Git Mauna Kea

Silviya Sotirova

Abstract

GitHub has become an effective tool in the world of software development and it is considered as a massive online storage space of collaborative works. Users can participate by downloading, studying, and building upon anything.

GitHub is a social coding network, which means that it is used not only from software engineers - everyone can create a profile, follow other users or manipulate repositories. GitHub is giving the opportunity to interact with software development. However, being able to assess the effectiveness of a repository for a specific project, is time-consuming and often requires programming knowledge.

This proposal is giving some background information of the GitHub repositories' usage in software development and scientific research, and is introducing the idea of a software tool that can automate the process of investigating how and where a GitHub repository can contribute.

1 Introduction

Open source repositories are software projects made available through the Internet.[10] GitHub[3] is one of the most widely and largely used code host services, with more than ten million repositories.[9] It is a git version control system and gives the opportunity for teamwork and analysis over the life of a project[25]. "Fork and pull"[1] is the model used in GitHub for manipulation of repositories, where a developer can store a project. With such practice an effective environment is created which supports different type of requests for downloading, called the pull requests, editing or contributing to a repository, called the push requests, and conducting code reviews. It, also, allows viewing the activity within projects, such as issues, commits and documentation. The GitHub's issue tracking system is helping with the creation of a report about the project's features - users can follow other users. As a consequence, there is a constant stream of updates about people, projects of interest, created project's issues and commits.[25].

All of these analysis can be called as repository mining techniques. They are useful for conducting a software engineering research and they are helpful for measuring a software project's productivity, contributions and life length. [13]

Another effective type of analysis is identifying the areas in which a repository might be useful. Text mining algorithms are helpful for investigating and analysing frequency of words and relevancy of files in a project. Also, it is common for a researcher to be interested in the results that a code will produce - the naming of the code items is important for introducing a function and presenting the outcomes from it. Analysis over the implementation of the code and its documentation might be automated by the help of a software program.

Problem Statement

The idea of the research proposal is connected with several problems that are encountered during the investigation of a GitHub repository. One of the them is resulting from the variety of repository contents and activity, which is increasing the time and the level of difficulty for which a researcher or software developer has to find and identify if a specific project fits the research purpose. Also, the combination of different observation made manually when analysing a repository - data visualisations, such as graphs and plots, might be helpful for finding relationships between results, but this requires time and the use of software. Furthermore, the precision and accuracy of calculations and results is unreliable and it can not be tested. Finally, there are other usages of GitHub repositories other than strictly software development - some users are creating repositories to archive data, to host personal projects without contributions to them, or conducting researches that are not using any software equipment.

The variety of issues in are complicating and extending the process of conducting a project that has the purpose to investigate a particular group of repositories and use them. All of the analyses, such as the number of commits, contributors, project's life cycle, and investigations over all other features that GitHub hold and the relationships between them, might be done with the help of a software tool. The research proposal is introducing an idea for creating a reproducible software that will automate the work of mining GitHub repositories and it will assess in which areas of study a specific repository can be used.

Paper's structure

In the Abstract and Introduction 1 section is presented the overall idea of the research proposal. The Introduction 1 section is presenting in more detail GitHub[3] and its repository features.

The Problem Statement 1 is introducing the problems with manual analysis of GitHub repositories, which led to the creation of the research proposal's objectives.

The Background 2 section is separated into several subsections - Mining Software Repositories, IPython Observatory, Software Mining Tools, which are illustrating the motivations behind the research proposal's aims.

National Importance section 3 is explaining the long term effects of the proposed research - is the research contributing to other areas of study and complementing and fitting other UK researches[11].

All of the proposed project's benefits and collaborations with other researchers are described in the Academic Impact section 4 and its hypothesis and objectives in section 5.

Finally, the final page, Methododlogy and Work Plan, is giving a scheme and work plan of a suggested implementation of the project. If the project is approved and funded, the steps and tasks for development might alter depending on technologies and challenges met along the way.

2 Background

Mining Software Repositories

The research paper is introducing an area of study, called Mining Software Repositories (MSR)[20], that is growing together with the massive amount of available software repositories. Its purpose is to investigate and create relationships between the repository data items in order to find unique and useful information about a particular software system and, also, optimize the work that needs to be done during the analysis. Researchers and software developers use their own experience and knowledge for repository investigation and they tend to concentrate on a particular part of the code, e.g. testers usually are focusing on features errors and bugs. However, this might lead to faulty and ineffective results. MSR is an approach for overcoming this problem, it is analysing into great detail repositories and it is building complex models between data items.

In the Hassan's paper [20] there are different types of software repositories and some of them are: historical repositories, such as source control repositories and bug repositories; run-time repositories containing information for the usage of an application; and code repositories, such as Sourceforge.net [8]. Here MSR is described as an approach that aims to convert "static record-keeping" repositories [20] to active ones that can be reused in other projects. More definitions and explanation for what is MSR can be found in Thomas's[34] and Godfrey's [16] paper - *"MSR is a technique used in the field of software engineering (SE) focused on analyzing and understanding the data repositories related to a software development project"*.

Mining Software repositories field of study is one of the inspiration for the creation of this research project. The aims of the two areas are the same:

Automation will eliminating manual work by implementing a scientific software tool which will perform and compute complex calculations and models between software and text features. It allows the repetition of tests and evaluation on various projects and is reducing the time for analysis[20].

Discovery and Validation techniques and models that will show relationships between data items and important information that can be tested for precision and accuracy[20].

MSR is trying to discover and implement useful analysis of repositories by helping in the decision process of a software project[16]. This research paper is concentrating on the features and information that one repository can give. For example, implementing a relationship between number of commits and number of contributors during the project's life, might show a unique finding for the usage of the project. Another example is comparison between the amount of text and the amount of code used in a repository - this might show if a repository is used only for archiving data or for software development purposes.

The listed items above are illustrating requirement point for the project that this research is trying to present. However, every software development investigation is facing certain limitations and issues. Two major problems that MSR is encountering are[20]:

Limited Access to Repositories Not all of the repositories are public, which means that they are available for everyone. Companies that are using GitHub as a version control service for their code are not interested of sharing details about their software systems, which makes research projects less effective since they can not access richer and more complex project than those that have less contributors and features[20].

Complexity of Data Extraction A lot of time, effort and software knowledge is required to extract data from repositories. The new GitHub API (Application programming interface)[4] is allowing access to a huge amount of repositories data. However, it has limitations for copying data and not all repository features are present. Also, non software engineering researchers do not have the expertise to create a software that will be analysing and extracting data[20]. This research proposal is suggesting a convenient and effective approach for gaining access to data in an easy to process format.

This research proposal might be considered as a starting point of developing a software project for mining repositories on GitHub. Being able to analyze a single repository and produce unique results for it, will support the idea for finding relationships between group of repositories.

The Hassan's paper [20] is a great source for understanding in great detail the study of Mining Software Repositories. Above we have seen some of the objectives and problems that MSR has. There are, also, positive effects from implementing MSR software systems, which will present why this research proposal's idea is an effective and innovative. The research [20] is listing several benefits:

Software Systems's aims: There are several aspects to consider when understanding a software system. Some of them are documentation, naming of the code, results, dependencies and test coverage. Evaluating a purpose of a software tool is a challenging process - most of the system do not have detailed documentation or well-structured code. Some repositories contain bugs and are lacking of test cases. An automated system might help with all of this issues. For example, in the Čubranić paper[15], a software tool, called Hipikat, is presenting features, such as old emails or errors [22]. [20].

Changes: Changing a single feature in a component might affect other components of the system. It is a great challenge to keep the consistency of a system after alteration. Instead of using dependencies between components, an automated common framework can use historical co-changes - e.g. change the specific feature in the same way it was changed before[22]. Hassan's and Holt's [21], Zimmermann's [37], Shirabad's [32], Ying's [36] papers are presenting and calculating that historical co-changes proposals have precision and accuracy of 30% in all tested cases, and have 44% correctly identified co-changes.

Detecting Bugs and Errors: Being able to handle failures and errors in a system is one of the most active and demanding field of study in software engineering research. It is an important aspect for the creation of tests and accuracy of outcomes. Developers have been manage bugs and fixes by predicting them. A better approach has been build - an automated system that flags and reports the location of a specific error. However, there are rarely found well structured and documented large software applications and the creation of common and specified behavior is not achievable. Because of that, analysis over huge data sets are implemented, which expose possible locations for bugs.

Team Management: Prioritising and allocating tasks across team members is an active topic in the world of software development. GitHub's interface and issue tracking system are of great help and support of companies that have to manage great amount of teams. All of the established GitHub's issues can be analysed and minded for understanding the team's dynamics. The research of Bird [12] is conducting astudy for mining data from source code repositories and mailing lists. The case studies done in the research are supporting the hypothesis - "1) the rate of immigration is non-monotonic; 2) demonstrated technical skill has an impact on the chances becoming a developer 3) social reputation also has an impact on becoming a developer;"[12].

User Experience: Mining Software Repositories techniques are analysing bugs and execution logs to avoid system's crashing. Michail and Xie [28] are suggesting a software tool that give the user a message for warning if there is a bug in the application. The user can choose to abort the action.

Reusable Code: It is effective if a code can be reused in different types of projects, but it is a challenging process. Developers need to handle each possible use case for a specific topic in order to create a code that can be reused. One more condition for the success of a reusable code is managing different type of technologies. For example, Mandelin's paper is introducing a technique that can access items in API. It is supporting the developers' approach of analysing data[27].

Automating Empirical Studies: Automation is closely connected with the resuability of the code. It is reducing the repetition of features and is generalizing techniques for the functionality of a software system. For instance, Robles' research [30] has observed patterns and behaviours of several large software systems, by the help of data investigation and analysis. A software tool for modeling common features will eliminate a huge amount of work and time. With automation and Mining Software Repository techniques and algorithms is supporting the verification of the generality of a software system's findings.

It is a common approach to use GitHub repositories for storing and archiving information not only locally. However, they consists unique findings and ideas, which could be revealed and applied by the help of MSR techniques. This area of study is discovering various patterns and features of software systems. Developers and researchers have found effective approaches of analysing and investigating data sets and software tools in order to identify bugs and errors and create reusable, automated code.

Mining Software Repositories is an ongoing topic that requires more exploration and support for proving it is beneficial for the world of software development. There are a lot of opportunities for establishing the importance of the MSR field and this research proposal is introducing an idea for automated framework and software that will eliminate manual work that is done on a daily basis. Great source of detail information is the web site [6], which is posting a lot of conference papers for unraveling software systems and addressing challenges in projects, with the help of MSR techniques.

IPython Observatory

We have seen aims, problems and positive aspects of MSR field of study. Knowing the achievements and properties that MSR can offer, is an effective starting point for the initialization of the Git Mauna Kea project. This subsection is explaining how IPython Observatory[33] project motivated the initialisation of the idea behind Git Mauna Kea, using the same structure and steps from the previous subsection.

What is the IPython Observatory[33] project? It is exploring a specific environment of researches available for everyone and investigates GitHub[3] repositories, which uses the software tool IPython Notebook[5]. Open source repositories are an effective source for understnading the structure and content of IPython Notebook scripts. Since GitHub[3] is an web-based Git repository hosting service, it can provide a huge amount of data for software repositories on the Internet.In order to understand how users are managing the site for collaboration on software, researchers have started to investigate and analyze GitHub repositories[26]. IPython Observatory's purpose is to examine the GitHub projects associated with IPython Notebook and assess how exactly they have used it. This is one example of a project that is contributing to the field of Mining Software Repositories, described in subsection 2, since it is researching the usage of a specific tool on GitHub[33].

The reasons for IPython Observatory to investigate that explicit software tool are connected with the methods used for conducting scientific researches. One of them is the availability of the tool - it is an open-source project that can be access by everyone. Furthermore, it represents an electronic version of the notebooks that researchers have and it is giving the opportunity to combine text, code and code output in one file[18]. This allows scientists to structure their ideas and computations. The project is aiming to analyze how is IPython Notebook[5] practiced by researchers and what impact will it have[33]. Helen Shen's paper[31] has described IPython Notebook as a interactive tool for creating journals and helping with keeping track of scientific thoughts and ideas. The "easy interface"[31] of IPython Notebook is effective for teaching, collaboration with others and recording notes[33]. In Perez's paper[29], this software tool has been presented as an approach to combine not only text and code, but even graphs and plots.

As a consequence of that, there are number of challenges and specific MSR features that IPython Observatory have to handle and consider in order to give a compelling assessment. In subsection 2 were listed some of the problems and positive aspects of Mining Software Repositories, which could be taken as a tasks for implementation in IPython Observatory and Git Mauna Kea projects[33].

Some of the challenges that are met along the implementation of IPython are:

Limitations of Data As mentioned in subsection 2, not all repositories are public and accessable. The GitHub API is delivering a great amount of data, but it has limitations for extraction. One of final stages of Git Mauna Kea is implementing a software tool that will have GitHub permissions for analysing all of the available data on GitHub - more detail in the last page, Methodology and Work Plan. As a consequence of this type of issues, IPython Observatory is loosing focus on its main purpose - assessing the IPython Notebook scripts and projects.

Analysis on GitHub features As mentioned above, investigating GitHub repositories produces crucial results for IPython Observatory project, but it is not its primary goal. Even some of the simplest extractions of data might be time-consuming and challenging to extract, and also, they require some computer knowledge. Git Mauna Kea can create a framework that can support projects that are investigating specific software tools, such as IPython Observatory.

One of the aims of IPython Observatory is to be able to produce findings in four areas of interest, which are closely connected with the positive aspects that MSR might bring to the software engineering world[33]:

Correctness The results from the analysis must be tested and logically proven to be accurate and precise. Also, everything should be documented, so that researchers can understand and examine the value of each result.

Reproducibility and Replication Reproducibility as explained in Johansson's paper [24] is "the results obtained from numerical simulations should be reproducible with an independent implementation of the method, or using a different method altogether." It is one of the main principles of the scientific method[14]. When an experiment is reproduced this means it is replicated.

Shared Understanding In order for researchers to understand a specific project reproducibility and a detailed documentation of the project's code should be present. Also, an interactive interface is an effective approach for presenting a project to researchers by allowing them to interact with some of its features.

Prioritization IPython Observatory project should be able to present its findings in a systematised structure - some of them might have more weight than others in a specific repository.

Debugging IPython Observatory project should be able to handle and analyse repositories that have bugs or errors. Some of the IPython Notebook scripts are only documenting a description of a failure and this should be categorised as another type of repository.

IPython is one of the motivations for the project that this proposal is presenting, because one of the main implementation parts of the research is analysing repositories. If an automated tool, investigating a single repository, was created then the time taken for the completion of IPython Observatory was going to be reduced. The difference between Git Mauna Kea project and IPython Observatory is the type of analysis and the outcomes - Git Mauna Kea is mainly concentrated on the automation of the investigation process and IPython Observatory is looking to derive unique relations between the results computed during the analysis.

Software Tools using Mining techniques

This subsection is giving an overview of two software tools used with mining techniques and automating a lot of analysis. The two software applications are practical examples for demonstrating the idea of the Git Mauna Kea project. Each one of them is briefly described in terms of aims and positive aspects.

The first tool that gave inspiration for the concept of the research proposal is the Waikato Environment for Knowledge Analysis(WEKA)[23][19]. It is a software tool for the usage of various machine learning algorithms on different types of data set, which are connected with real-world problems. It is created as a workbench that supports the application of the techniques[23]. WEKA was build as an assistance tool for domain specialists rather than the machine learning experts. Different types of manipulations are available with it, such as database linkage, cross-validation, data visualizations, comparison of rule sets and precision and accuracy calculations.

The Git Mauna Kea project has two stages - creating a framework that models repository data into logically connected components, and software tool that will allow the effortless interaction with the results from the analysis- more details in the last page, Methodology and Work Plan. WEKA is giving the inspiration for the second stage of the project - there are variation for the software tool that can be created, it could be a GitHub plugin, or a Python library with an interactive interface.

Another software tool that has influenced the Git Mauna Project is GHTorrent[17]. It was created to simplify the process of analysing GitHub data and studyies. GHTorrent is a scalable offline representation of the GitHub REST API data and it, also, allows the customisation of data dumps on demand[17] - users are making requests through the GHTorrent website [2] for any types of GitHub repositories.

With the GHTorrent tool, researchers from any kind of study areas are supported in their process of mining repository data and they can focus completely on finding unique results and relations. Positive aspects of the system are that it is lightweight and easy to use, it offers replication and reusability [17].

The GHTorrent tool is one of the approaches for extending the field of Mining Software Repositories. Also, it illustrates an example for software system that will analyse connections between repository data items and GitHub features.

3 National Importance

This section is explaining the long term effects of the proposed research[11]. In subsection 2 is described one of the motivations for the creation of the Git Mauna Kea idea. IPython Observatory[33] project encounters a lot of challenges during its implementation and the most time-consuming one is investigating GitHub repositories and discovering relations between their features. Unique connections between repository's properties is helpful and efficient for IPython Observatory project, but it is not its central purpose, which is researching repositories using IPython Notebook. Git Maun Kea would be of great support for projects like IPython Observatory[33] - it will eliminate a lot of manual work and it will reduce the time taken for implementation. Furthermore, the project will benefit to other areas of study, such as Mining Software Repositories, which is an emerging field in software engineering - demonstrated in subsection 2.

There are several challenges connected with this project, such as amount of data that it will be able to access, mentioned in subsection 2 in the list of problems. However, the tool will be automating the analysis work, which will produce results for every repository.

Nowadays, GitHub analysis is a trending and ongoing topic. The research project is giving detailed explanation of how it can meet national strategic needs and it could maintain unique world leading research activities, which are connected with open source repository services[11].

4 Academic Impact

The main impact of the project is mainly academical. Mining Software Repository field of study and projects connected with analysis over GitHub information will be enhanced from Git Mauna Kea in terms of time, accuracy and precision of results. National and international researchers will benefit from the project since it will allow them to analyse GitHub data without any programming knowledge or code implementation. The project will assist the automation of accessing GitHub data and the computation complex of relations between data items. The Methodology and Working plan page is describing in more detail what are the stages of the project that will support the investigation of GitHub repositories.

Another collaborator of this project is Timothy Storer, a lecturer in University of Glasgow. His role is including the approval of the project's idea and suggesting some insight about the proposed structure of the project. The project's functionality can be extended. As a consequence, the implementation time can vary.

5 Research hypothesis and objectives

In this section we are suggesting the hypothesis of the project and we are describing some of its objectives. The project consists two stages: modelling components of any GitHub repository and creating a software tool that implements that model. Here is the research hypothesis and the null hypothesis for the first stage:

H_1 : A general framework(model) can be created for the analysis over any GitHub repository.

H_0 : A general framework(model) can not be created for the analysis over any GitHub repository.

And below are theresearch hypothesis and the null hypothesis for the second stage:

H_1 : A software tool can be implemented for the automation of the model from stage 1.

H_0 : A software tool can not be implemented for the automation of the model from stage 1.

The project's objectives are connected with effective way of accessing GitHub data, accurate and precise calculations and automation. It is a novel idea since the GitHub community is a great source of information and it has been used from any type of users - non technology researcher and software developers. Furthermore, the project can be implemented in a reasonable amount of time - there are already some software tools that can support it, which we mentioned in section Software Tools with Mining techniques. There are a lot of open-source software tools, such as the Python programming language[7], which are easy to use and manage.

Git Mauna Kea aims to provide a detail report for a specific repository, e.g. users will be able to enter an input, such as the repository url, and they will be able to see results, graphs and plots for relations between data items. It , also, it looks to provide information and analysis over a specific topic that the user has entered as an input. For example, the user will be looking for repositories that relate to and contain IPython Notebooks aspects.

6 Conclusions

Th Git Mauna Kea project is proposing the idea of observing and analysing patterns and behaviour between data items in GitHub repositories. The idea originates from the concept of Mining Software Repositories field. It will support projects that are investigating a specific group of repositories on GitHub. Also, users would be able to identify if a repository is worth for evaluation and usage. In the future, the project might be extended and it will be able to model a group of repositories. The project's idea have a lot of potential and it has a high prospect for success.

Methodology and Work Plan

Network Structure of Social Coding in GitHub Methodology [35]

References

[1] Fork and pull model. https://help.github.com/articles/using-pull-requests/.

[2] Ghtorrent official web site. http://ghtorrent.org/.

[3] Github. https://github.com/).

[4] Github api. https://developer.github.com/v3/).

[5] Ipython notebook official website. http://ipython.org/notebook.html).

[6] Mining software repositories 2016 official website. http://2016.msrconf.org/#/home).

[7] Python official web site. https://www.python.org/.

[8] Sourceforge. https://sourceforge.net/).

[9] Wikipedia - github. urlhttps://en.wikipedia.org/wiki/GitHub.

[10] Wikipedia-open source software(oss). https://en.wikipedia.org/wiki/Open-source_software).

[11] Writting research proposal. https://www.epsrc.ac.uk/funding/howtoapply/preparing/writing/caseforsupport/).

[12] C. Bird, A. Gourley, P. Devanbu, A. Swaminathan, and G. Hsu. Open borders? immigration in open source projects. In *Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on*, pages 6–6. IEEE, 2007.

[13] E. Carlsson. Mining git repositories: An introduction to repository mining. 2013.

[14] M. F. Cohen. *An introduction to logic and scientific method*. Read Books Ltd, 2013.

[15] D. Čubranić, G. C. Murphy, J. Singer, and K. S. Booth. Hipikat: A project memory for software development. *Software Engineering, IEEE Transactions on*, 31(6):446–465, 2005.

[16] M. W. Godfrey, A. E. Hassan, J. Herbsleb, G. C. Murphy, M. Robillard, P. Devanbu, A. Mockus, D. E. Perry, and D. Notkin. Future of mining software archives: A roundtable. *Software, IEEE*, 26(1):67–70, 2009.

[17] G. Gousios, B. Vasilescu, A. Serebrenik, and A. Zaidman. Lean ghtorrent: Github data on demand. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 384–387. ACM, 2014.

[18] P. J. Guo and M. Seltzer. Burrito: Wrapping your lab notebook in computational infrastructure. In *TaPP*, 2012.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[20] A. E. Hassan. The road ahead for mining software repositories. In *Frontiers of Software Maintenance, 2008. FoSM 2008.*, pages 48–57. IEEE, 2008.

[21] A. E. Hassan and R. C. Holt. Predicting change propagation in software systems. In *Software Maintenance, 2004. Proceedings. 20th IEEE International Conference on*, pages 284–293. IEEE, 2004.

[22] A. E. Hassan and R. C. Holt. Using development history sticky notes to understand software architecture. In *Program Comprehension, 2004. Proceedings. 12th IEEE International Workshop on*, pages 183–192. IEEE, 2004.

[23] G. Holmes, A. Donkin, and I. H. Witten. Weka: A machine learning workbench. In *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, pages 357–361. IEEE.

[24] J. Johansson. Introduction to scientific computing in python. *http://nbviewer.ipython.org/github/jrjohansson/scientific-python-lectures/blob/master/Lecture-0-Scientific-Computing-with-Python.ipynb*, 2014.

[25] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian. The promises and perils of mining github (extended version).

[26] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian. The promises and perils of mining github. 2007.

[27] D. Mandelin, L. Xu, R. Bodík, and D. Kimelman. Jungloid mining: helping to navigate the api jungle. *ACM SIGPLAN Notices*, 40(6):48–61, 2005.

[28] A. Michail and T. Xie. Helping users avoid bugs in gui applications. In *Proceedings of the 27th international conference on Software engineering*, pages 107–116. ACM, 2005.

[29] F. Perez, B. E. Granger, and C. Obispo. An open source framework for interactive, collaborative and reproducible scientific computing and education, 2013.

[30] G. Robles, J. J. Amor, J. M. Gonzalez-Barahona, and I. Herraiz. Evolution and growth in large libre software projects. In *Principles of Software Evolution, Eighth International Workshop on*, pages 165–174. IEEE, 2005.

[31] H. Shen et al. Interactive notebooks: Sharing the code. *Nature*, 515(7525):151–152, 2014.

[32] J. S. Shirabad, T. C. Lethbridge, and S. Matwin. Supporting software maintenance by mining software update records. In *Software Maintenance, 2001. Proceedings. IEEE International Conference on*, pages 22–31. IEEE, 2001.

[33] S. Silviya. Ipython observatory(research proposal). page 25, 2015.

[34] S. W. Thomas. Mining software repositories with topic models.

[35] F. Thung, T. F. Bissyandé, D. Lo, and L. Jiang. Network structure of social coding in github. In *Software Maintenance and Reengineering (CSMR), 2013 17th European Conference on*, pages 323–326. IEEE, 2013.

[36] A. T. Ying, G. C. Murphy, R. Ng, and M. C. Chu-Carroll. Predicting source code changes by mining change history. *Software Engineering, IEEE Transactions on*, 30(9):574–586, 2004.

[37] T. Zimmermann, A. Zeller, P. Weissgerber, and S. Diehl. Mining version histories to guide software changes. *Software Engineering, IEEE Transactions on*, 31(6):429–445, 2005.