Analyzing Distributions of Emails and Commits from OSS Contributors through Mining Software Repositories
An Exploratory Study

Mário Farias[1,2], Renato Novais[1,4], Paulo Ortins[1], Methanias Colaço[3] and Manoel Mendonça[1,4]
[1]*Federal Institute of Bahia, Salvador, Brazil*
[2]*Federal Institute of Sergipe, São Cristóvão, Brazil*
[3]*Federal University of Sergipe, São Cristóvão, Brazil*
[4]*Fraunhofer Project Center for Software and Systems Engineering, Bahia, Brazil*

Keywords: Software Repository Mining, Open Source Contributions, Experimental Software Engineering, Software Visualization, Preferred Representational Systems.

Abstract: ***Context:*** Distributed software development is a modern practice in software industry. This is especially true in Open Source Software (OSS) community. In this context, developers are normally distributed around the world. In addition, most of them work for free and without or with low coordinating. Understanding how developers' practices are on those projects may guide communities to successfully manage their projects. ***Goal:*** We mined two repositories of the Apache Httpd project in order to gather information about its developers' behavior. ***Method:*** We developed an approach to cross data gathered from mail list and source code repository through mining techniques. The approach uses software visualization to analyze the mined data. We conducted an experimental evaluation of the approach to assess the behavioral patterns from OSS development community. ***Results:*** Our results show Apache developers' behavior patterns. In addition, we deepen the analysis of the Preferred Representational System of four top developers presented by Colaço et. al in (Colaço et al., 2010). ***Conclusion:*** The use of data mining and software visualization to analyze data from different sources can spot important properties of development processes.

1 INTRODUCTION

A challenge for software engineers and programmers is dealing with complex issues and large software systems while evolving their projects (Sjoberg et al., 2013). Large systems are complex and difficult to understand because of size and complexity that software has achieved. Software engineers frequently face maintenance tasks, which require the understanding of non-familiar software artifacts. Such tasks often imply problems regarding communication, compatibility, and complexity issues (Lanza et al., 2005). Therefore, ensuring the maintainability of software systems is a costly task, and improving this process is a continuous work of research in the software maintainability area. One possible way to deal with this complex scenario is to understand the development community behavior, through software repositories' data analysis, improving software development processes and practices (Heller et al., 2011).

Software repositories have been used to discover useful knowledge about the development, maintenance and evolution of software. However, some of these data sources (e.g. mailing lists) are not built in a structured and organized way. So, we need a considerable effort to gather evidence from those repositories. To this end, researchers have been developing different approaches (Licorish and MacDonell, 2014; Heller et al., 2011; Novais et al., 2013a; Canfora et al., 2011; Eyolfson et al., 2011). They use data mining, software visualization, text mining, and mining software repository. Some of them analyze each repository separately, even that to combine different techniques is a promising approach (Novais et al., 2013b).

Combine approaches from different areas may lead to great results. For example, enrich those techniques with information visualization may reveal valuable hidden software properties. Based on this premise, this paper presents an exploratory study that uses data mining and software visualization techniques to analyze open source software (OSS) developers' behavior. The study is particularly inter-

Farias M., Novais R., Ortins P., Colaço . and Mendonça M..
Analyzing Distributions of Emails and Commits from OSS Contributors through Mining Software Repositories - An Exploratory Study.
DOI: 10.5220/0005368603030310
In *Proceedings of the 17th International Conference on Enterprise Information Systems* (ICEIS-2015), pages 303-310
ISBN: 978-989-758-097-0
Copyright © 2015 SCITEPRESS (Science and Technology Publications, Lda.)
303

ested in analyze discussion mailing lists and source code repositories through the use of software visualization. It integrated and analyzed data originated from Apache Httpd mailing list and source code data.

Our approach extends the empirical studies presented in previous works (Colaço et al., 2012; Farias et al., 2014). The first work proposed strategies to mine Apache server mailing lists, aiming to classify the top Apache Httpd Contributor according to their Preferred Representational Systems (PRS). Next work analyzes Apache developers' behavior, mining mailing lists and source configuration management data (SCM Data).

This representational system gives us a preferred way to use one or more basic systems to communicate and learn. The basic systems usually discussed in the literature are : (1) Visual, (2) Auditory and (3) Kinesthetic. For more detail, the reader should read (Colaço et al., 2012).

In summary, this study deeps one of the research questions presented by (Colaço et al., 2010), introduces two new research questions and, when possible, compares the outcomes of both studies.

This paper is organized as follow. Section 2 presents related works. Section 3 describes our experimental evaluation. Section 4 reports and discusses our findings. Section 5 discusses some threats to validity. Finally, Section 6 concludes the paper highlighting future works.

2 RELATED WORK

This section discusses some related work concerned with identifying patterns in OSS development community through mining software repository or software visualization.

Heller et al. (Heller et al., 2011) proposed a strategy that mined a GitHub repository metadata and used visualization techniques to identify patterns in OSS development community. The study focused on specific patterns, such as the effect of geographic distance on developer relationships, social connectivity and influence among cities, and variation in project specific contribution styles. From the standpoint of behaviour patterns, in (Murgia et al., 2014) the authors have analyzed whether development artifacts like issue reports carry any emotional information about software development. The work has analyzed the Apache Software Foundation issue tracking system. The analysis shows that developers do express emotions (in particular gratitude, joy and sadness). Based on their findings, issue comments have potential as data source for emotion mining.

Some works have already considered email specific analysis to study OSS development process and behavior of people (Rigby and Hassan, 2007; Gill and Oberlander, 2003)Rigby and Hassan (Rigby and Hassan, 2007) have analyzed OSS mailing list content to find developers personalities and general emotional content. In (Gill and Oberlander, 2003), the authors investigated the impact of computer-mediated interaction on person perception. In particular, they studied how important traits for socialization and collaboration may be detected from the text of an email. To this end, they analyzed emails from 30 students at the University of Edinburgh.

Other works are focused on the use of software visualization to understand the OSS developers behaviour. They usually propose new visual metaphors to analyze OSS developers contribution. In (Licorish and MacDonell, 2014), the authors used psycholinguistics, text mining and visualization to examine repository data. Besides that, they demonstrated the utility of combining these approaches to illuminate details of teams' behavioral processes evident in their artifacts. Müller et al. (Müller et al., 2010) presented a visualisation and statistics system called Subversion Statistics Sifter. It explores the structure and evolution of data contained in Subversion repositories. They use statistical graphics and graph plots to analyze both developer activity and source code changes.

Two works are closest to the research presented here. First, Canfora et al. (Canfora et al., 2011) mined explicitly documented cross-system bug fixings from versioning repository and data from two project mailing lists. They tried to identify Cross-System-Bug-Fixings activities between FreeBSD and OpenBSD. They also investigated the social role of developers performing such activities by means of social network analysis. We based our cross-system mailing list in this work. Second, in (Colaço et al., 2010), the authors introduced a psychometrically-based neurolinguistic analysis tool to classify developers through email mining. They conducted an experiment to assess the Preferred Representational Systems of top developers at Apache server mailing lists. In our study, we extended their e-mails and Preferred Representational System analysis.

3 EXPERIMENTAL EVALUATION

This section describes the planning and the operation of the experimental evaluation we conducted to validate our approach. The experimental process follows the Wohlin's guidelines (Wohlin et al., 2012). The next section presents the gathered evidence and the results.

3.1 Goal Definition

The main goal of our study is to reveal interesting behavioral patterns in open source software contributions, such as the effect of geographic distribution between e-mails and commits and correlation over time between emails and commits of OSS project developers. For reaching this goal, we had to gather information through preprocessing and text mining, mining software repository and software visualization to analyze mailing lists, commits from projects and geographic location (geo-location) of contributions in OSS projects.

3.2 Planning

3.2.1 Context Selection

The experiment context was open source project repositories. These repositories have a large amount of e-mails and commits. Commonly, the data is not ready to use. It is necessary to clean the data to avoid misleading understanding. For that, we developed powerful computational procedures following (Colaço et al., 2010; Colaço et al., 2012). On top of that, we did a detailed manual analysis of the committers' profiles in order to gather geographic information. The approach of this study followed three steps: First, we extracted data from: a) Apache's commits repository; b) Apache developer's mailing list; and c) geographic information from geo-location services; Second, we crossed the data collected in the previous step in order to associate the data to the developer that produced it; and finally, we built interactive visualizations that helped users to discover relevant information. Next, we present each of these steps in detail.

We developed four modules for this study in order to provide an able environment to integrate and analyze two repositories from a system. The first is a module for Extraction, Transformation and Load (ETL) of emails. The second module is for mining source code repositories. The next, the integrator module. Finally, the fourth is a visualization module. These visualizations are publicly available at (http://goo.gl/RSs4VR).

3.2.2 Research Questions

This work aims to investigate OSS developers' behavior. To do this, we mined two software repositories in order to analyze the research questions addressing the distribution of email and commits over time in the project. Our efforts were focused on relation between the PRS presented by Colaço et al. in (Colaço et al., 2010) and the contribution made by each developer in the Apache project in depth (Question ii). Moreover, we present one question addressing the distribution of email and commits over time in the project (Questions i). Our research questions are described as follow:

i. How are commits and emails distributed over time among the Apache Project community?

ii. Is there a link between OSS developers' context-specific PRS (Colaço et al., 2010) and contribution made by each developer?

3.2.3 Participant and Artifact Selection

In this study, we used as object of analysis the Apache Httpd Project (http://httpd.apache.org/). Apache Httpd is a HTTP server that aims to offer a robust, efficient and bug free implementation of HTTP services. Accordingly to (NETCRAFT, 2013), it is used for more than 300 million websites that represents almost 40% of the world websites. Apache Httpd is maintained by dozens of developers through Apache Software Foundation (ASF). This foundation is comprised of a community of developers and users that provides support for more than 100 well-known open source projects.

Over its 17 years of development, the project received more than 60,000 commits, totaling more than two millions of lines of code written by more than 100 developers around the world. These developers use a mailing list to communicate with each other. They send emails to discuss several activities, such as development of new features, bug fixes, user problems, and so on. This data can provides useful information about the project evolution and developers' behavior. Because of that, it may be used for several studies in mining software repository.

To answer our research questions, we extracted and analyzed the body of 100,479 email messages and 33,586 commits from the Apache repositories between 1995 and 2005. We selected the four developers who had the greatest number of commits. We refer to these developers as "Dev A", "Dev B", "Dev C" and "Dev D". We also grouped all the other developers, and refer to them as "Cluster", these developers represent the rest of population. We analyzed the same developers and same period used in the related study (Colaço et al., 2010).

3.2.4 Preparation

We prepared a pilot for testing our approach. The pilot study was carried using a small sample of emails and commits, which was chosen at random. Thus, the pilot helped us to calibrate some specific characteristics of our modules and to find improvement point, such as performance of the crossing data and geographic information.

3.3 Experiment Execution

To conduct this research, we developed four modules. The modules were necessary in order to provide an able environment to integrate and analyze two different source repositories. The first is a module for Extraction, Transformation and Load (ETL) of the emails. The second module is for Mining Source Code Repositories. The third on is the integrator module. Finally, the fourth is the visualization module. They are described as following.

ETL OF THE EMAILS: Our approach uses Text Mining (TM). Similar to conventional data mining, text mining consists of phases that are inherent to knowledge discovery process (Fayyad et al., 1996). In this sense, we need to pay special attention to preprocessing, because the used data is unstructured for computer analysis. This means that before setting the text data to be mined, it is necessary to convert each document to a suitable format. A set of emails organized by month was treated as a text document by our approach.

We based on (Colaço et al., 2010; Colaço et al., 2012) processes to mining the email list. Those processes consist of various steps. All of the steps have as final purpose producing data collections with high semantic content. Next, we briefly present the steps. For more details on how to do preprocessing and to clean messages, we suggest to read the used references (Rigby and Hassan, 2007; Witte et al., 2008).

`Step 1`: The original documents are not always represented in a sole textual format. Due to this reason, it is necessary to convert them to a unique format. For that, we needed to eliminate any attributes of presentation formatting, such as footer, signature, source code and attachments.

`Step 2`: To change the letter case to upper case or lower case. This facilitates the matching process. We changed the letter case to lower case;

`Step 3`: To separate each email message. The goal of this step is to recover important properties, such as ”from”, ”to”, ”subject”, ”date”, ”time of the day” and ”weekday”. Identifying the start and the end of each message is a challenge task. In general, there is no a pattern header [1] throughout the mailing list's lifecycle. We had to perform a qualitative analysis in a sample of emails and created a heuristic to detect the headers. After that, we analyzed the outcomes in order to validity our heuristic.

`Step 4`: To group the mined data, summarize and store it into a database.

We applied all these steps to 100,479 emails sent between 1995 and 2005.

MINING SOURCE CODE REPOSITORIES: Apache Httpd uses a SVN repository. We used computational procedure to extract the data from the repository. In the studied period, there are 33,586 commits made by 110 different developers. We built a parser to extract data from commits. From each commit, we extract the ”author name”, ”date”, ”time of the day” and ”weekday” that the commit was made, and the files changed by the commit.

INTEGRATOR MODULE: In order to establish a link between extracted data from email and commits we developed a integrator module, for this at least one property must be shared by both data sources, as shown in Figure 1. For example, either an ID or an email in both data sources should be equal. However, this was not possible, since, in Apache Http project, the data sources have different users' profiles. This was another challenge in our approach. It was necessary to match different kinds of data, e.g. email address with nickname and name with nickname. In order to perform this task, we adapted the approach proposed by (Canfora et al., 2011). Our integrator module is composed of the following steps:

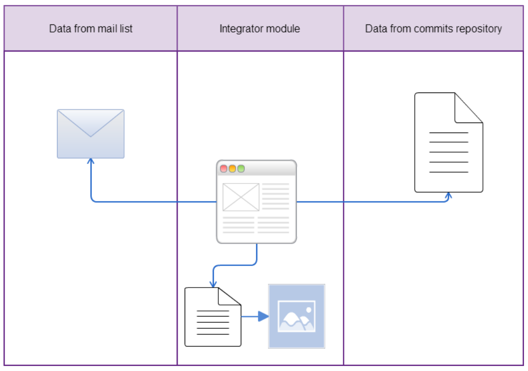

Figure 1: Module Integrator.

(i) Split Email Sender in Developer Name and Email Address. Data from email sender is composed of name and email. These properties had to be split. For example, an entry like ”john lennon <john@email.com>” would result in ”john lennon” and ”john@email.com”;

[1] A pattern that indicates the start of the emails, e.g., From owner-new-httpd@hyperreal.com Wed Nov 1 12:13:11 1995, From dev-return-62023-apmail-httpd-dev-archive=httpd.apache.org@httpd.apache.org Mon Sep 01 06:08:51 2008

(ii) Adjust Special Characters and Upper Case Letters. In this step, names are converted to lower case while special characters were removed. For example, an entry like ”Mário L. Castrol” would result in ”mario l castrol”.

(iii) Match Emails and Commits Identifiers. To perform this task, we used the Levenshtein Distance[2] to verify the similarity between them. We performed several tests and after it we assumed the heuristics: (1) there is a low possibility that an email sender and a repository committer being the same person if the Levenstein distance is equal or less than 0.7; (2) there is a large possibility that an email sender and a repository committer being the same person if the Levenstein distance is equal or above than 0.9. For the ones in the interval]0.7..0.9[, we considered the following steps.

a) Abbreviation of middle name is ignored. For the developers and committers with two or more words in the name, we consider only the first and the last ones. If, after that, there is a match between them, then, we considered the same person, unless if there is another developer with the same first and last name. For example, the entries ”mario l castrol” and ”mario t. castrol” are considered as the same person, if there is not another ”mario castrol” in one of the repositories.

b) First name abbreviated. When there is a name composed of an abbreviated first name plus a last name (e.g. ”j. lennon”), we considered it equals to another name composed of the same last name and a first name that starts with the abbreviated letter. As in the previous step, it only happens when there is not another developer with the same characteristics;

c) Only last name. If there is only the last name and in the collection data there is one, and only one, last name equals to it, we considered them as the same person. For example, the entries ”gonzalez” and ”johnson gonzalez” are considered the same person, if there is not another person with the last name ”gonzalez”.

d) Only initials. If only initial letters composes the name and match another name, we considered them as the same person. For example, the entries ”mwvd” and ”michael-willy van dick” are considered the same person. The exception here happens when exist another developer with the same initial letter, e.g., ”michel-willy victor dagg”.

(iv) Match Email Address. Finally, we compared the nickname in the email address (which comes before the ’@’ symbol) with the committer’s name. If they match we considered as the same person. For example, the entries ”jonnylennon@software.net” can be matched to ”Jonny Lennon”.

[2]http://lucene.apache.org

Even considering all crossing data process, it was not possible to find a match between some emails and commits. In those cases, the data was ignored and corresponded to 29,698 emails. Thus, we considered in our analysis only 70,781 emails.

INTERACTIVE VISUALIZATIONS: Our approach uses six visualizations to help to analyze the extracted data. The first two are heat maps showing the commits and emails distribution around the world (Figure 3). In these heat maps, commits and emails concentration are represented by a gradient composed by 11 colors varying from green (small amount) to yellow (medium amount) and from yellow to red (large amount), each color is used accordingly to a dynamic scale varying from zero until the greatest concentration of points. In Figure 3, one can easily spot the places with more commits. There are also three charts, showing the commits and emails distribution over the years. Figure 2 displays all the commits and email for the whole period. It shows them considering over time, the hour of the day and the days of the week. The Figures 4 and 5 are a bubble chart visualizations. We used it to represent file types that were more modified over time. The larger the bubble size, larger is the number of modifications made in a specific file type.

Besides the graphics, the tool also provides some interaction mechanisms that allow us to filter data. The first interaction is a range slider[3] that is used to filter data by a specific period. The second is a list where we can choose to filter data by a specific developer. There also the maps built-in interactions like zooming and dragging.

After that, we retrieved the geographic information (latitude/longitude and the time offset) for each committer aiming to know the origin of commits and emails. Apache Httpd project does not have this data for all developers. They provide this information only for the core committers (the ones who contributes more to the project).

In these cases, Apache Httpd project provides a page with complementary information about them. Unfortunately, core committers represent only 63 from the total developers (110). For the others, we needed to perform a manual task to retrieve their geographic information. They also have different time offsets. This brings out another issue, since it is necessary to consider the time zone when collecting the weekday and time for each commit. In this case, we needed to get each developer’s time zone and adjust the times for the Apache server time.

After retrieving geographic information, we found

[3]http://jqueryui.com/slider

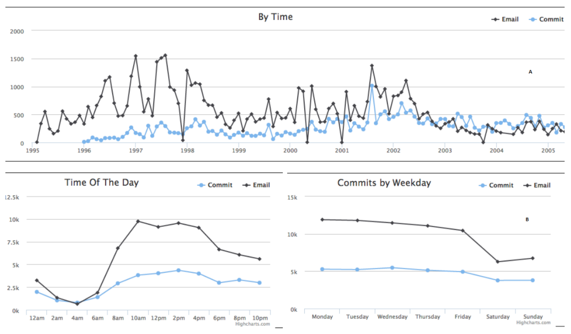

Figure 2: Interactions over time between emails and commits of developer population.

27 more profiles, totaling 90 from the 110 available developers. We decided to remove the commits from those developers, which we could not find geographic information. So, we reduced the amount of analyzed commits from 33,586 to 31,611.

4 RESULTS AND DATA ANALYSIS

The collected data built a rich set of information. Nonetheless, in general, data extracted from software repositories are too difficult to be analyzed in the same state that they were stored (Mazza, 2009). Thus, we decide to use visualizations to reorganize them in such manner that users can easily understand the whole database. We discuss now the results of this study. To answer the research questions, we analyzed the data taking into consideration (i) the relation between emails and commits of the Apache Project developers and (ii) The beginning of Apache Project. (iii) OSS developers' contribution and Preferred Representational Systems.

i. Relation between Emails and Commits of the Apache Project Community: Through the interaction with the period filter to generate heat maps over the time, we perceive that the heat zones (regions where contributions were made) used to appear first in the emails' map and after in the commits' map, it's evidence that in this project developers first interact in the email list and after commit code to the repository.

We could confirm these behaviors in the Apache Httpd Web Site. According to the site, changes to the code are proposed and voted on the mailing list and only after they are approved, they are committed in the repository. On top of that, we could also identify that the regions that have more participation in the emails list are also the regions with more participation in the code repository (see Figure 3). An exception is the Japanese developers' behavior. In this case, there is a considerable amount of commits (bottom of Figure 3) but a low participation in the discussion mailing list (top of Figure 3). It may suggest an introverted behavior due to cultural factors.

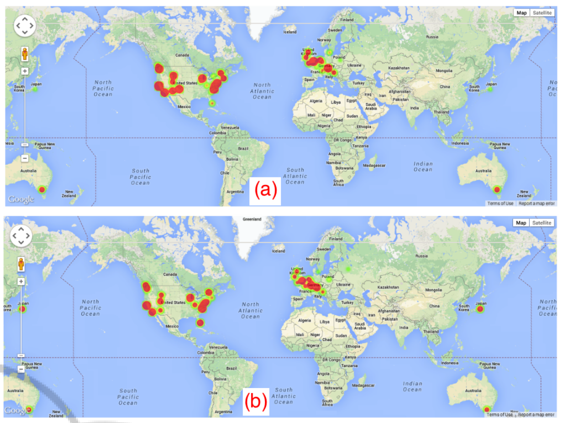

Figure 3: Heat maps showing amount of (a) emails and (b) commits.

We also perceived that there is a correlation between emails and commits timestamps. Developers normally commit code and discuss in the list in the same time as well in the same weekday. Figure 2 shows the interactions over time between emails and commits of the Apache Project developers.

ii. The Apache Project Beginning: Analyzing carefully the data evolution in Figure 2, we can see that the discussion in the email list started a year before the first commits. This is an interesting behavior, since they had time to discuss before to start the implementation. However, it is common to observe OSS projects starting on the other way around. First, a project is created with few developers. They start to commit and create the first release. After that, users start to use it. Using the software, bug fix and request for new features will rise. So, users use a bug tracking system to report the issues. At the end, the developers start the discussions in the email list. We decided to investigate why the Apache httpd evolved in this way. We looked the website and discovered that the Apache Httpd was a continuation of NCSA Httpd, which stopped to be developed when Rob McCool left NCSA in 1994. A group of webmasters then started to develop their own extensions and bug fixes, in 1995, they solved to join all this features and bug fixes in a unique distribution and then the Apache Group was created.

iii. OSS Developers' Contribution and Preferred Representational Systems: The PRS is the one that the person tends to use more than the others to create his/her internal representation (Colaço et al., 2010). In this respect, we accept as true that developers with a kinesthetic profile have more contributions related to code files (emotions and physical experience, holding and doing practical hands-on experiences), while developers with an auditory and visual profile have more contributions related to documentation and architecture (creation of internal images/sound and the use of seen or observed things, including diagrams, demonstrations, flip-charts, sounds reminders, etc.). To confirm or refuse the findings presented by Colaço et al. in (Colaço et al., 2010), we proceed with analysis for each developer as follows.

Dev A and C had a dominance of the kinesthetic profile (Colaço et al., 2010). It's evidence that this developer should work directly with code. We may check it in Figure 4 (Dev A and Dev C), the **Dev A** has a lot of commits changing files with the extensions header (.h) and C files (.c). The **Dev B** was pointed out as visual profile (Colaço et al., 2010), contributing more with documentation and architecture. However, we found out that this developer has had really valuable contribution in commits. It is evidence that he also contributes with source code in project, this fact is confirmed by second score (kinesthetic) presented by (Colaço et al., 2010). The developer is the one from all developers analyzed with most contribution in code files (see Figure 4 in Dev B).

The Dev D has an auditory profile and also a strong visual profile. Furthermore, is also, the one who has the more balanced score between the different profiles. Figure 4 (Dev D) shows the distribution of file performed by this developer. He/she was the developer who more contributed with different files types. In addition, we discovered that this developer contributes with the project documentation, translations and so on. It was confirmed through predominant contributions involving XML files (.xml), HTML files (.html) and image files (.gif and .png). In Colaço et al. in (Colaço et al., 2010), they also created a **Cluster** that is composed of all developers except the top committers (Dev A, B, C and D). In his study, the cluster has a kinesthetic profile, which was also confirmed by our study (Figure 5), there are a lot of commits made by these developers related to code files (.h and .c).

5 THREATS TO VALIDITY

Apache Http is one of the most mature and large OSS project. As showed in Section 3, we experienced several challenge on the process used to analyse its data. We tried to overcome the issues following approaches found in the literature. However, there are still threats to validity.

We did not consider the number of developers analyzed enough to generalize these results for other OSS projects. So, our approach needs further investigation to assure the external validity.

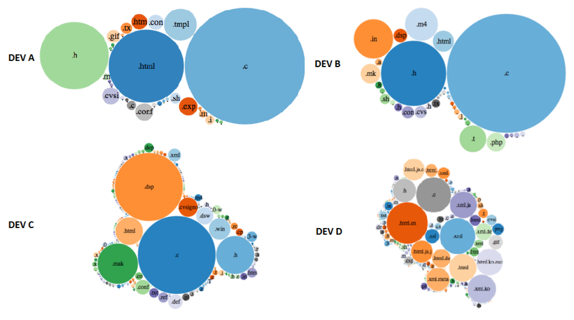

Figure 4: Bubble chart: Apache project's contributions.

We obtained the geographic locations through public profiles. These data may not represent the actual residence of each developer in the moment they contribute to the project. So, we assumed that commits and emails were sent from the developers' place to reduce the threats to internal validity. To this end, we tried to gather data from a questionnaire sent via email to some developers in order to provide us additional information but, unfortunately, we did not receive many replies.

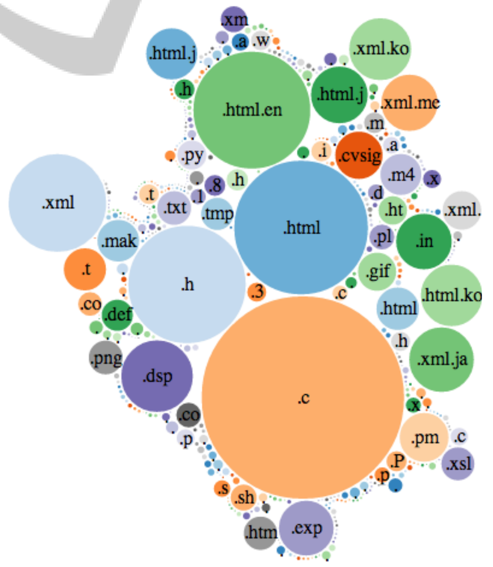

Figure 5: Bubble Chart: Clusters' Contributions.

After retrieval geographic information and data processing, some commits and e-mails were disregarded, totaling 18,8% of developers, 5.88% of commits and 29.55% of e-mails analyzed. Aiming to reduce this threat, we performed a deep qualitative analysis to uncover geographic information. At the end, the data discarded represent small percentage of our sample, which does not compromise our analysis.

6 CONCLUSION AND FURTHER WORK

In this paper, we presented an useful and innovative approach that extracts information from two important software project data sources. We mined and tried to match emails list and source code repository data. This approach can be used to discover hidden behavioral patterns in unstructured data from software repositories. We also believe that OSS leaders can use our approach to increase developers' contributions or to keep contributors in their projects. OSS managers can also use our approach to split tasks according to each developers' profile or to tracking team's contributions over time considering weekdays and day periods.

We have evidences that discussion lists and repositories can be used to measure project activity or to predict each other. We now draw answers to our research questions stated in the section 3. Regarding RQ1, we may confirm that commits and emails follow the same pattern distribution in the Apache evolution. In respect to RQ2, our analysis confirmed the findings discussed by (Colaço et al., 2010) for developers A, C, D, Cluster and refused the developer B. However, we found out that this developer has had really valuable contribution in commits, this setting was also dealt by (Colaço et al., 2010).

Our future work will address three key issues: (1) improve our approach by extracting other relevant data from other OSS. This work is in process; (2) extend this study to mine data from PostgreSQL, emails and commits, aiming to compare to findings performed by (Colaço et al., 2012); and (3) develop new interactive visualizations.

REFERENCES

Canfora, G., Cerulo, L., Cimitile, M., and Di Penta, M. (2011). Social interactions around cross-system bug fixings: The case of freebsd and openbsd. In *MSR*, pages 143–152.

Colaço, M., Mendonça, M., @and Paulo Henrique, M. F., and Corumba, D. (2012). A neurolinguistic method for identifying oss developers' context-specific preferred representational systems. page 112 to 121.

Colaço, M., Mendonca, M., Farias, M., and Henrique, P. (2010). Oss developers context-specific preferred representational systems: A initial neurolinguistic text analysis of the apache mailing list. *MSR*, pages 126–129.

D'Ambros, M., Lanza, M., and Robbes, R. (2010). Commit 2.0. In *WW2SE*, pages 14–19. ACM.

Eyolfson, J., Tan, L., and Lam, P. (2011). Do time of day and developer experience affect commit bugginess? In *Proceedings of the 8th Working Conference on Mining Software Repositories*, MSR, pages 153–162.

Farias, M. A. F., Ortins, P., Novais, R., Colaço, M. J., and Mendonca, M. (2014). Recovering valuable information behaviour from oss contributors: An exploratory study. In *SEKE*, pages 474–478.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). The kdd process for extracting useful knowledge from volumes of data. *Commun. ACM*, 39(11):27–34.

Gill, A. J. and Oberlander, J. (2003). Perception of e-mail personality at zero-acquaintance: Extraversion takes care of itself; neuroticism is a worry.

Heller, B., Marschner, E., Rosenfeld, E., and Heer, J. (2011). Visualizing collaboration and influence in the open-source software community. In *MSR*, pages 223–226.

Lanza, M. and Ducasse, S. (2003). Polymetric views-a lightweight visual approach to reverse engineering. *IEEE TSE*, 29(9):782–795.

Lanza, M., Marinescu, R., and Ducasse, S. (2005). *Object-Oriented Metrics in Practice*.

Licorish, S. A. and MacDonell, S. G. (2014). Combining text mining and visualization techniques to study teams' behavioral processes. In *MUD*, pages 16–20.

Mazza, R. (2009). *Introduction to Information Visualization*.

Müller, C., Reina, G., Burch, M., and Weiskopf, D. (2010). Subversion statistics sifter. In *ICAVC*, pages 447–457. Springer-Verlag.

Murgia, A., Tourani, P., Adams, B., and Ortu, M. (2014). Do developers feel emotions? an exploratory analysis of emotions in software artifacts. In *MSR*, pages 262–271. ACM.

NETCRAFT (2013). Web Server Survey. NetCraft Website. http://news.netcraft.com/archives/2013/06/06/june-2013-web-server-survey-3.html/.

Novais, R., Nunes, C., Garcia, A., and Mendonca, M. (2013a). Sourceminer evolution: A tool for supporting feature evolution comprehension. In *ICSM*, pages 508–511.

Novais, R. L., Torres, A., Mendes, T. S., Mendonça, M., and Zazworka, N. (2013b). Software evolution visualization: A systematic mapping study. *IST*, 55(11):1860 – 1883.

Pattison, D. S., Bird, C. A., and Devanbu, P. T. (2008). Talk and work: A preliminary report. In *MSR*, pages 113–116. ACM.

Rigby, P. C. and Hassan, A. E. (2007). What can oss mailing lists tell us? a preliminary psychometric text analysis of the apache developer mailing list. In *MSR*. IEEE Computer Society.

Sjoberg, D., Yamashita, A., Anda, B., Mockus, A., and Dyba, T. (2013). Quantifying the effect of code smells on maintenance effort. *TSE*, 39(8):1144–1156.

Witte, R., Li, Q., Zhang, Y., and Rilling, J. (2008). Text mining and software engineering: an integrated source code and document analysis approach. *Soft. IET*, 2(1):3–16.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in Software Engineering: An Introduction*. Springer.