by maintaining, updating and creating new projects. People working on these projects find their work useful as the contribution is helpful to the community. Contributions to these projects can also be profitable and highlight their skills as developers. Contribution to OSSPs is good way to collaborate with people from all around the world, to get good ideas and a platform to develop them. Because of these advantageous reasons, today many developers are joining OSSPs.

OSSPs mainly focus on crowdsourcing. Because of its capability of giving freedom geographically and also flexibility about the time, communication between contributors becomes important to manage the development environment smoothly. Everyday need for constant and reliable communication leads to communication overhead. We can refer to communication overhead as an exchange of information in terms of messages, emails, documents, etc. Communication overhead is unnecessary for contributors and one can feel overburdened by it. Several authors have studied the role of these communications using social networks [2][3][4]. Research on areas (joint task force and humanitarian airlift mission force) other than software development shows that communication overhead might affect the productivity of an overall team [5]. Considering this study, the goal of this thesis is to explore if the relationship between communication overhead and productivity is present in the context of OSSPs.

Furthermore, team member longevity refers to how long a person has been giving service or contributing to OSSPs. Some of the recent studies show that high team longevity affects team collaboration and team performance positively [6]. Studies show that different metrics like turnover and team stability affect the productivity of OSSPs [7] while team mobility affects team creativity in terms of generating new ideas [8].

As productivity has always been an interesting point of discussion for the software development community, the main goal of the thesis is to research the relationship between communication overhead and productivity as well as the relationship between team member longevity and productivity.

1.1 Problem Statement

There are thousands of developers contributing their time and effort to OSSPs. Generally, communication between contributors and developers happens to get the information they need. Communication overhead could lead contributors to waste their time and effort, that is not desirable for an individual as well as for the whole project.

In addition, free and open accessibility of these OSSPs leads to many different options (feature development, documentation, design, etc.) to contribute. There are developers or contributors whose contribution for one single project can differ from some days to months or years. This leads to different longevity of the contributors.

Communication overhead and team member longevity could have effects on productivity according to recent studies [6] [7] [8]. In this context, the following research questions are formulated:

RQ1: What is the relationship between communication overhead and productivity in OSSPs?

Motivation: Nowadays there are plenty of tools, media, and communication channels available. Because of the vast access to all these in OSSPs development environment contributors, core developers, and organization management teams are facing communication and collaboration issues. These issues can categories as lack of useful communication, more time spending on unproductive communication, or getting the same information from more than one source and checking it all. In cases where these kinds of communications end up having overhead, it is important to have an understanding of its interconnection with productivity. Answer to this can be helpful for OSSPs project organizers, core members, and leaders as well as to the contributors overall. Analysis of such communications could also be useful for new users to learning up the curve, make social connections.

RQ2: What is the relationship between team member longevity and productivity in OSSPs?

Motivation: Team member's independence of participating in OSSPs for as long as they want leads to its effect on productivity [9] and it is useful to know how high longevity of a contributor would affect productivity. The results will further help in having management decisions that can lead to improved OSSPs' development environment.

1.2 Thesis outline

This thesis is structure as follows. Chapter 2 describes the background of the study. It represents in detail information about three variables (communication overhead, team member longevity, and productivity) that are further discussed in the study. Chapter 3 describes the method followed for the research. It describes metrics that are considered to measure the variables. It represents defined hypotheses to answer the research questions as well as methods used to test them. The data analysis process followed for the research is also presented in the same section. Chapter 4 represents the results of the study in terms of the answers that we get from the data analysis for the defined research questions as well as test results for the hypotheses. Chapter 5 is the discussion section. In this chapter discussion of the result, if there is a relationship between communication overhead, longevity and productivity is there or not is presented. Also, a comparison of the study result with previous studies is discussed in detail. Chapter 5 also presents the limitations of the study.

2 Background

In this section, the content is focused on discussing basic knowledge and details about three concepts which will be discussed further in the thesis. These concepts are communication overhead, productivity, and team member longevity. The aim of this study is to describe the relationship between these concepts for open source projects (OSSPs) that uses agile software development.

Agile software development is a methodology where cross-functional teams use a collaborative and iterative approach to progress through different software development life cycle phases like requirements gathering, design, development, testing, and deployment. Also, some studies suggest that the use of the agile software development process helps increase the productivity of software development teams [10].

Open source projects are projects with open source licenses and are available open on the internet for everyone to access, use, or even modify its codebase as per their requirements. Handling, maintaining, creating overall people who are responsible for these projects are contributors. According to Wikipedia, there are more than 180k open source software projects with more than 40 million contributors.

OSSPs environment is global, free, and very flexible. A huge number of people are part of this community. When a new person joins the project, it is essential to get the information he will need throughout his contribution. These contributors communicate with other team members to share a knowledge base of the project [11] or discuss issues related to set up, development, design, bug reporting and solving and even some extent of social conversations are going on [3]. The use of different communication channels and the freedom of OSSPs develop situations related to communication overhead [1].

Communication overhead in simple terms is

"Communication Overhead is the proportion of time you spend communicating with members of your team instead of getting productive work done." [12]

Productivity is being an interesting topic for research for years now. Especially in the area of software development and in open source software development it is one of the most argued or discussed topics. For IT productivity can be

Productivity = Size of Application Developed per Labor consumed during development [13]

There is a lot of different kind of productivity measures or metrics that can be considered. These include sprint burn down, velocity, throughput, cycle time, code coverage, bug rates, line of code, etc. In this study, we will consider velocity as a measurement metric for productivity as we consider OSSPs that follow agile development. Studies show velocity as a well-known measure for productivity [14] [15] [20]. Here, velocity is measured as a total of story points completed per sprint.

Another parameter is team member longevity. "Longevity" in simple terms means life span. Here, team member longevity refers to the service provided by a contributor(contribution) of a person for an OSSP or a team. OSSPs teams are global, flexible, and self-managed. In this kind of environment, longevity can be determined by commits, issues raised, mail or messages replied or raise by a contributor, issues solved, the action of merging a pull request, etc. In this study, we calculate the longevity based on the number of sprints for which team member was contributing in terms of sending or receiving emails as well as solving the issues. Contribution for the high number of sprints refers to more longevity.

3 Related Work

There are many studies in the field of psychology on teams, team coordination, team productivity. However, these studies are not related to OSSPs but they can help understand the psychological concept and further base of this research.

Ji hao and Jin Yan [6] studied two different engineering teams with around 56 and 67 members in each. It focuses on how team coordination and team longevity affects team performance. Team performance in the study refers to engineering teamwork efficiency. The study concludes that that team longevity does play part in the relationship between team coordination and team performance. In particular, if team longevity is high, then team members can coordinate better and it ends up with an overall rise in team performance and vice-versa. This leads us to the following question: could this correlation exist in the area of OSSPs?

Yulin Fang and Derrick Neufeld [16] studied phpMyAdmin OSSP. They used the theory of legitimate peripheral participation (LPP) and concluded that the process of acting knowledgeably and being identified within the community are two major factors of a high team member longevity. The study also suggests that these kinds of participants mainly engage themselves in helping others conceptually or give a contribution to improve the code that leads to improved productivity. The study focuses on only one OSSP and also uses LLP theory to conclude. Via a different approach, there is a possibility of new findings in the area of how this high longevity team member participation is affecting the development environment.

Another study [5] performed under American Psychological Association on the joint task force and humanitarian airlift mission force was aimed to see the effect of communication overhead on team cognition, which mean a team/team member's ability to perform a certain task via sharing, storing or retrieving crucial knowledge. The study suggests that if there is a need to share or exchange the knowledge between the team members to perform a task then, there will be communication between team members and it can end up in communication overhead and that affects performance negatively. Finally, the authors report that if a team's need of transferring knowledge could be lessened by having mutual awareness, then communication overhead will be less and it will end up in better performance. Although there is no direct evidence to support this claim, it opens the gate for further research.

Studies related to OSSPs such as [2] [3] [4] suggest that there are communication networks, social networks, and social groups that get formed and exist in OSSPs. Robertsa J., Hann IH., Slaughter S. [10] focuses on the mailing list of Apache HTTP server OSSP and concluded that there are communication groups and subgroups. [4] also concludes that there are sub-communities in self-organized OSSPs and the collaborative behavior of these communities mainly focuses on the technical discussion. It uses email social networks to get the findings. [2] suggests social networks in Apache HTTP server OSSPs email list using a directed graph with In-degree and Out-degree. These findings lead to further study of the effect of these social networks in the development environment and the basis for studying the communication overhead by using social networks.

In summary, there are research gaps in terms of the correlation between communication overhead, team member longevity, and productivity, especially in the domain of OSSPs. There is research regarding the result that social networks exist in OSSPs development environment. So, further study related to different variables that could affect the environment will add value to the research field.

4 Methodology

This section includes the description of the study design. To get answers to the research questions and to achieve thesis goals, data gathering, data cleaning and analytical/statistical procedures have been conducted. In the data gathering process, data from two data sources email list and Jira issue report are collected. The unmasking process has been performed to get the users for which we have both the data. After cleaning and filtering these datasets, communication overhead, team member longevity, and productivity have been calculated and analyzed. More details on the steps are described further in the section. Figure 1 shows an overview of the steps. Repository with the datasets and scripts used during this section is available on a request to the author because of the privacy concern.

4.1 Research questions and hypothesis

Two research questions have been formulated to address our research goal which is to research the relationship between communication overhead and productivity also the relationship between team member longevity and productivity**.**

RQ1: What is the relationship between communication overhead and productivity in OSSPs?

Hypothesis:

H1: There is a negative correlation between communication overhead and productivity. [5]

H2: There is a negative correlation between the number of developers with communication overhead and the productivity of sprints.

Procedure: The approach followed in the study to answer this research question is to first calculate metrics (section 4.4.1) for the two mentioned variables statistically and finding the correlation between them. For that first communication, overhead is calculated using the number of emails received or sent during the sprints, where sprints are categorized in three (HIGH, AVERAGE, and LOW) categories based on velocity. In addition to these number of developers having communication overhead and who are also contributing to the sprints calculated. After getting this data information further statistical testing of both the defined hypothesis is done using Mann-Whitney U-Test to compare different categories of the sprints and Kendall's tau correlation test. Results derived from the tests are discussed in detail in the result section of the study.

RQ2: What is the relationship between team member longevity and productivity in OSSPs?

Hypothesis:

H3: There is a positive correlation between team member longevity and productivity. [6]

Procedure: The approach followed in the study to answer this research question is to first calculate two mentioned concepts statistically and finding the correlation between them. For that first team, member longevity is calculated using the number of sprints the user is giving service, where sprints are categorized in three (HIGH, AVERAGE, and LOW) productive categories. After getting this data information further statistical testing of both the defined hypothesis is done using Mann-Whitney U-Test to compare different categories of the sprints and Kendall's tau correlation test.

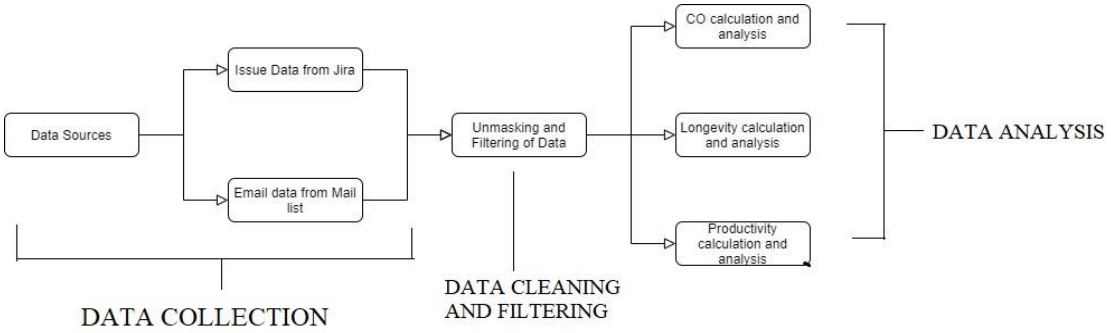


Figure 1. Overview of steps involved during the study

4.2 Data collection

As part of data collection for the research, an initial dataset consisting of OSSPs was used during this study. The initial dataset contained information for 72 open-source projects. Out of them, 22 projects were selected since they have all the information needed like issue reports with sprints data and story points. Another data set the study considers is an e-mail list of the projects. The websites of these 22 projects were inspected manually to find email list server information. Only two projects had publically available mail servers.
At last these two OSSPs MESOS, FAB is selected to be discussed and analyzed further. The study analyses two main data sources: issue reports and emails collected from the projects' mailing list. Detailed information for the selected projects is given in table 1.

Table 1. Detailed OSSPs information

Name	**Description**	**Jira URL**	**Git URL**	**Mail list URL**
MESOS	It is a cluster management program for distributed systems	https://issues.apache.org/	https://github.com/apache/mesos	https://mail-archives.apache.org/mod_mbox/mesos-dev/
FAB	An enterprise solution to build, deploy and run distributed block chains	https://jira.hyperledger.org/	https://github.com/hyperledger/fabric	https://lists.hyperledger.org/g/fabric

Email Data Collection in detail

For the apache Project (MESOS), email list data has been gathered using a python script. This script is getting all apache project's email list data into the MySQL database which is then modified to get data only for the MESOS project. Email data downloaded to MYSQL, ER diagram [17] structure is described in Appendix II, figure 9. Using a script Appendix II, figure 10, necessary data was exported as a .csv file to proceed further. [18]

Exported “email_data.csv “contains fields:

MessageId: Unique Id for the email message

Date: Date of the message sent or received

Email: Email Id of sender or receiver

Name: Full name of sender or receiver

Username: Username of sender or receiver

TypeOfRecipient: type of the email message interaction (has values: FROM or TO or CC)

For hyperledger projects, that is FAB email list data fetched from their server by downloading .zip files. This .zip file contains .mbox file for the project-specific mail list. Later converted this .mbox file to .csv file. This .csv file contains self-explanatory fields as mentioned:

Date Time, From Name, From Email, To Name, To Email, Subject, Message Id, Body Text,

Body HTML.

Using pandas python data analysis library and Excel software same fields as MESOS email data .csv file generated for FAB project.

JIRA Issue report collection

JIRA issue reports are publically available for the projects mentioned in *table 1*. Using the data extractor tool on the Jira server issue reports are obtained. Out of the data sets of two projects, the MESOS dataset is used previously in researches like [7] [19] [20]. Obtained datasets contain three .csv data files: issues dataset, changelog dataset, and Jira projects information dataset. From these files two types of datasets were prepared for the two OSSPs: Sprint dataset and Issues data set. (using python pandas library and Excel).

Issues data set initially contained fields like: *assignee.name, components, created,creator.name, description, fixVersions, issuetype.name, issuetype.subtask, priority.name, reporter.name, resolution.description, resolution.name, resolutiondate, status.id, status.name, status.statusCategory.name, summary, updated, versions, watches.watchCount, key, storypoints, project, sprint*

Once the data was collected, the next step in the process was to clean the data and an unmasking process to recognize unique users from both the data sources. Finally, analytical processes to calculate three variables of interest were performed. These variables are (1) communication overhead, (2) productivity, and (3) team member longevity. Communication overhead was calculated using the total number of sent and received emails (directed graph as visual representation), productivity was using velocity metric (calculated using story points) and team member longevity was calculated using developers’ activity on communication channel (email). A detailed description of every step in the procedure is discussed further.

4.3 Filtering and cleaning the data

After getting data from both the mailing lists and the JIRA issue tracker, a data cleaning and filtering process performed. In this process, data that are not relatable also data that are incomplete or irrelevant were removed from the dataset.

Email data filtering and cleaning

From email data .csv files obtained from the data collection phase have been filtered and cleaned. First, as part of the cleaning process converted .mbox file of the FAB project to a .csv file using MBOX to CSV converter tool. Fixed data fields of these .csv files to match with MESOS using Excel. After that from MESOS email data file, null values are removed. Also, rows that have a username field empty or numeric undesired values are removed. Email data with no date specified were removed. Emails that are auto-generated by common users or irrelevant users were removed. E.g. (hyperledger TCS). For FAB emails for the years 2016, 2019, and 2020 were removed because sprint data were not available for these years for the Project.

Issue data filtering and cleaning

From the initial issue data set fields that are important for the analysis were filtered out. Fields for the final issue data set is:

Assignee.name: username to whom issue was assigned

Created: date field on which issue was created

Key: key ID of the issue

Storypoint: storypoint for the issue

Project: Project to which issue belongs (MESOS, FAB)

From issue data for all the projects necessary fields as mentioned below were filtered out. *Assignee.name, Created, Key, Storypoint,* Project

These field names were renamed as *username, created_date, key, storypoints, and project.*

Issues with a username not specified or if the username is null that is removed. Issues with the wrong format of date or null value with a date were removed. Issues with story point 0 or null were removed also. As we need sprint data for our further data analysis, issues are removed that does not have sprints or project to which it belongs.

Using sprint field from the datasets sprint dataset for each of the projects was created. Each of these sprint datasets has *name,start_date, end_date* attributes. These data sets were corrected manually to avoid overlapping of the sprints as some of there had no start date specified. In these cases, the start date was considered as the next day of the sprint end date.

These fields are filtered from the *sprint* data filed from the issue data set*:*

Name: Name of the sprint

Start Date: start date for the sprint

End Date: end date for the sprint

Unmasking process

On the issue data set and email list data set unmasking process was performed. During this process, users with similar Id/usernames from issue data and email data were identified. First usernames were collected by doing merging of issue-data and email data on the username field. Still some usernames were common but not identified in the process. Manual searching in Excel was also done to identify these unique users. For example, if the username is apaul and email address adam.paul@xyz.com for email data then different usernames identified as apaul, adam.paul, AdamPaul, adpaul etc.

After these unmasking processes on all three Project issue and email data sets username .csv file was created. Issue data and email data sets were then filtered as per username which is common in both and taken into consideration for further research.

Detailed information of dataset considered after performing these data filtering and cleaning, unmasking for all the three projects is given in table 3. Information of the data set before performing these steps is also presented in table 2.

Table 2 data before performing filtering and cleaning processes

Project	**# Issues**	**# Sprints**	**# Users (Issues)**	**Year (Issues)**	**Year (Emails)**	**# Emails**	**# Users (Email)**
MESOS	2304	62	74	2014 - 2016	2014-2016	111958	1031
FAB	13058	137	445	2016- 2020	2016-2021	9722	1308

After applying data cleaning and filtering steps, from table 2 this thesis research will consider datasets mentioned in table 3. Here, *year(Issues)* and *year(Emails)* from table 2 are represented as the *year* in table 3 by considering years for which both Issues and Emails data are available. Similarly, *Users(Issues)* and *Users(Emails)* from table 2 are represented as *users* in table 3 by considering the number of users who are presented in both Issues and Emails data.

Table 3 data after performing filtering and cleaning processes

Project	**# Issues**	**# Sprints**	**Year**	**# Emails**	**# Users**
MESOS	2003	47	2014-2016	50022	56
FAB	742	19	2016-2019	1940	102

4.4 Data Analysis

The analysis process will include getting statistical values and visualization for our three variables communication overhead, Productivity, and Team member longevity. The first part of this section will include metrics that are considered to present these concepts statistically. Also as the goal of the thesis is to conclude the relationship between Communication overhead, Productivity, and Team member longevity after defining metrics, we will check the effect of it's on each other by processing defined metrics. The later part of the section will describe the process that is considered to study the correlation of defined metrics for the thesis.

4.4.1 Metrics

Communication Overhead

Communication overhead is the time that one spends during communication with team members other than doing assigned work [12]. In the domain of open source software projects (OSSPs) as studies show that social networks do exist [2] [3] [4]. These social networks and groups are formed based on the contributor's day-to-day communication with other team members. Thus, in the research, we consider contributor's communication with each other over email lists as a measure for communication overhead.

Three types of communication overhead were calculated:

CO1 – number of emails sent or received during a sprint (number of emails sent or received by a developer). CO1 is calculated as CO2 + CO3

CO2 – number of emails sent during a sprint (number of emails sent by a developer)

CO3 – number of emails received during a sprint (number of emails received by a developer)

A directed graph has been used to visualize these communication overheads for selected projects. In this directed graph nodes will represent developers or contributors. As directed graph's edges (in-degree and out-degree) will represent communication going on between the developers in different communication channels which here are email conversations. These edges will have weights that represent the number of the email exchange. For a node(developer) total in-degree weight is the number of emails received by that developer and out-degree weight is the number of emails sent by a developer. Example of these graph illustrated in figure 2. For Dev1 CO2 is 7 CO3 is 0 thus CO1 is 0.

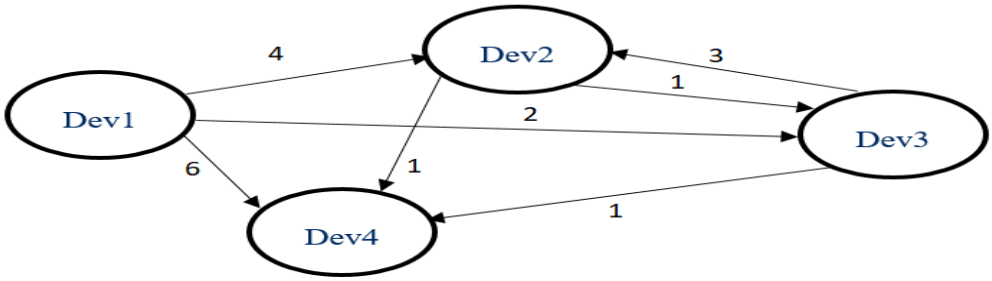


Figure 2. Directed graph for CO

Productivity

There are many arguments about having a proper metric to calculate or statistically represent productivity in the area of software development as well as in OSSPs. [14] [15] In some previous studies productivity measures considered are velocity, focus factor, work capacity, etc. Here, we are considering velocity as a metric to represent productivity. Other research that has considered velocity as a metric as a measure for productivity are [7] [19] [21].

Velocity is nothing but the rate of progress in the given time period. Velocity in software development can be defined as the total amount of work done during one iteration or sprint in the case of agile software development. Therefore, velocity is calculated as the sum of story points for the issues with the status "closed" or "done" or "resolved" as work is done for a sprint or iteration.

Productivity = velocity (total # of story points completed in a specific timeframe (sprint))

Team member longevity

Team member longevity as mentioned earlier is measured by a developer's service throughout the sprint via email conversations. To calculate team member longevity email datasets, issue datasets, and sprint datasets are used. Developer longevity is referred to as the number of sprints for which the developer was actively contributing via solving issues of the sprint or taking participation in email conversations.

Team member longevity = number of sprints for which team member is contributing

After calculating team member longevity number of developers with high team member longevity was obtained using detecting outliers. This process is discussed in detail below section.

4.4.2 Analysis Process

This section is to describe the processes followed in order to identify the effect of previously defined metrics (communication overhead, productivity, team member longevity) on each other. There are researches saws that factors like new team members, old team members, turnover have a relationship with productivity [7]. Here, we considered the number of developers with communication overhead in a sprint and the number of developers with high longevity in a sprint to study their effect on sprint productivity.

To study above mentioned scenarios, we categorize the sprints as follows:

Categorization of sprints according to their velocity

Sprint categorization has done based on velocity (total number of the issue story points). Each sprint was classified as HIGH, AVERAGE, or LOW by using two different statistical methods:

High: a group of sprints with a high value of velocity

Average: a group of sprints with an average value of velocity

Low: a group of sprints with a low value of velocity

1. Quantile categorisation:
   This method considers percentile quantile to assign the category to a sprint based on the velocity value into categories.
   Low category sprint velocity value <= 0.25 quantile values
   Average category sprint velocity = 0.25 to 0.50 quantile values
   High category sprint velocity >= 0.75 quantile values

2. Statistical Process Control: [22]

   This method uses variation of values to see the change in the process over time. There are studies in the software engineering domain that have used the method to identify any changes in performance. In this study, this method is used to categorized data as it uses Upper Control Limit, Control Limit, and Lower Control Limit to do the same. This study considers x-bar chart as our sample group can be considered as 1 to 10. As per the eq. 1. Control limit presented as x-bar which is the median of each velocity. The sample size-specific anti-biasing constant d2 = 1.128 and MR-bar is the median moving range [7].

$$UCL\bar{x} = \bar{x} + 3\,\frac{\overline{MR}}{d2},\ CL\bar{x} = \bar{x}, LCL\bar{x} = \bar{x} - 3\,\frac{\overline{MR}}{d2}$$ e.q. 1

Sprints with velocity values greater than or equal to Upper Control Limit are considered as High productive sprints. Sprint with velocity value less than or equal to Lower Control Limit and remaining are considered as Low productive and Average productive sprints respectively.

Outlier detection

To identify the number of developers with communication overhead, two outlier detection techniques were applied to the values of CO1, CO2, CO3 obtained from the mailing list data.

1. IQR (Tukey's fences):
   The interquartile range is a widely used method to detect outliers. [23] For grouped email dataset based on sprint and user name, IQR outlier detection method as represented in e.q. 2. applied on metrics (communication overhead) CO1, CO2 and CO3. Detected outlier accepted as a developer with communication overhead.

   IQR = Q3 – Q1
   Outlier: [Q1 – k*IQR, Q3 + k*IQR], here k = 1.5 e.q. 2.

2. Statistical Process Control/ Control Limit Chart: [22]
   Considered the same approach as discussed earlier in the sprint categorization section and assigned values that are out of limits (Upper Control Limit and Lower Control Limit) as outliers (developers with communication overhead).

As a result of applying these techniques, we are also able to identify the number of developers with communication overhead for each sprint. Since there are three different metrics for communication overhead (CO1, CO2, and CO3), two different techniques to detect outliers (IQR, SPC/CL), we were able to define:

Devs_with_co1_IQR: number of developers with CO, calculated based on CO1 and IQR
Devs_with_co2_IQR: number of developers with CO, calculated based on CO2 and IQR
Devs_with_co3_IQR: number of developers with CO, calculated based on CO3 and IQR
Devs_with_co1_CL: number of developers with CO, calculated based on CO1 and CL
Devs_with_co2_CL: number of developers with CO, calculated based on CO2 and CL
Devs_with_co3_CL: number of developers with CO, calculated based on CO3 and CL

As longevity is the number of sprints for which the developer was contributing, to check its effect on sprint productivity we considered sprint categorization as specified earlier in the section. Also, to get developers with high longevity we had used the same outlier detection technique for developer longevity parameter and we had identified parameters:

Devs_with_high_long_IQR: number of developers with high longevity calculated using IQR

Devs_with_high_long_CL: number of developers with high longevity calculated using CL

All these numbers were later compared for different categories (HIGH, AVERAGE, LOW) of the sprints.

5 Results

This section describes the results obtained after applying the method described in chapter 3. Descriptive statistics, as well as different visualisations using statistical charts, are presented. First communication overhead and productivity relation discussed based on the method followed.

5.1 Communication Overhead and Productivity (RQ1)

Communication overhead is nothing but email conversations of developers for a project. Three types of communication overhead have been considered. CO1 is total emails sent and received, CO2 is total emails received and CO3 is the total email sent by a contributor. Following the approach from a related study communication network here is presented using a directed graph [2]. Appendix II, figure 11 (MESOS), and figure 14 (FAB) show these social networks in the form of a directed graph. As the graphs suggest, some developers are having more emails to read as well as to give responses to than other developers in both the projects.

Table 4. MESOS descriptive result for all the sprints

	velocity	co1	co2	co3	devs_with _co1_IQR	devs_with _co2_IQR	devs_with _co3_IQR	devs_with _co1_CL	devs_with _co2_CL	devs_with _co3_CL
count	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00
mean	155.24	349.44	185.22	164.22	2.07	1.71	0.42	3.02	2.33	1.36
std	179.44	272.07	106.53	175.31	2.20	1.62	0.69	2.69	1.89	1.60
min	6.00	39.00	31.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
0.25	49.00	109.00	96.00	8.00	0.00	0.00	0.00	1.00	1.00	0.00
0.50	102.00	260.00	172.00	41.00	1.00	1.00	0.00	3.00	2.00	0.00
0.75	176.00	534.00	244.00	290.00	4.00	3.00	1.00	5.00	3.00	3.00
max	800.00	968.00	450.00	623.00	7.00	6.00	2.00	9.00	6.00	5.00

Table 4. describes statistical values like mean, standard deviation, minimum values, maximum values, values for 25% of the quartile and 50%(median) and 70% also. The table shows all these values for all considered sprints for the MESOS project. Total sprints after filtering and cleaning datasets were 47, which refers as a count in the table. Table columns are co1(total emails during 47 sprints), co2(total emails sent during 47 sprints), co3(total emails received during 47 sprints). Also, the table shows that the minimum value for velocity is 6. Where in some sprints there is only one email has been received. While there are sprints that have developers with having 0 communication overhead. Maximum velocity for the sprint is 800 while some of the sprints have a max of 9 developers with co1 if calculated using control limit charts.

Table 5. shows the values for all considered sprints for the FAB project. Total sprints after filtering and cleaning datasets were 19, which refers as a count in the table. Table columns are co1(total emails during 19 sprints), co2(total emails sent during 19 sprints), co3(total emails received during 19 sprints). Columns “devs_with_x_IQR” and “devs_with_x_CL” refer to the count of developers with a different type of communication overhead (x = CO1, CO2, CO3) calculated with two different earlier mentioned outlier detection methods (IQR and CL). Also, the table shows that the minimum value for velocity is 1. Where in some sprints there is only one email has been received. While there are sprints that have developers with having 0 communication overhead. Maximum velocity for the sprint is 130 while

some of the sprints have a max of 5 developers with co1 if calculated using control limit charts as well as using IQR.

Table 5. FAB descriptive result for all the sprints

	velocity	co1	co2	co3	devs_with _co1_IQR	devs_with _co2_IQR	devs_with _co3_IQR	devs_with _co1_CL	devs_with _co2_CL	devs_with _co3_CL
count	19.00	19.00	19.00	19.00	19.00	19.00	19.00	19.00	19.00	19.00
mean	38.58	34.26	21.58	12.68	1.00	0.84	0.63	1.11	1.16	0.63
std	38.28	27.01	16.01	11.29	1.33	1.01	1.07	1.41	1.42	1.07
min	1.00	8.00	7.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
0.25	14.50	16.00	10.50	5.50	0.00	0.00	0.00	0.00	0.00	0.00
0.50	25.00	23.00	15.00	9.00	1.00	1.00	0.00	1.00	1.00	0.00
0.75	46.00	41.50	24.00	18.00	1.50	1.00	1.00	2.00	2.00	1.00
max	130.00	107.00	61.00	46.00	5.00	3.00	4.00	5.00	5.00	4.00

Table 6. MESOS categorized sprints using control limits and its communication overhead

	category CL	count	mean	std	min	0.25	0.50	0.75	max
velocity	Average	40.00	100.00	70.77	6.00	40.00	91.50	137.75	282.00
	High	5.00	597.20	175.19	354.00	502.00	619.00	711.00	800.00
	Low	-	-	-	-	-	-	-	-
co1	Average	40.00	332.95	262.94	39.00	107.50	233.50	479.50	968.00
	High	5.00	481.40	340.08	84.00	166.00	604.00	687.00	866.00
	Low	-	-	-	-	-	-	-	-
co2	Average	40.00	176.05	98.98	31.00	95.00	167.00	232.75	432.00
	High	5.00	258.60	147.43	78.00	153.00	274.00	338.00	450.00
	Low	-	-	-	-	-	-	-	-
co3	Average	40.00	156.90	172.95	1.00	7.75	34.50	287.00	623.00
	High	5.00	222.80	203.95	6.00	13.00	266.00	413.00	416.00
	Low	-	-	-	-	-	-	-	-

*Data for the LOW productive sprint are not available for control chart sprint categorization as lover limit for velocity is 0.

If data for all the 47 (MESOS) and 19 (FAB) sprints are considered for comparison there is not much statistical variation can be found here as there is not much difference for mean, median (50%), and standard deviation values. This will be tested further during verifying the hypotheses.

To analyze data categorically and not the whole bunch of 47 or 19 sprints together, we had categorized sprints in three HIGH, AVERAGE, and LOW categories as per their velocity. The categorization had done using quartile and control charts methods. Table 6 and Table 7 shows statistical descriptions of co1, co2, and co3 for both of the sprint categorization methods (quantile and control chart) for MESOS while Table 8 and Table 9 represents the same for the FAB project.

Table 7. MESOS categorized sprints using quartile and its communication overhead

	category_quantile	count	mean	std	min	0.25	0.50	0.75	max
velocity	Average	22.00	103.05	33.71	49.00	83.75	101.00	116.00	170.00
	High	12.00	373.92	225.56	176.00	202.50	262.00	531.25	800.00
	Low	11.00	21.09	12.41	6.00	12.00	20.00	28.50	40.00
co1	Average	22.00	312.05	268.84	65.00	102.25	201.00	473.00	968.00
	High	12.00	438.58	319.40	80.00	129.25	485.50	637.50	941.00
	Low	11.00	327.00	220.31	39.00	119.50	345.00	466.00	679.00
co2	Average	22.00	177.09	96.29	63.00	97.50	176.00	228.75	432.00
	High	12.00	233.58	130.38	78.00	122.75	230.00	317.75	450.00
	Low	11.00	148.73	85.72	31.00	89.50	158.00	203.50	304.00
co3	Average	22.00	134.95	181.43	1.00	7.00	25.00	258.00	623.00
	High	12.00	205.00	196.23	1.00	7.50	209.50	384.50	512.00
	Low	11.00	178.27	140.15	7.00	25.50	187.00	292.50	375.00

As per the data and categorization techniques used and from the tables that are presented for both the projects, it is conclusive that most of the sprints are falling into the AVERAGE category of the categorization. Where second place is taken by HIGH category sprints though the difference is not significant with LOW category. Table 6 and Table 9 show that no LOW productive sprint for the control chart method as it takes LCL as 0 which in this case means that there are no sprints with 0 velocity value for any of the examined projects sprint.

Minimum and Maximum values are similar as described in Table 4 and Table 5 for co1, co2, and co3 as well as for the velocity of the sprints for both the projects. The mean values are an average of all the values and also sensitive for extreme values of the sample. The median is less sensitive for extreme values. All three medians, mean and std. values are taken into consideration. However, after looking into median (50%) values for co1, co2, and co3 of three different categories for both the projects from it is conclusive that there is rarely a significant difference for co1, co2, co3 - median, mean, and std. values for average and high production sprints. That is more specific for high and average productive sprints for received emails during the sprint, this difference also might be affecting the overall difference.

Table 8. FAB categorized sprints using quartile and its communication overhead

	category_quantile	count	mean	std	min	0.25	0.50	0.75	max
velocity	Average	10.00	29.20	12.04	15.00	18.50	25.50	38.75	47.00
	High	4.00	102.25	29.92	60.00	94.50	109.50	117.25	130.00
	Low	5.00	6.40	5.32	1.00	2.00	6.00	9.00	14.00
co1	Average	10.00	38.10	31.33	10.00	13.25	29.00	51.50	107.00
	High	4.00	26.25	9.46	16.00	20.50	25.50	31.25	38.00
	Low	5.00	33.00	30.07	8.00	16.00	23.00	34.00	84.00
co2	Average	10.00	22.80	17.83	7.00	9.50	17.50	30.00	61.00
	High	4.00	17.25	6.08	10.00	13.75	17.50	21.00	24.00
	Low	5.00	22.60	19.67	7.00	12.00	14.00	24.00	56.00
co3	Average	10.00	15.30	13.61	3.00	5.25	11.50	21.50	46.00
	High	4.00	9.00	3.56	6.00	6.75	8.00	10.25	14.00
	Low	5.00	10.40	10.50	1.00	4.00	9.00	10.00	28.00

Table 9. FAB categorized sprints using control limits and its communication overhead

	Category_CL	count	mean	std	min	0.25	0.50	0.75	max
velocity	Average	16.00	24.00	17.39	1.00	12.75	20.00	38.25	60.00
	High	3.00	116.33	12.34	106.00	109.50	113.00	121.50	130.00
	Low	-	-	-	-	-	-	-	-
co1	Average	16.00	35.13	29.36	8.00	15.00	21.00	46.50	107.00
	High	3.00	29.67	8.02	22.00	25.50	29.00	33.50	38.00
	Low	-	-	-	-	-	-	-	-
co2	Average	16.00	21.94	17.43	7.00	9.75	13.00	26.00	61.00
	High	3.00	19.67	4.51	15.00	17.50	20.00	22.00	24.00
	Low	-	-	-	-	-	-	-	-
co3	Average	16.00	13.19	12.23	1.00	4.75	8.00	20.50	46.00
	High	3.00	10.00	3.61	7.00	8.00	9.00	11.50	14.00
	Low	-	-	-	-	-	-	-	-

*Data for the LOW productive sprint are not available for control chart sprint categorization as lover limit for velocity is 0.

RQ1 (What is the relationship between communication overhead and productivity in OSSPs?)

The aim of research question one is to find the relationship between communication overhead and productivity.

H1: There is a negative correlation between communication overhead and productivity.

Hypothesis one assumes that if communication overhead increased in the OSSPs environment then productivity would be decreasing. This is suggested in the study [5]. In this hypothesis, first, we plot communication overhead (co1, co2, and co3) vs. velocity in a box plot graph. Figure 3 (MESOS) and Figure 4 (FAB) show graphs for both (quartile and control limit) categorized sprint velocity. In the graphs, the first row (3 subgraphs) represents quartile sprint categorization while the second row (4,5,6 subgraphs) represents control charts sprint categorization based on the velocity of the sprint. The X-axis represents a different type of communication overhead (CO1, CO2, CO3). Different colors are used to represent the categories (HIGH, AVERAGE, and LOW) of the sprints.

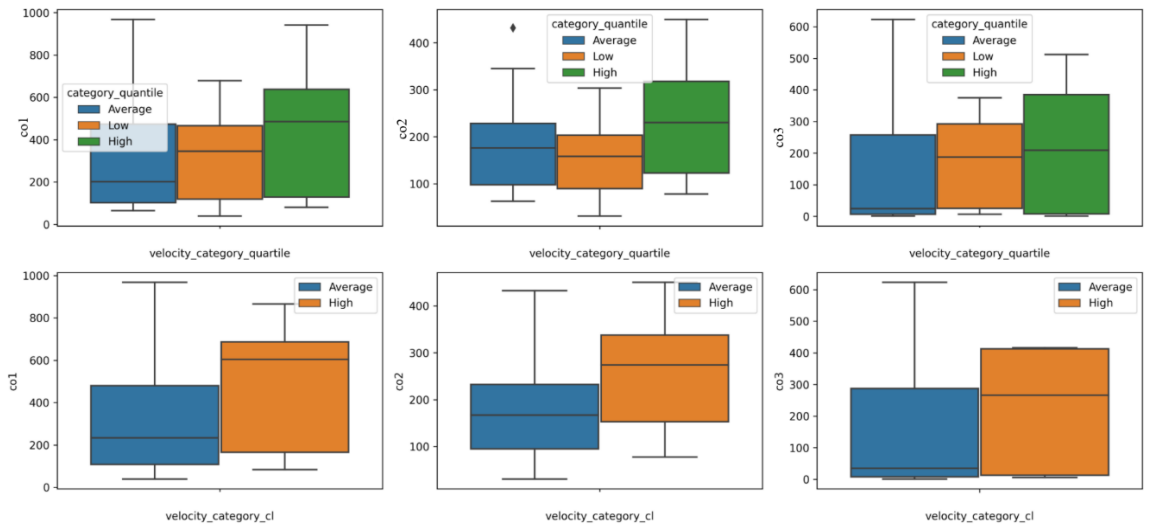


Figure 3. MESOS communication overhead in categorized sprints(quartile and CL)

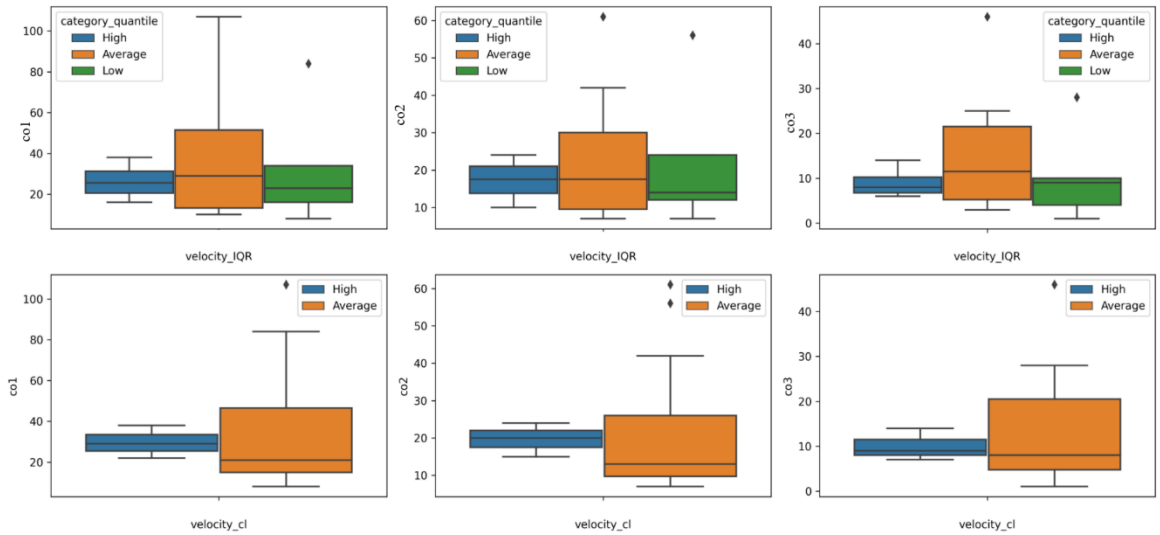


Figure 4. FAB communication overhead in categorized sprints(quartile and CL)

As per Figure 3 for MESOS project co3 total emails received during the sprint (6th subplot), there is having a noticeable median variance between high and average productive sprints while overall co1 (total emails sent and received – 4th subplot) do not show much difference when sprints are categorized using control chart categorization method. Regarding the graphs where sprints are categorized using quartiles (first row of the subplot in the graph), there is no significant difference in communication overhead (co1, co2, c03) between LOW, AVERAGE, and HIGH productive sprints.

As for project FAB, Figure 4 suggests that there is not a visually noticeable difference in the communication overhead (co1, co2, and co3) between HIGH, AVERAGE, and LOW productive sprints and communication overhead when using any (quartile and control chart limit) of the sprints categorization methods.

For both the charts developers participating in high productive sprints have received more emails than developers in average productive sprints. Overall after visualizing the data the information is not enough to conclude hypothesis 1; thus further statistical test has been performed.

Table 10 presents results of the Mann-Whitney U-test for control chart sprint categorization of MESOS and FAB projects. This test checks if there is any statistical difference in co1, co2, and co3 between AVERAGE and HIGH productivity sprints for the projects. As P-value for the considered parameters is less than 0.05 (Alpha constant value) the results show that there is no difference in any type of communication overhead (number of emails sent and received) during the different categories of the sprints.

Table 10. MESOS and FAB H1 Mann-Whitney U-test results for **control chart sprint categorization** (HIGH and AVERAGE productive sprints)

Project	CO	AVG, LOW productive sprint		AVG, HIGH productive sprint		HIGH, LOW productive sprint		Is there any diff.?
		W-value	P-value	W-value	P-value	W-value	P-value	
MESOS	co1	-	-	70.000	0.296	-	-	No
	co2	-	-	65.000	0.213	-	-	No
	co3	-	-	79.500	0.470	-	-	No
FAB	co1	-	-	15.000	0.876	-	-	No
	co2	-	-	13.500	0.696	-	-	No
	co3	-	-	15.500	0.938	-	-	No

*Data for the LOW productive sprint are not available for control chart sprint categorization as lover limit for velocity is 0.

Table 11 shows the Mann-Whitney U-test results for categorized sprints (using quartile) and communication overhead. This test was aimed to know the difference of communication overhead (sent and received emails) for HIGH, AVERAGE, and LOW productive sprints in both MESOS and FAB projects. Again the results show that there is no difference for the considered groups as P-value for them is almost 1 and not less than 0.05.

After these tests, Kendall's tau correlation test was performed to check the overall correlation between productivity(velocity) and communication overhead. Table 12 shows the results of the test for both projects. After analyzing the result as P-value it suggests that there is no correlation between the considered parameters.

As per the discussion, we can say that for performed methods and considered datasets for the projects MESOS and FAB, hypothesis one is not confirmed.

Table 11. MESOS and FAB H1 Mann-Whitney U-test results for **quartile sprint categorization** (HIGH, LOW, and AVERAGE productive sprints)

Project	CO	AVG, LOW productive sprint		AVG, HIGH productive sprint		HIGH, LOW productive sprint		Is there any diff.?
		W-value	P-value	W-value	P-value	W-value	P-value	
MESOS	co1	117.000	0.895	103.000	0.309	83.000	0.316	No
	co2	140.500	0.468	96.000	0.204	90.000	0.151	No
	co3	88.000	0.214	112.500	0.493	66.000	1.000	No
	co1	28.000	0.759	22.000	0.832	10.500	1.000	No

FAB	co2	24.000	0.951	20.500	1.000	10.500	1.000	No
	co3	29.000	0.668	22.000	0.831	10.500	1.000	No

Table 12 MESOS and FAB H1 Kendall tau correlation test Results (velocity and communication overhead)

Project	Parameter	B-value	P-value	H1
MESOS	co1	0.024	0.814	No Confirmed
	co2	0.094	0.363	No Confirmed
	co3	-0.069	0.506	No Confirmed
FAB	co1	0.197	0.463	No Confirmed
	co2	0.197	0.463	No Confirmed
	co3	0.197	0.463	No Confirmed

H2: There is a negative correlation between the number of developers with communication overhead and the productivity of sprints.

For further analysis on research question one and to know the effect of communication overhead on productivity, another parameter that has been taken into consideration is the number of developers with communication overhead during a specific sprint category. Hypothesis two is to check the relation of productivity of each category (HIGH, AVERAGE, and LOW) with the number of developers with all three defined communications overhead (co1, co2, and co3) box plots as presented in Appendix II, figure 12,13 (MESOS) and figure (15,16). The first row in both the box plots represents the number of developers identified using the IQR (Turkey's fences) method where the second row of both the figures represents the number of developers with different communication overhead identified using control limits.

Appendix II, figure 12 represents the box plots where categorization of the sprints is done based on control limits. As per the graphs presented in Appendix II, figure 12, it is noticeable that both average and high productive sprints do not have difference in the number of developers with communication overhead (CO1, CO2, CO3). Similarly, Appendix II, figure 13 shows no difference in number of developers with communication overhead between HIGH, AVERAGE and LOW productive sprint (here categorization is done using quantile) for MESOS.

Appendix II, figure 15 and 16 represents the box plots where categorization of the sprints is done based on control charts and quantile for project FAB. It also shows similar results which are discussed for MESOS project box plots. For FAB project also there is no difference in number of developers with communication overhead between categorised sprints. Similar as hypothesis one, to conclude results more precisely for hypothesis two also statistical tests has been conducted.

Table 13 shows the result of the Mann-Whitney U-test to test if there is any difference in the number of developers with communication overhead between AVERAGE and HIGH productive sprints for the MESOS and FAB projects. The results show that there is not much

difference as the p-value is not less than 0.05. Here, sprints are categorized using control chart limits.

Table 13. MESOS and FAB H2 Mann-Whitney U-test results for **control chart sprint categorization** (HIGH and AVERAGE productive sprints)

Project	Devs with CO	AVG, LOW productive sprint		AVG, HIGH productive sprint		HIGH, LOW productive sprint		Is there any diff. ?
		W-value	P-value	W-value	P-value	W-value	P-value	
MESOS	Devs with co1(IQR)	-	-	64.00	0.186	-	-	No
	Devs with co2(IQR)	-	-	74.50	0.355	-	-	No
	Devs with co3(IQR)	-	-	88.00	0.610	-	-	No
	Devs with co1(CL)	-	-	47.00	0.055	-	-	No
	Devs with co2(CL)	-	-	51.50	0.078	-	-	No
	Devs with co3(CL)	-	-	75.00	0.339	-	-	No
FAB	Devs with co1(IQR)	-	-	14.00	0.737	-	-	No
	Devs with co2(IQR)	-	-	7.500	0.152	-	-	No
	Devs with co3(IQR)	-	-	20.00	0.603	-	-	No
	Devs with co1(CL)	-	-	15.50	0.933	-	-	No
	Devs with co2(CL)	-	-	11.50	0.458	-	-	No
	Devs with co3(CL)	-	-	20.00	0.603	-	-	No

*Data for the LOW productive sprint are not available for control chart sprint categorization as lover limit for velocity is 0.

Table 14 shows the results of the Mann-Whitney U-test for quartile sprint categorization using velocity. The results show that there not much difference in the number of developers with communication overhead between categorized sprint groups like HIGH, AVERAGE, and LOW as the P-value is not less than 0.05 for any category and parameter.

Table 14. MESOS and FAB H2 Mann-Whitney U-test results for **quartile sprint categorization** (HIGH, LOW, and AVERAGE productive sprints)

Project	Devs with CO	AVG, LOW productive sprint		AVG, HIGH productive sprint		HIGH, LOW productive sprint		Is there any dif
		W-value	P-value	W-value	P-value	W-value	P-value	
	Devs with co1(IQR)	85.500	0.166	91.00	0.126	75.50	0.575	No

MESOS	Devs with co2(IQR)	102.00	0.467	104.5	0.317	74.00	0.638	No
	Devs with co3(IQR)	106.00	0.478	107.0	0.270	71.00	0.750	No
	Devs with co1(CL)	109.00	0.656	93.00	0.159	83.00	0.305	No
	Devs with co2(CL)	118.00	0.922	87.50	0.107	85.50	0.236	No
	Devs with co3(CL)	107.50	0.585	108.5	0.360	75.50	0.563	No
FAB	Devs with co1(IQR)	24.000	0.948	21.00	0.939	9.000	0.896	No
	Devs with co2(IQR)	20.500	0.591	14.50	0.444	11.00	0.896	No
	Devs with co3(IQR)	34.000	0.238	26.50	0.344	10.50	1.000	No
	Devs with co1(CL)	25.500	1.000	22.50	0.761	9.000	0.896	No
	Devs with co2(CL)	25.000	1.000	19.00	0.940	11.00	0.896	No
	Devs with co3(CL)	34.000	0.238	26.50	0.344	10.50	1.000	No

Table 15 MESOS and FAB H2 Kendall tau correlation test Results (velocity and # of devs. with communication overhead)

Project	Devs. with CO	B-value	P-value	H2
MESOS	Devs with co1(IQR)	-0.026	0.814	No Confirmed
	Devs with co2(IQR)	-0.029	0.794	No Confirmed
	Devs with co3(IQR)	0.000	1.000	No Confirmed
	Devs with co1(CL)	0.057	0.599	No Confirmed
	Devs with co2(CL)	0.099	0.370	No Confirmed
	Devs with co3(CL)	0.011	0.924	No Confirmed
FAB	Devs with co1(IQR)	0.032	0.911	No Confirmed
	Devs with co2(IQR)	0.160	0.575	No Confirmed
	Devs with co3(IQR)	0.356	0.242	No Confirmed
	Devs with co1(CL)	0.032	0.911	No Confirmed
	Devs with co2(CL)	0.160	0.575	No Confirmed
	Devs with co3(CL)	0.356	0.242	No Confirmed

Similar to hypothesis one for hypothesis two also Kendall tau correlation test was performed and Table 15 presents the results for the same. The results show that there is no correlation between productivity (velocity) and the number of developers with communication overhead participating in the projects FAB and MESOS.

As per the discussion of the results it is clear that hypotheses one and two for the MESOS and FAB open-source software projects are not confirmed for the data and methods that are selected for the study.

5.2 Developer longevity and Productivity (RQ2)

RQ2 (What is the relationship between team member longevity and productivity in OSSPs?)

Research question 2 is to study if productivity and team member longevity are correlated or not. At first developer's contribution has been calculated and longevity has been identified and categorized. After categorizing data of developer's longevity, the number of developers with a high longevity and sprint productivity plots were produced as part of visual analysis. The number of developers with high longevity refers to developers who have participated in more sprints or developers having high participation as a form of more email conversations during sprints.

H3: There is a positive correlation between team member longevity and productivity.

Hypothesis 3 suggests that if team member longevity increases then productivity will also increase. To verify this hypothesis, we have defined a parameter "number of developers with high longevity' using this parameter and categorized productivity (HIGH, AVERAGE, and LOW) figure 5 and figure 6 are box plots that are presented based on the datasets analyzed. They are for MESOS and FAB projects respectively.

Presented graphs suggest that there is a small between high and average productive sprints. High productive sprints have more developers with high longevity than the average productive sprints. While low productive sprints show mixed results.

To know the results more in detail and to test the hypothesis, similarly, as we did earlier Mann-Whitney U-test and Kendall tau correlation test are performed and results are discussed further in the section.

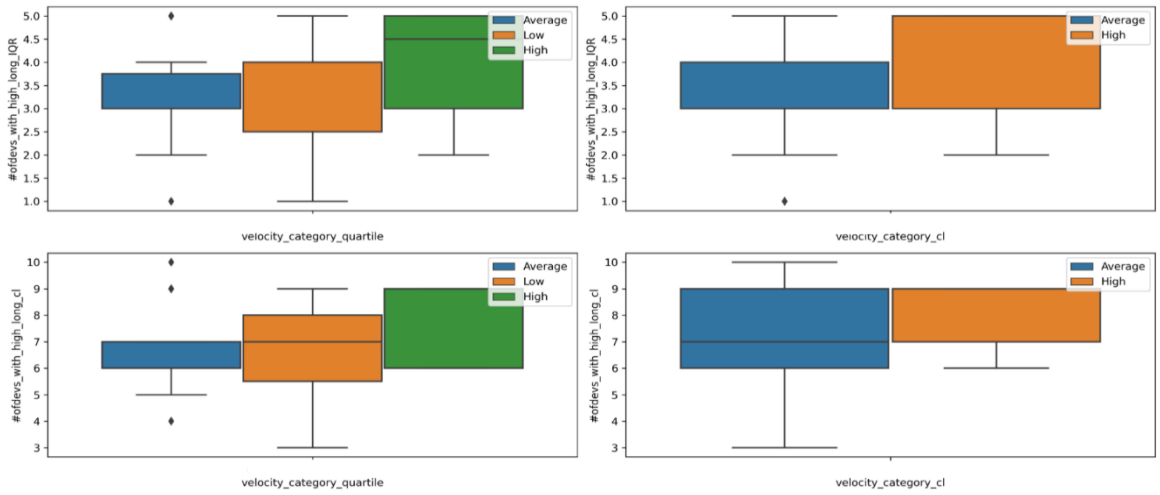


Figure 5. MESOS number of developers with high longevity in categorized sprints (quartile and CL)

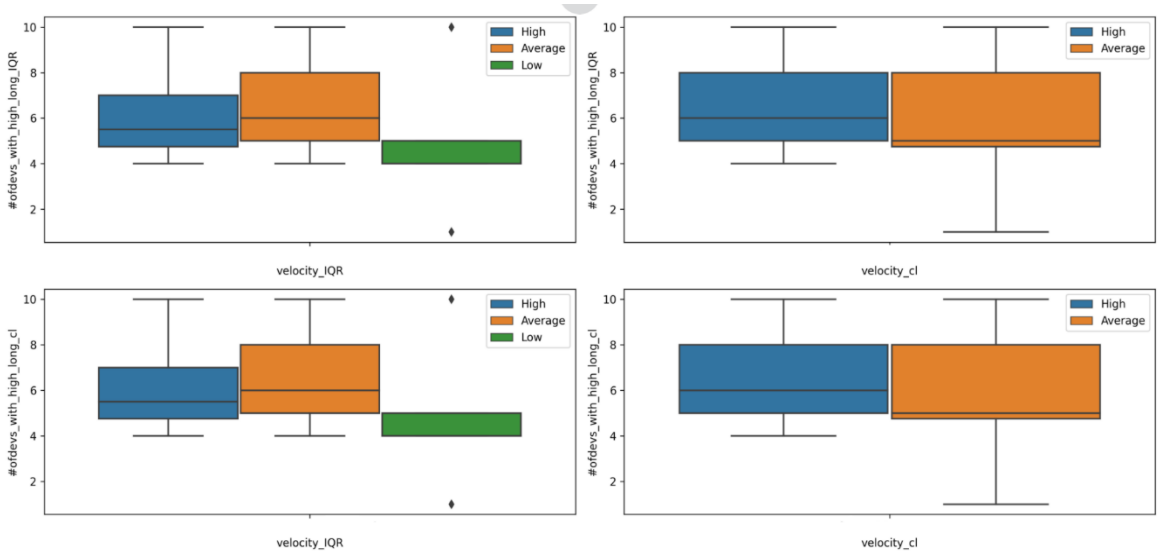


Figure 6. FAB number of developers with high longevity in categorized sprints (quartile and CL)

Table 16 presents the results of the Mann-Whitney U-test for MESOS and FAB projects. This test was conducted to check whether there is a difference in the number of developers with high longevity between AVERAGE and HIGH productive sprints. The results show not much difference is there as P-value is not less than 0.05.

Table 16 MESOS and FAB Mann-Whitney U-test Results for **control chart sprint categorization** *(velocity (AVERAGE, HIGH) and # of devs. with communication overhead)*

Project	**Devs with high Longevity**	**AVG, LOW productive sprint**		**AVG, HIGH productive sprint**		**HIGH, LOW productive sprint**		**Is there any diff.?**
		W-value	**P-value**	**W-value**	**P-value**	**W-value**	**P-value**	
MESOS	Devs with high log. (CL)	-	-	54.000	0.092	-	-	No
	Devs with high log. (IQR)	-	-	77.500	0.355	-	-	No
FAB	Devs with high log. (CL)	-	-	20.500	0.732	-	-	No
	Devs with high log. (IQR)	-	-	20.500	0.732	-	-	No

*Data for the LOW productive sprint are not available for control chart sprint categorization as lover limit for velocity is 0.

Table 17 shows the result of the Mann-Whitney U-test for both projects. Here, the test was performed on the dataset where sprints are categorized using quartile. The test aims to verify the difference in the number of developers with high longevity in HIGH, AVERAGE, and

LOW productive sprints. As the P-value for the test is for every tested parameter and category is not less than 0.05 it is concluded that there is not much difference.

Table 17 MESOS and FAB Mann-Whitney U-test Results for **quartile sprint categorization** *(velocity and number of devs. with communication overhead)*

Project	**Devs with high Longevity**	**AVG, LOW productive sprint**		**AVG, HIGH productive sprint**		**HIGH, LOW productive sprint**		**Is there any diff.?**
		W-value	**P-value**	**W-value**	**P-value**	**W-value**	**P-value**	
MESOS	Devs with high log.(CL)	127.500	0.815	92.500	0.147	86.500	0.208	No
	Devs with high log.(IQR)	116.000	0.846	107.000	0.304	74.500	0.544	No
FAB	Devs with high log.(CL)	36.500	0.168	22.000	0.828	14.000	0.379	No
	Devs with high log.(IQR)	36.500	0.168	22.000	0.828	14.000	0.379	No

Table 18 describes the result of the Kendall tau correlation test for both projects. The test is performed to test the correlation of velocity and the number of developers with high longevity for the whole dataset. Results of the test show that there is no correlation between velocity and Team members with high longevity.

Table 18 MESOS and FAB H3 Kendall tau correlation test Results (velocity and number of devs. with communication overhead)

Project	**Devs with high Longevity**	**B-value**	**P-value**	**H1**
MESOS	Devs with high log. (CL)	0.162	0.214	No Confirmed
	Devs with high log.(IQR)	0.158	0.260	No Confirmed
FAB	Devs with high log.(CL)	-0.179	0.397	No Confirmed
	Devs with high log.(IQR)	-0.179	0.397	No Confirmed

As per the discussion of the results in this section, it is clear that hypothesis three for the MESOS and FAB open-source software projects are not confirmed for the data and methods that are selected for the study. More discussion on the results is presented in the further section of the study.

6 Discussion

An in-depth discussion of the results presented in the earlier section is presented in this section. Also, research questions and defined hypothesis has discussed based on the results.

6.1 Interpretation of the Results

The main goal of the research is to find the correlation between communication overhead, team member longevity, and productivity in OSSPs that follows agile software development. The aim behind looking for the correlation is to know the effect of these concepts in the whole OSSPs environment. The results show that there is no correlation between the concepts found for the studied projects.

Research question one was to verify the negative correlation between communication overhead and productivity. This has been done by defining two hypotheses, one is considering the effect of emails that have been sent or received (communication overhead) during the sprints over the sprint productivity. The second hypothesis was identifying the number of developers with communication overhead that is participating during the sprint. After testing these hypotheses, the conclusion is there is no correlation between these variables.

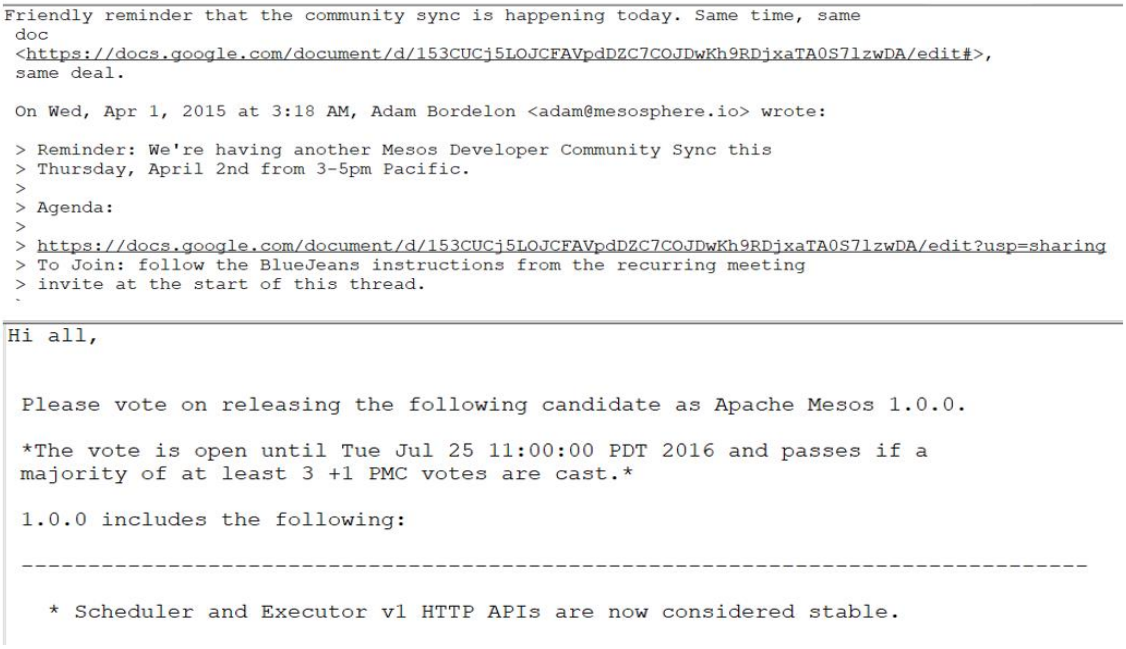
Research question two was to verify the direct correlation between team member longevity and productivity. This had been answered by following a similar method that has been followed for the RQ1. The third hypothesis defined was about identify the number of the developers with high team member longevity contributing during the more productive sprints. After conducting the corresponding statistical tests, the conclusion for the research question was there is no correlation between the concepts that have been considered for RQ2.
Analysis and results for the third research question are contradictory for the study like [6]. Here, the reason for the negative result could be the conduction of a very exploratory data analysis process. We have considered the longevity of the users as a constant with respective of time which is could be improved in the further studies.

The reason for getting negative results for the defined hypotheses could be the lost data during the data cleaning and filtering. The process that could have affected the result is presented in Tables 2 and 3. Table 2 presents the dataset before the cleaning and filtering process while Table 3 dataset after the mentioned processes. Here from the tables for Fabric project issue number before the process was 13000 which dropped to 700. This large drop is because of the lack of data for the sprints in which the issues belong also some of the issues have null sprints and null user names that have been filtered out. Similarly, for email data for MESOS out of 111958 emails less than half emails 50022 has been filtered while for FAB out of 9722 only 1940 emails have been filtered. The reason here for the big difference is, as a lot of the emails were belonging to the usernames that did pass the unmasking process. For these usernames, issue data was not there.

The results of the study contradict some researches that show that communication overhead has a negative effect on productivity these studies are not in OSSPs domain [5]. While another study in OSSPs suggests that type communication or collaboration network does affect productivity [24]. The result of the study [24] shows that if the communication network is centralized then it has a positive effect on productivity and in bigger team decentralization of communication network lead to a negative effect on productivity. The study considers bug rate as a measure of productivity. Formerly mentioned study [5] considers domains (joint task force and humanitarian airlift mission force) which are not related to OSSPs while the latter [24] is considering different parameter (bug rate) as a measure of productivity. As

these studies are considering different contexts and measuring metrics for the study variables, it is hard to come to a firm conclusion.

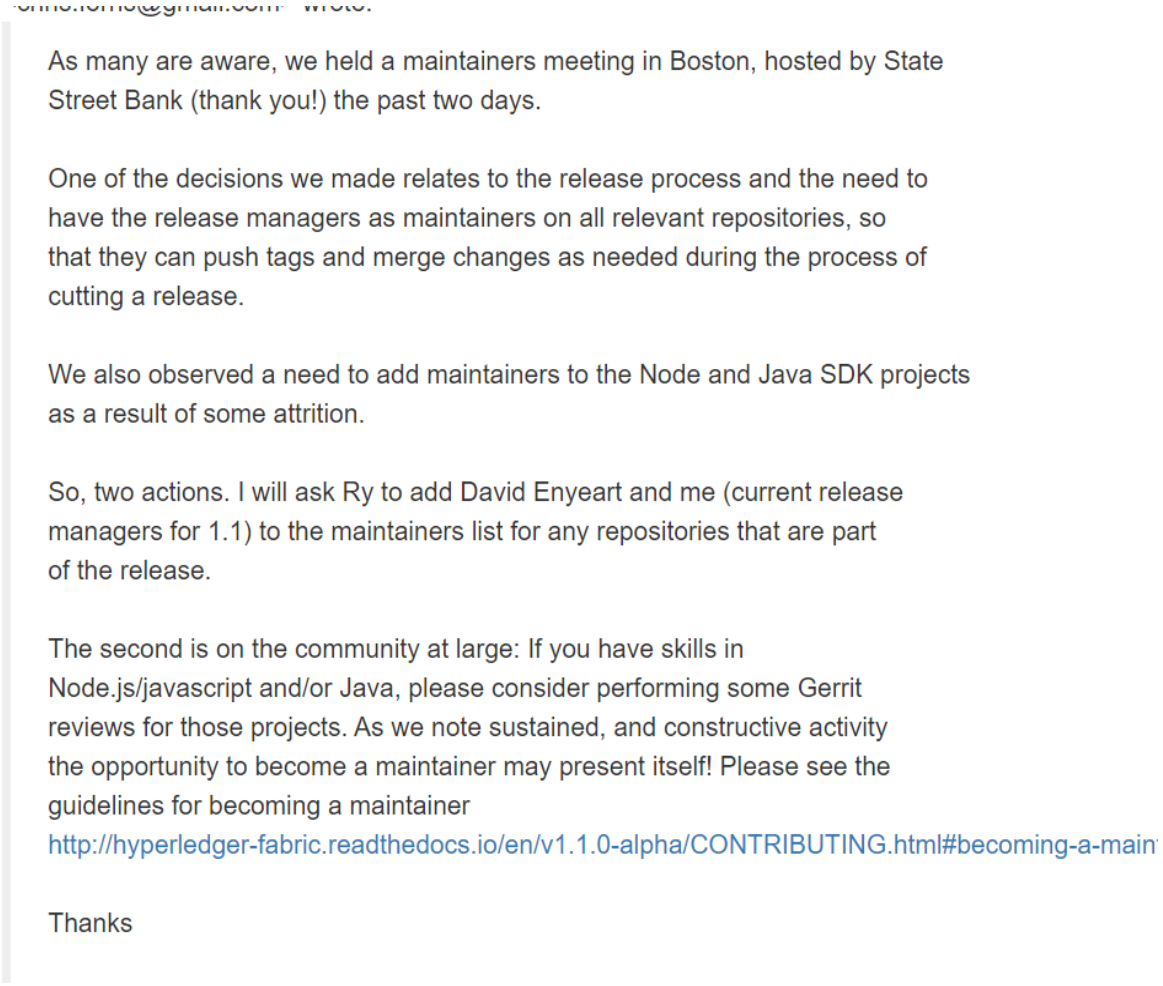
To explain the negative results, the content of the emails was also analyzed. We have analyzed 4 randomly selected emails for the users having high communication overhead that have been identified from the directed graphs of both projects. Appendix II, figure 11 (MESOS), and figure 14 (FAB). A sample of some of the emails has presented in Appendix II, figure 7 (MESOS), and figure 8 (FAB).



```
Friendly reminder that the community sync is happening today. Same time, same
 doc
 <https://docs.google.com/document/d/153CUCj5LOJCFAVpdDZC7COJDwKh9RDjxaTA0S7lzwDA/edit#>,
 same deal.

 On Wed, Apr 1, 2015 at 3:18 AM, Adam Bordelon <adam@mesosphere.io> wrote:

 > Reminder: We're having another Mesos Developer Community Sync this
 > Thursday, April 2nd from 3-5pm Pacific.
 >
 > Agenda:
 >
 > https://docs.google.com/document/d/153CUCj5LOJCFAVpdDZC7COJDwKh9RDjxaTA0S7lzwDA/edit?usp=sharing
 > To Join: follow the BlueJeans instructions from the recurring meeting
 > invite at the start of this thread.
```

```
Hi all,


 Please vote on releasing the following candidate as Apache Mesos 1.0.0.

 *The vote is open until Tue Jul 25 11:00:00 PDT 2016 and passes if a
 majority of at least 3 +1 PMC votes are cast.*

 1.0.0 includes the following:

 --------------------------------------------------------------------------------

   * Scheduler and Executor v1 HTTP APIs are now considered stable.
```

Figure 7 sample email communication of one of the developers with communication overhead in the MESOS project



As many are aware, we held a maintainers meeting in Boston, hosted by State Street Bank (thank you!) the past two days.

One of the decisions we made relates to the release process and the need to have the release managers as maintainers on all relevant repositories, so that they can push tags and merge changes as needed during the process of cutting a release.

We also observed a need to add maintainers to the Node and Java SDK projects as a result of some attrition.

So, two actions. I will ask Ry to add David Enyeart and me (current release managers for 1.1) to the maintainers list for any repositories that are part of the release.

The second is on the community at large: If you have skills in Node.js/javascript and/or Java, please consider performing some Gerrit reviews for those projects. As we note sustained, and constructive activity the opportunity to become a maintainer may present itself! Please see the guidelines for becoming a maintainer
http://hyperledger-fabric.readthedocs.io/en/v1.1.0-alpha/CONTRIBUTING.html#becoming-a-main

Thanks

Figure 8 sample email communication of one of the developers with communication overhead in FAB project

After doing quick searches on social networking site (LinkedIn) we came to know that email samples of the MESOS project happen to be from the user who is contributing to the project as a part of the project management chair. There were a notably large number of emails from this user are present. The content of the email is mostly about scheduling sync up or asking for contribution during the release voting process.

In addition to this, an email sample of the FAB project had been sent from the user who is contributing to the project as a member of the Technical Steering Committee, and emails from also are mostly related to invites and making other users to aware of the changes in the different processes.

After analyzing these emails, it is acceptable that the fact of not analyzing the email content has been played its part in the results. Considering the content and also analyzing the role of the user in the OSSPs organization could be helpful in the future. Deciding which type of users and email form them should be taken into consideration by analyzing the content of the emails during performing the analysis methods might change the results. Also considering other measuring metrics like bug rate, line of code, focus factor, etc for productivity could help conclude overall results. To conclude results more precisely further research is needed.

6.2 Lessons Learnt

This part of the discussion presents some insights on what are the steps of the research were challenging and required more effort.

The major challenges faced during this study were to collect the data and clean and filter the collected datasets. Here, we have collected data for email conversations and issue reports. In the data collection, the tough part was to find the OSSPs that have both email list data, as well as they, are following agile software development. From 22 OSSPs that have been taken into consideration, only 2 or 3 projects have been found which have both the datasets. In data cleaning and the filtering major part was to find the unique users for which we have both the data. Also as discussed earlier because we are following velocity as the measurement of the productivity from all issue data when we filtered data there were a lot of data that have been neglected as they were not part of sprints.

Not many researches are there to refer to specifically in the area of communication overhead and productivity. Methods that are used to measure the concepts as well as for categorization are also limited.

6.3 Limitations

It also presents limitations and research gaps that have been identified.

One of the limitations of this study is that it is only has been conducted on two OSSPs. Considering only one measurement for productivity and analysis it. This could be a reason that results are not very much reliable as there are many more measures that could have been taken into consideration [13] [14] [20]. In addition to this only email, list conversations are taken into consideration while there could be communication happening on different communication channels like Gitter, Slack, or even issue tracking tools comment section, etc. Another constraint that could have affected the results is not checking or analyzing the content and tone of the email messages.

7 Conclusion and future work

This section summarises the research study. In this study, we have examined two open-source software projects MESOS and FABRIC. We have collected datasets of email communication and issue reports for both projects. After collecting the data further analysis was performed to explore the correlation of communication overhead, team member longevity, and productivity. Velocity of a sprint was considered as a measure of productivity for the analysis. Sprint data from both the projects was categorized in three categories HIGH, AVERAGE and LOW. The categorization of sprints based on productivity (velocity) was performed using quantile and statistical process control (control chart limit) methods. Two more metrics "number of developers with communication overhead" and "number of developers with high longevity" in the given sprint categories were also studied.

Three hypotheses were defined based on the proposed research questions. To verify these hypotheses statistical tests were performed. *Mann-Whitney U-test* was carried out to recognize if there was any difference in communication overhead (sent CO3 and received emails CO2, CO1), number of developers with communication overhead, number of developers with high longevity between defined categories of the sprints. *Kendall tau correlation test* was carried out to recognize the overall correlation for the three defined variables.

The results of the statistical tests show that there was no correlation between sent or received emails, the number of developers with communication overhead (sent CO3 and received emails CO2, CO1), team member longevity, and productivity (velocity) for considered projects and methods.

The factor that might have affected the results could be the number of projects that had been considered and the measurement metrics that had defined for the concepts like communication overhead and productivity.

Limitations described in the earlier section of the thesis lead to discussion for future work. As a part of future work, the study could be extended for more OSSPs. It would be good if different communication channels were taken into consideration. Also, exploring the content of the messages transferring through the communication channels will be useful to come to a firm conclusion. Considering different parameters for measuring variables would be helpful. At last, the results will be useful for the users to up their learning curve, build social connections, or brainstorm on important ideas.

8 References

[1] S. Choudhary, C. Bogart, C. Rose and J. Herbsleb, "Using Productive Collaboration Bursts to Analyze Open Source Collaboration Effectiveness," 2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER), 2020, pp. 400-410.

[2] Christian Bird, Alex Gourley, Prem Devanbu, Michael Gertz, and Anand Swaminathan. 2006. Mining email social networks. In Proceedings of the 2006 international workshop on Mining software repositories (MSR '06). Association for Computing Machinery, New York, NY, USA, 137–143.

[3] A. Guzzi, A. Bacchelli, M. Lanza, M. Pinzger and A. van Deursen, "Communication in open source software development mailing lists," 2013 10th Working Conference on Mining Software Repositories (MSR), 2013, pp. 277-286.

[4] Christian Bird, David Pattison, Raissa D'Souza, Vladimir Filkov, and Premkumar Devanbu. 2008. Latent social structure in open source projects. In Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering (SIGSOFT '08/FSE-16). Association for Computing Machinery, New York, NY, USA, 24–35.

[5] MacMillan, J., Entin, E. E., & Serfaty, D. (2004). Communication overhead: The hidden cost of team cognition. In E. Salas & S. M. Fiore (Eds.), Team cognition: Understanding the factors that drive process and performance (p. 61–82). American Psychological Association.

[6] Ji H and Yan J (2020) How Team Structure Can Enhance Performance: Team Longevity's Moderating Effect and Team Coordination's Mediating Effect. Frontiers in Psychology.

[7] E. Scott, K. N. Charkie and D. Pfahl, "Productivity, Turnover, and Team Stability of Agile Teams in Open-Source Software Projects," 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Portoroz, Slovenia, 2020, pp. 124-131.

[8] Guo, W., Gan, C. and Wang, D. (2020), "The mobility of team members and team creativity: exploring the mediating role of team cognition", *Journal of Organizational Change Management*, Vol. 33 No. 6, pp. 1111-1122. [9] Yulin Fang & Derrick Neufeld (2009) Understanding Sustained Participation in Open Source Software Projects, Journal of Management Information Systems, 25:4, 9-50.

[10] A. Ahmed, S. Ahmad, N. Ehsan, E. Mirza, and S. Sarwar, "Agile software development: Impact on productivity and quality," presented at the 2010 IEEE International Conference on Management of Innovation & Technology, 2010, pp. 287–291.

[11] G. von Krogh, S. Spaeth and S. Haefliger, "Knowledge Reuse in Open Source Software: An Exploratory Study of 15 Open Source Projects," *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 2005, pp. 198b-198b.

[12] Josh Kaufman (2005). The Personal MBA: Master the Art of Business. New York, U.S., Penguin Random House.

[13] Goparaju Purna Sudhakar, Ayesha Farooq, Sanghamitra Patnaik(2012). Measuring productivity of software development teams.

[14] S. Downey and J. Sutherland, “Scrum metrics for hyperproductive teams: how they fly like fighter aircraft,” in 2013 46th Hawaii Int. Conf. on Syst. Sciences, pp. 4870–4878, IEEE, 2013.
[15] E. Kupiainen, M. V. M¨antyl¨a, and J. Itkonen, “Using metrics in agile and lean software development – a systematic literature review of industrial studies,” Inf. & Soft. Tech., vol. 62, pp. 143–163, 2015.

[16] Yulin Fang & Derrick Neufeld (2009) Understanding Sustained Participation in Open Source Software Projects, Journal of Management Information Systems, 25:4, 9-50.

[17] ER diagram for downloaded email data: https://github.com/bateman/MailigListStats/blob/f6056731638d9d81feeeb41e6ade95d8063dc504/db/mlstats_model_dbdesigner.png

[18] Apache email data fetch: https://github.com/collab-uniba/personality/tree/master/src/python/ml_downloader

[19] E. Scott and D. Pfahl, “Using developers’ features to estimate story points,” presented at the Proceedings of the 2018 International Conference on Software and System Process, 2018, pp. 106–110.
[20] K. Petersen, “Measuring and predicting software productivity: A systematic map and review,” Inf. & Soft. Tech., vol. 53, no. 4, pp. 317–343, 2011.
[21] M. Choetkiertikul, H. K. Dam, T. Tran, T. Pham, A. Ghose, and T. Menzies, “A Deep Learning Model for Estimating Story Points,” *IEEE Transactions on Software Engineering*, vol. 45, no. 7, pp. 637–656, Jul. 2019.
[22] W. A. Florac and A. D. Carleton, Measuring the software process: statistical process control for software process improvement. Addison-Wesley Professional, 1999.
[23] Outlier: https://en.wikipedia.org/wiki/Outlier
[24] Jermakovics A., Sillitti A., Succi G. (2013) Exploring Collaboration Networks in Open-Source Projects. In: Petrinja E., Succi G., El Ioini N., Sillitti A. (eds) Open Source Software: Quality Verification. OSS 2013. IFIP Advances in Information and Communication Technology, vol 404. Springer, Berlin, Heidelberg.

[25] L. Barton, G. Candan, T. Fritz, T. Zimmermann and G. C. Murphy, "The Sound of Software Development: Music Listening Among Software Engineers," in *IEEE Software*, vol. 37, no. 2, pp. 78-85.

[26] A. Trendowicz and J. M¨unch (2009). Factors influencing software development productivity—state-of-the-art and industrial experiences, Elsevier, Pages 185-241.

[27] Naveen Raman, Minxuan Cao, Yulia Tsvetkov, Christian Kästner, and Bogdan Vasilescu. 2020. Stress and burnout in open source: toward finding, understanding, and mitigating unhealthy interactions. In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER '20). Association for Computing Machinery, New York, NY, USA, 57–60.

[28] Robertsa J., Hann IH., Slaughter S. (2006) Communication Networks in an Open Source Software Project. In: Damiani E., Fitzgerald B., Scacchi W., Scotto M., Succi G. (eds) Open Source Systems. OSS 2006. IFIP International Federation for Information Processing, vol 203.

Appendix

I. Abbreviation

OSSPs - Open Source Software Projects

CO – Communication Overhead

MESOS – Apache Mesos

FAB – Hyperledger Fabric

RQ – Research Question

IQR – Interquartile Range

UCL – Upper Control Limit

LCL – Lower Control Limit

CL - Control Limit

Std. – Standard Deviation

II Additional tables and figures

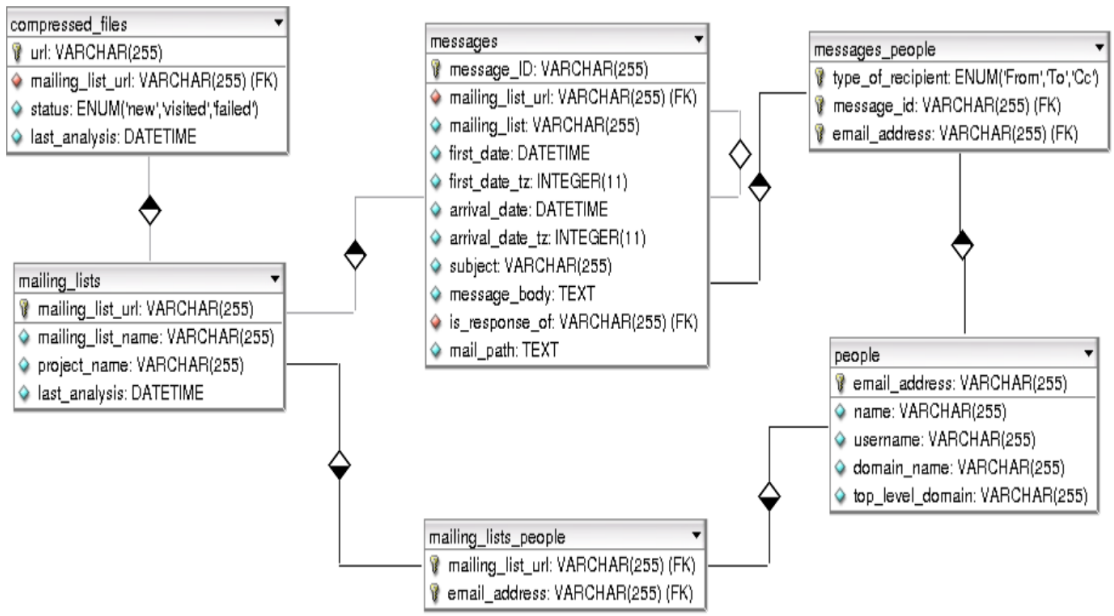


Figure 9 ER diagram for apache email list download

```
cursor.execute ("select mp.type_of_recipient, m.arrival_date, m.message_id, p.email_address, p.name, p.username from
messages_people as mp inner join messages as m on m.message_id = mp.message_id inner join people as p on mp.email_address =
p.email_address where mp.mailing_list_url IN ('http://mail-archives.apache.org/mod_mbox/mesos-dev/', 'http://mail-
archives.apache.org/mod_mbox/mesos-user/') and YEAR(m.arrival_date) in (2014, 2015, 2016)")

data = cursor.fetchall ()
d = {}
for row in data:
    if row[1] not in d:
        d[row[1]] = []
    d[row[1]].append(row[0]+"_"+row[4])
with open('email_data.csv', mode='w') as data_file:
    data_writer = csv.writer(data_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL)

    for row in data:
        data_writer.writerow([row[0], row[1], row[3], row[4], row[5]])
cursor.close ()
```

Figure 10 Script snippet to export email data as .csv

MESOS graph and plots:

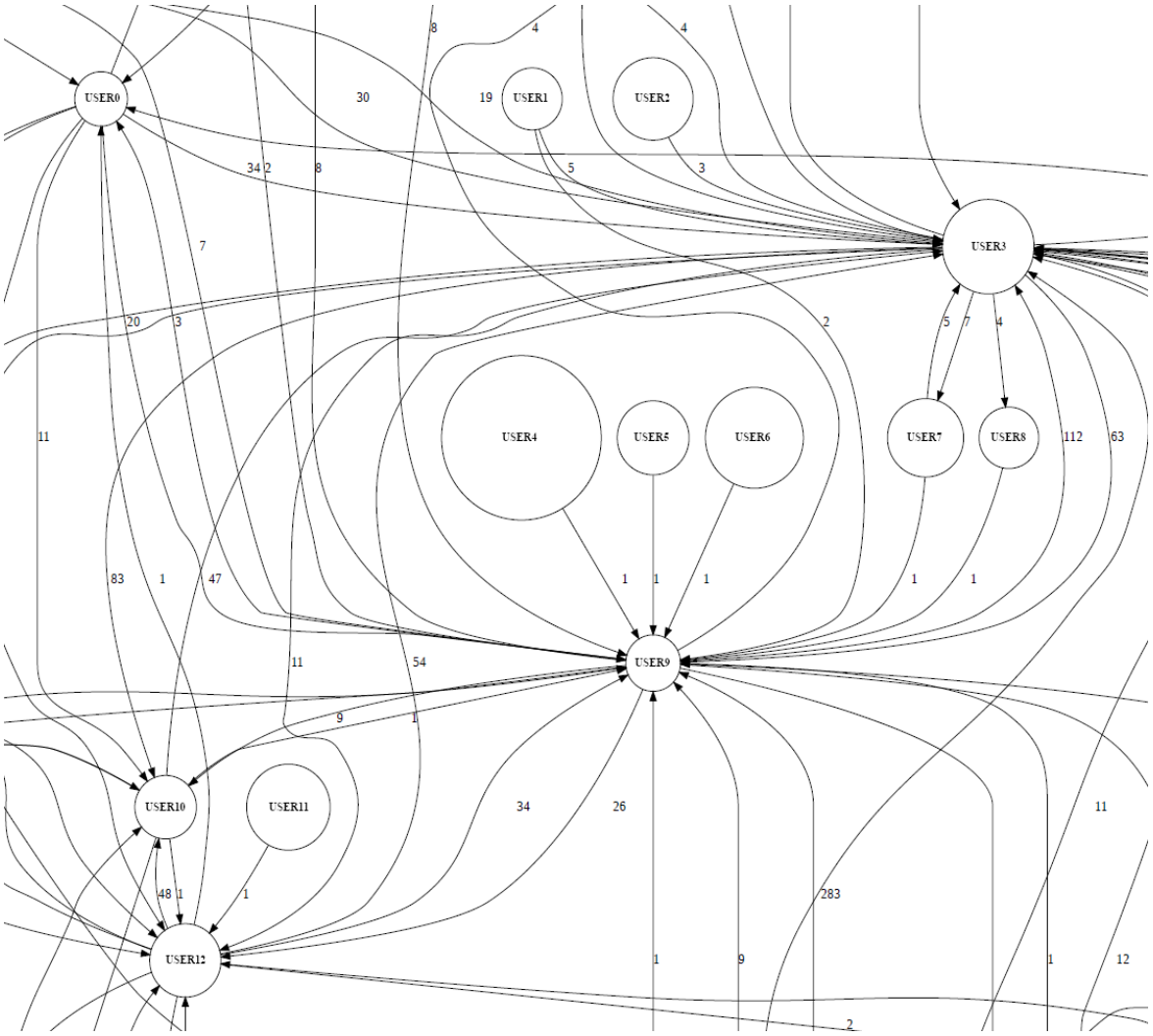


Figure 11. part of MESOS project email communication network represented using a directed graph

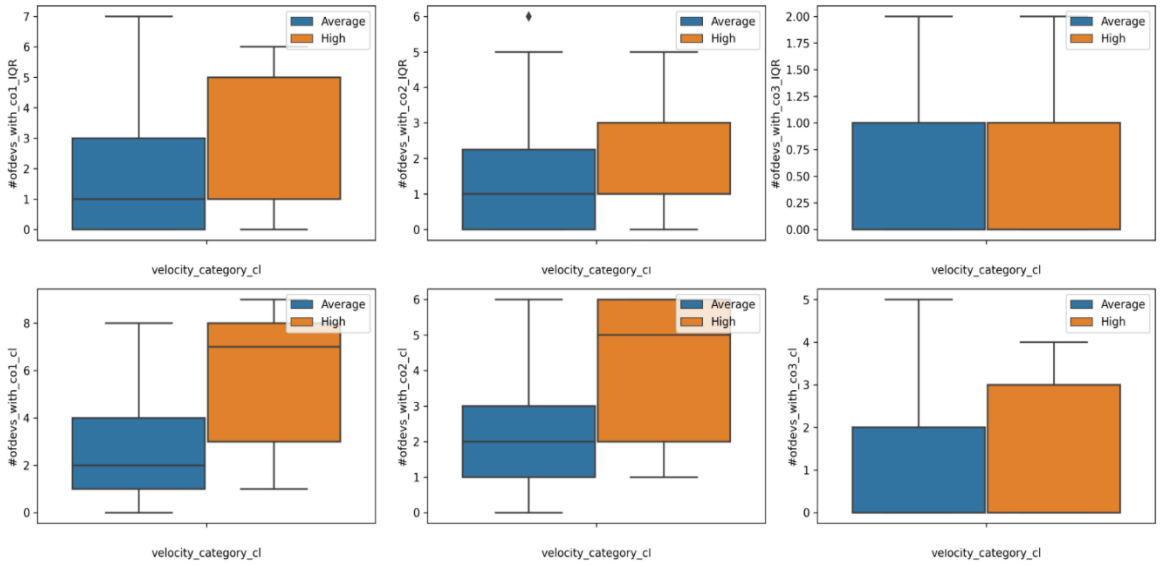


Figure 12. MESOS number of developers with communication overhead in categorized sprints(CL)

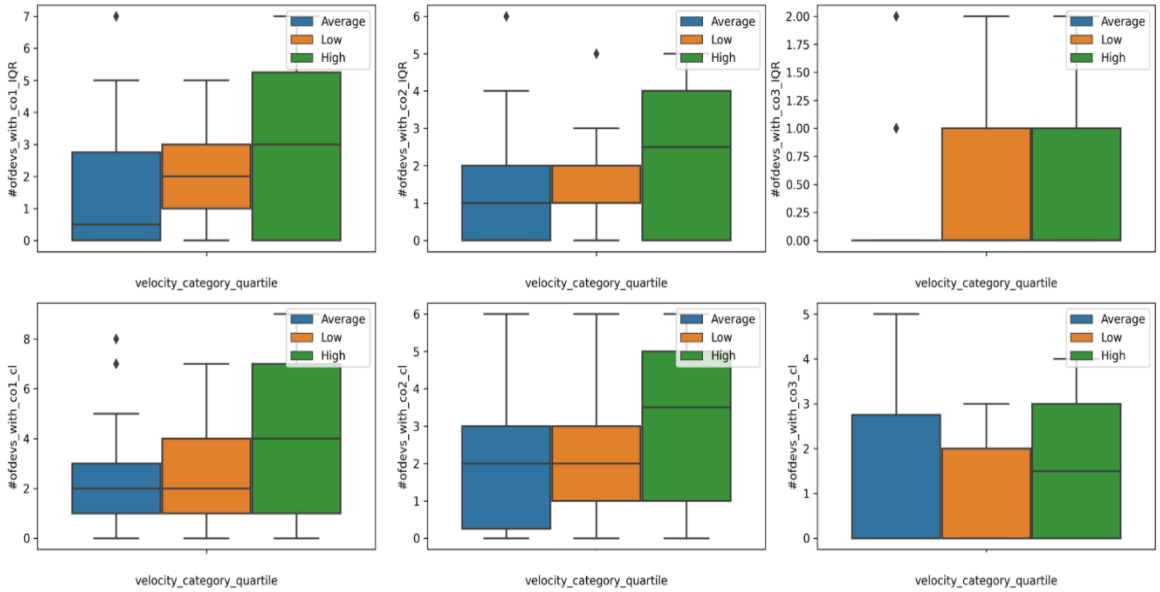


Figure 13. number of developers with communication overhead in categorized sprints(quartile)

FAB project graph and plots

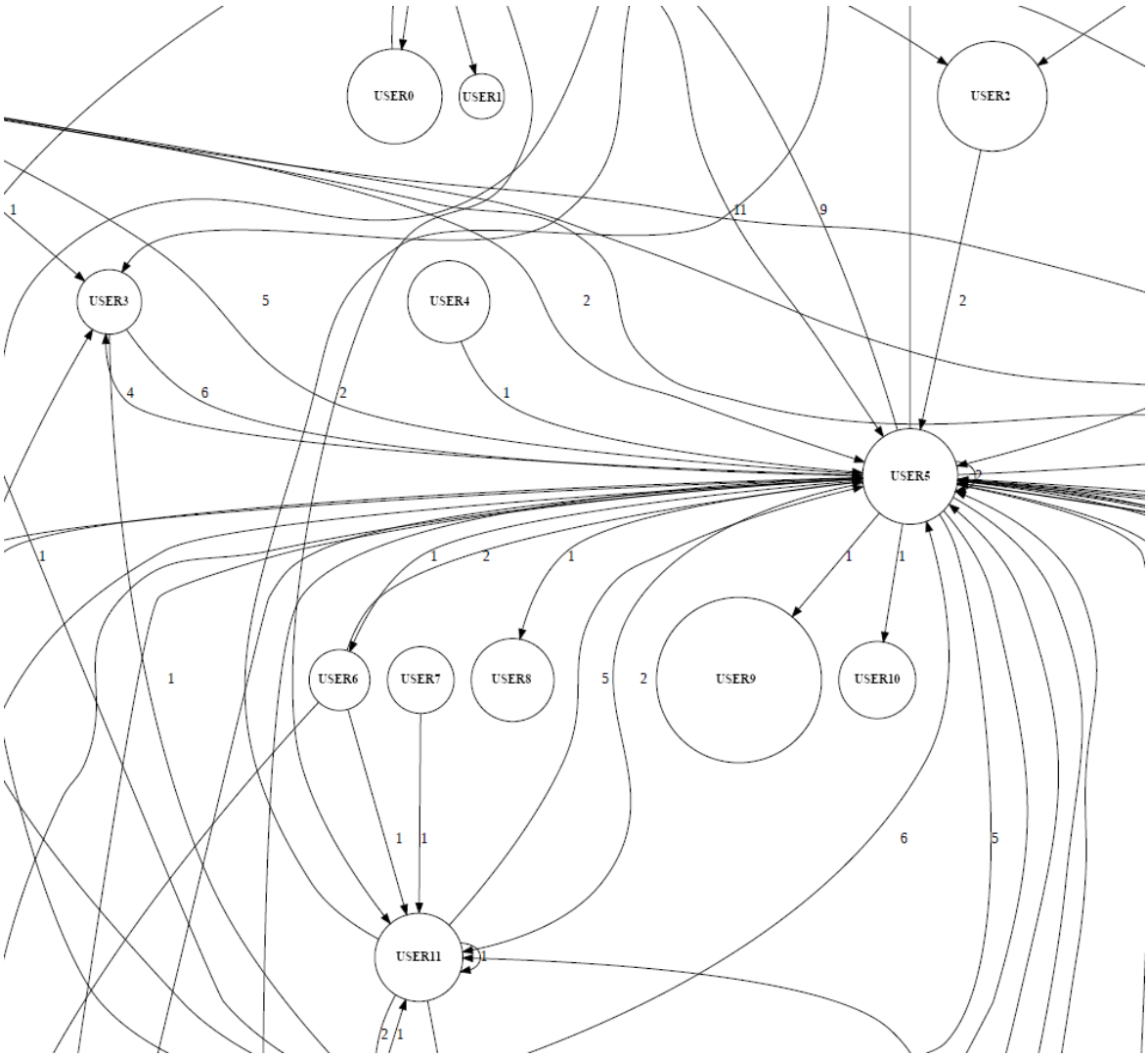


Figure 14. part of FAB project email communication network represented using a directed graph

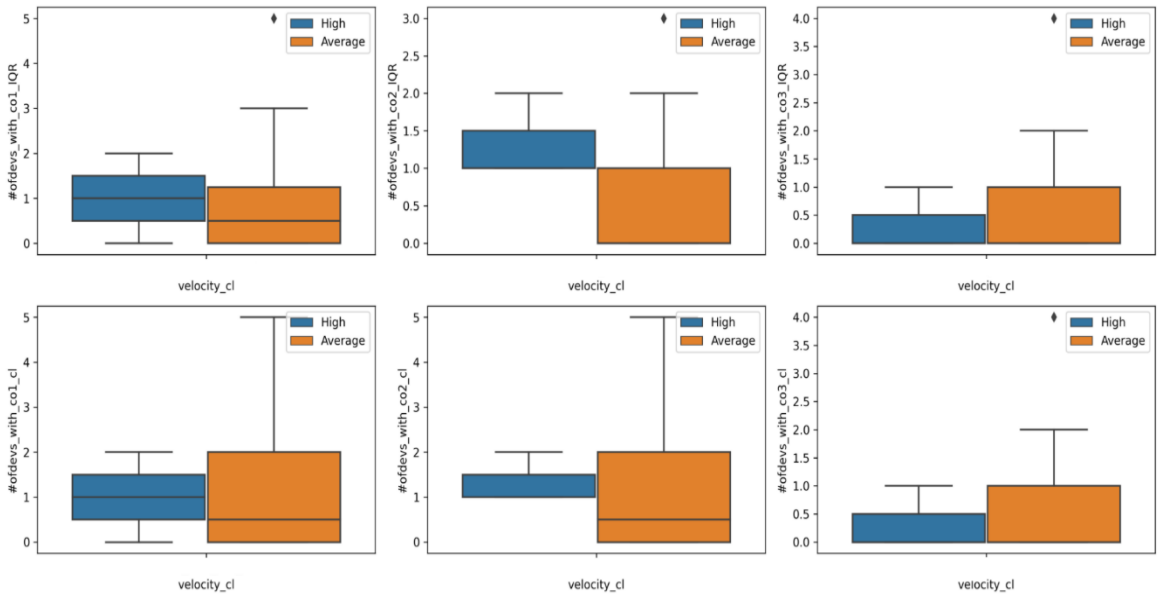


Figure 15. FAB number of developers with communication overhead in categorized sprints(CL)

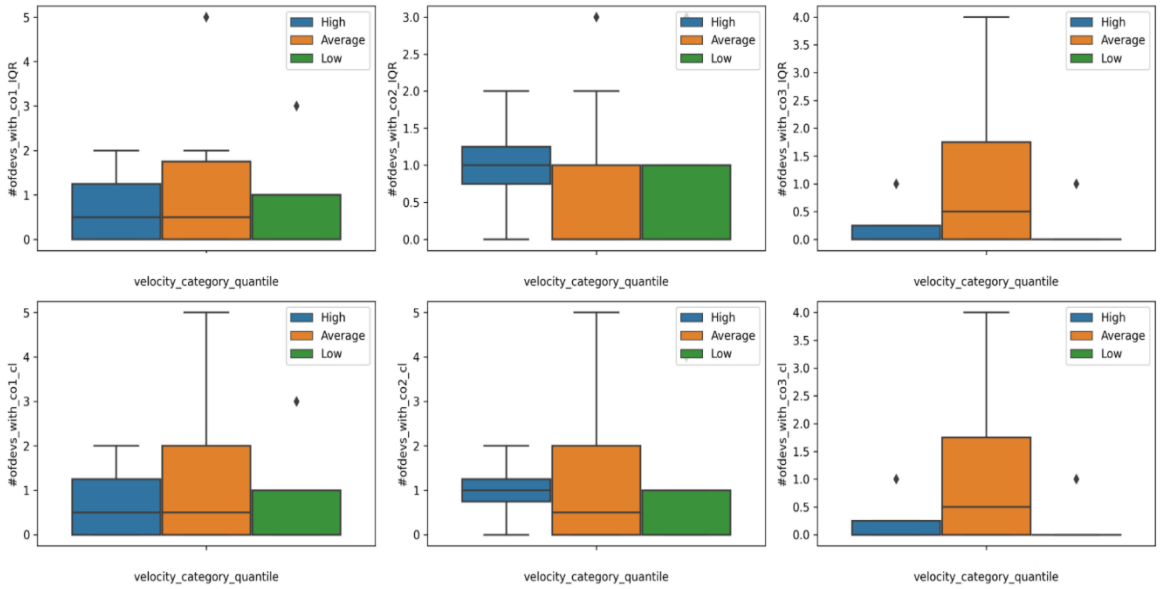


Figure 16. FAB number of developers with communication overhead in categorized sprints(quartile)

III License

Non-exclusive license to reproduce thesis and make thesis public

I, Hariti Atulkumar Sanchaniya,

1. herewith grant the University of Tartu a free permit (non-exclusive license) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Communication Overhead in Open Source Software Projects,
   supervised by **Ezequiel Scott**.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons license CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive license does not infringe on other persons' intellectual property rights or rights arising from the personal data protection legislation.

Hariti Atulkumar Sanchaniya

14/05/2021