SOFTWARE ENGINEERING AS A SOCIAL PROCESS: A STUDY OF RESEARCH TRIANGLE PARK

by

Lindsey Lanier

April, 2016

Director of Thesis: Mark Hills, PhD

Major Department: Computer Science

Powered by increased capabilities of online collaboration tools, open source software development has become a critical component of the development landscape for both individual developers and for organizations. However, very little is known about how developers and organizations in regions with a heavy information technology presence participate in the open source community. This research uses public data from GitHub, a popular online open source collaboration tool, to gain a better understanding of open source software development activity in one such region, the Research Triangle Park in North Carolina. Research Triangle Park is one of the largest research parks in the world, and the largest in the United States. Within the 7,000 acre park, there are currently more than 200 companies employing around 50,000 people in a number of fields of expertise, with information technology playing a large role. To explore how developers and organizations in this region are involved in open source development, we use data mined from GitHub to aggregate information and trends about the developers and organizations themselves, their open source projects, and their use of the social media features within GitHub. The data collection tools and techniques created to enable this research are designed to easily enable similar studies to be performed for other geographical regions.

SOFTWARE ENGINEERING AS A SOCIAL PROCESS: A STUDY OF RESEARCH TRIANGLE PARK

A Thesis

Presented to The Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Software Engineering

by

Lindsey Lanier

April, 2016

SOFTWARE ENGINEERING AS A SOCIAL PROCESS: A STUDY OF RESEARCH TRIANGLE PARK

by

Lindsey Lanier

APPROVED BY:

DIRECTOR OF THESIS: _____

Mark Hills, PhD

COMMITTEE MEMBER: _____

Nasseh Tabrizi, PhD

COMMITTEE MEMBER: _____

Junhua Ding, PhD

CHAIR OF THE DEPARTMENT OF COMPUTER SCIENCE: _____

Venkat Gudivada, PhD

DEAN OF THE GRADUATE SCHOOL: _____

Paul J. Gemperline, PhD

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Open source software development is continuing to grow within information technology communities. The data capable of being extracted from collaborative development platforms gives researchers an opportunity to answer many questions about open source project activity and trends. However, at this point in time, we know little about how people are participating in open source development from a regional perspective. This research is aimed at studying open source activities through public data within GitHub [2] to understand what the open source software engineering demographic looks like in Research Triangle Park (RTP), NC, one of the most prominent research parks in the world.

GitHub is a widely used platform for open source software development and collaboration. Its extensive features give software developers the capability to join forces with others from all over the world on a variety of projects, from small projects to major projects that are being used at the enterprise level. GitHub provides a public REST API which grants researchers the opportunity to mine and study this data from an empirical perspective. Using the GHTorrent Project [3], [4] as our foundation for information, we focus on answering a few key questions throughout this research:

1. What do the metrics look like around individual users and organizations and their involvement in open source projects in the RTP area?

2. What do the overall numbers look like with regards to projects being developed by users and organizations in the RTP region? How active are these projects, and are they still relevant since inception?

3. How do users and organizations in the RTP region use the social media features available on GitHub?

While the answers to these questions have focused on the RTP region, the scripts developed for this empirical study can also, with some generalization, be used to study other geographic areas, which will enable future research including answering similar questions for other regions and comparing multiple regions. Chapter 2 will go through key reasons on why RTP was chosen as a strategic location for this research. Chapter 3 will provide more information around GitHub, with an overview of terms and associated workflows, as well as an overview of the GHTorrent project. Chapter 4 will discuss related research on this topic, particularly studies into open source data. Chapter 5 and 6 will go through the methodologies used for data collection and then provide analysis of the RTP data collected. Lastly, we will wrap up in Chapter 7 with closing remarks about this research and ideas for future research.

Research Contribution: Through empirical techniques applied to data mined from project repositories on GitHub, this thesis quantitatively shows how both individual developers and organizations participate in open source software development in the Research Triangle Park region of North Carolina. The process used to extract and compute over the repository data has been scripted to ensure the reported results can be easily replicated, while the scripts themselves have been designed to enable future studies of other technology regions as well as comparative studies of multiple regions. The data collection tools and techniques created to enable this research are

publicly available on GitHub (User ID: LindseyKLanier, [5]) under an open-source license.

Chapter 2

Overview of Research Triangle Park

In this chapter, we first define Research Triangle Park, the region studied in this thesis. We then explain why this region was chosen for this study, focusing on the rapid growth of the region and the significant presence of high tech companies and major universities.

2.1 Where is Research Triangle Park?

Research Triangle Park (RTP) is the largest research and science park in North America, and one of the largest in the world. The park, which stretches 7,000 acres across Wake and Durham counties was founded in 1959 by the Research Triangle Foundation. RTP is not a city, but it has its own special county district [1]. The Research Triangle Region Organization, a partnership dedicated to overseeing collaboration between businesses, government and various other institutions within the region identifies RTP as the following counties (see Figure 2.1): Chatham, Durham, Edgecombe, Franklin, Granville, Harnett, Johnston, Lee, Moore, Nash, Orange, Person, Vance, Wake, Warren, and Wilson [1]. Additionally, the RTP Organization has stated that the RTP metro area has a population of 1.3 million people, and that there are 3 million people within a 60-mile radius of the park [6]. Our project utilized public NC League of Municipalities (NCLM) data to identify all of the cities in each of the RTP counties

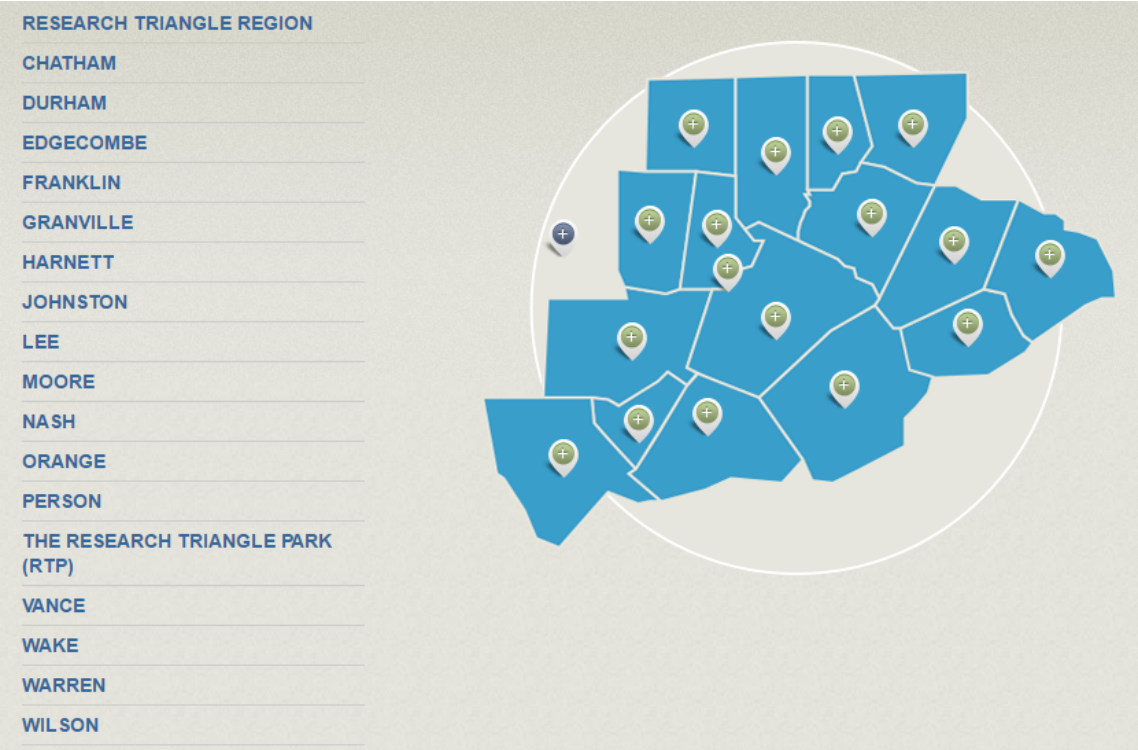

Figure 2.1: RTP region as defined by Research Triangle Region Organization [1]

to assist our research [7].

2.2 Why Research Triangle Park?

Many reasons can be attributed to our choosing of RTP and its surrounding communities for initial research into local open source activities. Over the last decade, numerous media outlets and journalists have continuously given this region nicknames such as Start-up Capital of the South [8] and The Next Silicon Valley [9]. Between 2000 and 2012, Raleigh's population grew 47.8 percent, which topped Forbes's list of fastest growing metropolitan areas in the country. This number was more than three times the overall growth of all other metropolitan areas [10]. The Google for Entrepreneurs Network has invested in 3 different locations in Raleigh/Durham (RDU)

for what they call "American Underground". American Underground is referred to as a "Start-Up Incubator" where software groups can rent space for their teams to meet and work [11]. The Research Triangle Park Organization continues to drive their mission to "change the course of history" by bringing people together and fostering innovation and creativity in the RTP community [12]. They've recently open up a new center, The Frontier, with free space for teams to collaborate and participate in a variety of networking events [13].

Outside of the start-up communities, many big technology players have a presence in RTP–Red Hat, Citrix, Cisco, NetApp, IBM, Microsoft, Google, SAS, and others, with more news about others following every year. There are several local prestigious universities including the University of North Carolina Chapel Hill, North Carolina State University, and Duke University, all of which are consistently producing top talent within the science, math and technology industries. According to the Research Triangle Park Organization [12], local companies and organizations have won Nobel Prizes and the Pulitzer. To date, they've recorded that there have been 245 company start-ups, 3,256 patents, and 1,970 trademarks.

At the beginning of 2016, the North Carolina State of Technology Industry Report was released. We found some of the statistics produced by this group (NC Technology Association and Economic Leadership of Raleigh) relevant to reasons we chose the RTP region, even though they were specific to the state as a whole [14].

A few of the highlights from this report included:

- North Carolina is the #2 state in Information Technology (IT) employment growth from 2009-2014.
- North Carolina is the #4 state for University Technology Licenses and Options Executed with 242 licenses and options in 2014.
- Future North Carolina tech sector growth is expected to outpace national and American South averages.

- Over the past year technology employment grew by 3.6 percent, and technology establishments grew by 5.2 percent.

These statistics can serve as a proxy for choosing to do research on other areas with a large and growing presence in the information technology industry.

Chapter 3

GitHub and The GHTorrent Project

In this chapter, we will give an overview of GitHub as a platform as well as the GHTorrent project and it's associated data source. GitHub and GHTorrent provide data which is the foundation to the research performed in this thesis.

3.1 GitHub

GitHub is a web-based platform used for version control and collaboration between developers. Users can choose to utilize standard Git commands in a terminal to manage their code on GitHub or use the GitHub graphical user interface (GUI) and associated features on GitHub.com. As of February 2016, there were 12 million people collaborating across 31 million repositories on GitHub [2]. Users wanting to create a GitHub account for their projects have two options, public (free) and private (paid). Public repositories are heavily used by open-source teams to allow for code storage and collaboration from users all over the world. GitHub gives users the option to create "organizations" to more easily manage teams. Organizations allow teams to group one or more projects under one umbrella. Multiple owners can be mapped to organizations for settings and permissions management.

This research pulls data on GitHub's source code management features with a focus on mining RTP specific statistics as well as looking into the involvement that

RTP users have in the social networking aspect of the platform. Source code packages hosted on GitHub are referred to as repositories. We utilize the GHTorrent Project [4] for data collection of RTP specific (public) repositories. Public repositories can be browsed and contributed to by not only the owner but anyone else interested in downloading a local copy. Changes by GitHub users can be submitted to the repository owner for review in the form of a "pull request". Changes are saved to repositories through what are known as "commits". There are various workflows that could be implemented for GitHub projects. One popular option is to utilize "forking" and "pull requests". A "fork" is a clone of a repository, saved as a new repository on one's own GitHub account. Users might implement their own enhancement to the cloned application, or could also work on an associated "issue". One or more commits can be pushed to this clone. Once the developer is finished with their changes, they can then submit a "pull request" to the original repository owner for review. If the repository owner is happy with the changes, they can merge the "pull request" into a branch on the original repository. This is the main workflow that we focus on throughout this research.

During the data collection phase of this research, we noted some important details about commits which were unexpected. The author for any given commit is the user who wrote the code to be committed, the committer is the user who pushed this code into the main software repository. Git allows users to commit on behalf of another person, hence the commit can be pushed by someone who did not author it. Depending on the workflow used to commit the code change, the author/committer information can vary. For example, if a user commits a code change through the command line on their local machine they can include author and committer information as free text form (not ideal if they spell their e-mail address wrong). GitHub has a large knowledge base which documents its various features, including a help article

which explains this commit information further [15].

There are also many useful social networking features available on GitHub. GitHub allows users to follow friends and track what they are working on. Users can also follow particular repositories and see associated updates, also known as "stargazing". GitHub also provides an interface which shows which repositories are trending at any given time, and which can be sorted by programming language.

3.2 The GHTorrent Project

GHTorrent was created for the software research community, a place to easily come to query specific sets of open source repository data without having to go through the GitHub REST API directly. GHTorrent has a relational database (MySQL) which stores structured data on users, repositories, commits, etc. GHTorrent stores the bulk of its data in a MongoDB instance. MongoDB is a type of NoSQL database which utlizes JSON and allows for rapid changes and agile development. MongoDB stores data in the form of JSON documents within what they call "collections". Collections can compare slightly to a relational database's "table" but they are quite different themselves in that they are dynamic, flexible and do not need to have a defined schema. The collections included in GHTorrent's MongoDB instance are outlined in Table 3.1. MongoDB provides advanced querying operations such as Map-Reduce and aggregation which gives data miners flexibility when conducting their research [16]. GHTorrent chose this technology to be able to more easily manage the large collections of dynamic data as well as accommodate changes to the GitHub schema quickly, among other reasons such as giving the research community flexible options for querying [3].

The GHTorrent project included 4TB of data in their MongoDB instance (as

Collection name	Github API URL
commit_comments	#user/#repo/commits/#sha/comments
commits	repos/#user/#repo/commits
events	events
followers	users/#user/followers
forks	repos/#user/#repo/forks
issues	/repos/#owner/#repo/issues
issue_comments	repos/#owner/#repo/issues/comments/#comment_id
issue_events	repos/#owner/#repo/issues/events/#event_id
org_members	orgs/#org/members
pull_request_comments	repos/#owner/#repo/pulls/#pullreq_id/comments
pull_requests	repos/#user/#repo/pulls
repo_collaborators	repos/#user/#repo/collaborators
repo_labels	repos/#owner/#repo/issues/#issue_id/labels
repos	repos/#user/#repo
users	users/#user
watchers	repos/#user/#repo/stargazers

Table 3.1: List of collections available in the GHTorrent MongoDB instance.

of January 2015) [4] but according to more recent updates on their Twitter feed [17] (@ghtorrent), it has been constantly growing and aims to pull not only more recent data but to store the entire historical GitHub data source. Their Twitter feed (@ghtorrent) keeps the community up to date with regular posts, most recently noting that Microsoft has agreed to sponsor the project on Azure for the next 2 years. GHTorrent's users collection was recently geocoded using OpenStreetMap (OSM) data and interestingly, the code for geocoding was provided by a local RTP contributor (GitHub Login: DerekTBrown). This particular feature was extremely helpful for our research in that we could query users based off their geocoded city or state instead of parsing the free text location field ourselves.

As documented in [3] which was written during the initial stages of GHTorrent's development (2012), there are several challenges that researchers would face if they decided to use the GitHub REST API directly, challenges that the GHTorrent project

has overcome through their provided data source. One of the main challenges is that the GitHub data source is massive. Unless you know exactly what you are looking for and where to look for it, it may take days to search through the collections given the 5000 request/hour limit per API key. The GHTorrent project has done the heavy lifting for us by reverse engineering and documenting the GitHub schema and REST API. Having that information readily available saves researchers much time from having to do this themselves.

In order to overcome the 5000 request/hour limit per API key, GHTorrent designed their workflow to be distributed, with the research community donating SSH keys to allow for extra workers and concurrency across a cluster of machines. SSH keys can be donated by members of the GitHub community [1]. GHTorrent uses the REST API [18] (Events Stream) to collect static repository information as well as other events such as associating commits with users and repositories. They have a messaging layer (RabbitMQ) which sits between the events stream and aids with pushing out further requests to the API from there. A single event can ultimately lead to many downstream requests to gather more information from other parts of the REST API. GitHub's event stream is constantly flowing with new information as users are working inside projects. For example, each commit fires off a number of creation requests to other parts of the overall schema. A documented limitation for this is that the event stream handles all additive actions but does not report deletions. [4]

As with all large scale projects, there are challenges and limitations documented for GHTorrent. As highlighted in the previous section, it is possible for the author and the committer to be two different people and there can sometimes be issues with the mapped user credentials when commits are pushed via the command line (free text is allowed). If GHTorrent is unable to map the user to a commit, it creates a

[1] We donated SSH keys to enable this.

fake user entry until the next request is created when it will attempt to resolve the fake entry. As of 2013, there were several thousand fake users in the data source. Another potential problem with the quality of the data is staying on track of any regular updates to the GitHub schema itself. As GitHub makes changes to its overall schema, the GHTorrent project has to closely monitor in case any changes are required to its processes. Another potential problem with the quality of this data source is that there are known periods of missing data from the event stream due to an error in a couple of the GHTorrent scripts—these periods include a few days in March 2012 and mid-October through mid-November 2012. [4]

Chapter 4

Related Work

There have been a large number of studies done on open source software, particularly the mining of GitHub data to gain a better understanding of how the widely popular tool is being used across the globe. Due to the incredibly large nature of GitHub as a platform (as highlighted in Chapter 3), it is important for researchers to understand first the questions they want to answer and second the various options they have available to collect GitHub data prior to starting the mining process. Cosentino, Luis and Cabot collected 231 works on mining GitHub, eventually narrowing these down to a set of 93 which they use to try and better understand how GitHub repositories are being mined, how the data ends up being used and how associated limitations are presented and addressed [19]. They discuss overall limitations with the use of various GitHub datasets and the fact that they are not always current or consistent. They also found that two thirds of the 93 selected works did not provide enough information for future comparative studies to replicate the work. They discuss limitations of their own study and conclude by highly suggesting that researchers ensure datasets and instructions to replicate studies on mining GitHub are shared with the research community.

Prior to beginning this research on the RTP area, ideas on what we wanted to focus on were formed through the perusal of various published papers. Related work

on topics such as the impact of geography on contributions in GitHub, determination of local skill sets based off GitHub data, programming language trends, the impact social networking features have on users as well as a study that looked to determine the promises and perils of mining GitHub through user surveys will be cited and discussed further in this chapter. The most important piece of research that we initially focused on was the GHTorrent project created by G. Gousios. Much of the information sourced from this project's public research is already outlined in Chapter 3. We were able to use the GHTorrent data source as a means to quickly get started in searching for RTP specific data on users, their repositories and overall open source activity, which we later use to answer questions on what the social engineering demographic looks like in RTP.

A study was done by a group of researchers (including Gousios from GHTorrent) on the promises and perils of mining GitHub [20]. At that point, there were no known studies of the quality and properties of data available through GitHub. This research was focused around trying to gain a better understanding of how users take advantage of the various capabilities that they are exposed to within GitHub, particularly committing, submitting pull requests and issues. Out of the 1,000 surveys sent to active users who had listed their e-mail address on their GitHub account, only 240 responses came back. Although the sample was small, the researchers were able to collect some valuable information which they share as part of their research. This study concluded, in summary, by showing that most repositories are inactive and only created for personal reasons. We can conclude similar results as part of our research on the RTP area given the number of repositories that were created and never touched again. These researchers also found out through a survey, that many developers host their code on GitHub for the sole reason of free hosting, having no desire to ever open it up for collaboration. In addition, 38% of users involved in the survey said

they use GitHub primarily for their own projects, with no intention of collaboration with other developers. Due to these findings, the authors provide recommendations based on their research to suggest the best way to study GitHub data as a whole by explaining in detail some of the different workflow use cases that they are aware of for personal and project work.

In today's communities (from our experience), development of software is quite often split between different regional teams. As audio/video collaboration technologies continue to advance, the distance of these teams matter less. However, there are still many instances where regional clusters contain concentrated software engineering industries—RTP being an example as we are demonstrating through this research. Takhteyev and Hilts did research on the geography of open source software through GitHub [21]. These researchers came up with a unique method to study regional teams and their associated involvement in open source projects. They started with a single account (one of GitHub's founders) and began collecting more accounts and information based off of their connections, ending up with a sample of 70,414 accounts. As we mentioned previously, the location field in a GitHub user's public profile is free text and hence difficult to query directly. This team created a method to code the location using Geonames.org, Yahoo's GeoAPI, as well as some manual intervention. They were able to conclude (based on sample accounts) that 39% of all GitHub accounts are located in the United States, the remaining 61% being spread across the globe (second largest country being the United Kingdom at 7%). The top 5 clusters identified in the United States were San Francisco, New York, Boston, Seattle, and Chicago.

David Rusk and Yvonne Coady from the University of Victoria did some research on analyzing the popularity of programming languages in their local community (Victoria, BC, Canada) in 2014 [22]. One of the goals to this research was to give employ-

ers as well as potential employees a better idea of what the local skill sets look like through a talent pool repository as well as an overview of the common technologies used in any given location. This team created a tool which pulled data out of the GitHub REST API and dumped it into a MongoDB instance. It is unclear why they did not use the GHTorrent data source as it was cited as related work and all of this information could have been pulled directly from the GHTorrent MySQL instance itself. They were able to create some visualizations similar to what was done during the data analysis phase of this project and provided these to local developers and businesses for feedback. They discussed that most of the developers found the information quite helpful, some being a bit concerned with developer privacy given the researchers were publishing names and other bits of personal information within the project itself. Others pushed back and highlighted the fact that these profiles were already public to begin with and that listing them within the project was a required feature. Employer feedback was also positive, they were happy with the fact that they could find developers with specific skills, code samples they've written, as well as contact information all in one place. Their aim was to be able to expand this to other locations as part of future work. As our research looks at similar metrics for the RTP region, publishing this data could prove useful in the future.

In 2013, Begel, Bosch and Storey conducted interviews with 4 leaders from the software development industry to try and gain a better understanding around the social networking aspects of open source software development [23]. Brian Doll, an engineer in the marketing space for GitHub was interviewed as part of this initiative. We found some of his answers particularly interesting and could relate them back to some of the questions we were looking to answer as part of this research. Storey asked Doll, "How does social networking play a role in the services you provide?" Doll then started describing a recent e-mail he had received from a past colleague

who had mentioned he was following his activity on GitHub, such as the repositories he had starred. In this way, users are able to keep up with various projects that their connections are watching or actively participating in. We explain how often RTP users are utilizing this feature as part of our local research. Doll was also asked how other relationships were formed on GitHub, outside of the stargazing technique. He started describing some of the use cases for creating a GitHub organization user type which is something we look at the usage of for our local research. Doll mentions that it is quite common for projects to be managed in this way on GitHub because "it's the cleanest way for them to give permissions to several developers with different levels of access to the code." Another important point that Doll makes in this interview is the importance that GitHub sees in ensuring users can put a face to the name of the project's main developer. This way, the user who is doing the bulk of the work is getting the actual credit for it. For this reason, GitHub keeps the user login ID in the URL structure of each repository, and also makes heavy use of avatars in activity feeds. This point is interestingly related to the current controversy over privacy between GHTorrent project and GitHub users [24]. Doll also discusses his opinion around the benefit of open source software development for people looking to get a job within a programming company, regardless of experience, and discusses articles that he has read which claim GitHub as being the new resume for programmers.

Chapter 5

Data Collection

This chapter will walk through the data collection method that was developed for this thesis. The GHTorrent project was chosen as our main source to collect GitHub data on local projects in order to gain a better understanding of what the local open source community looks like in RTP. The ultimate goal of this research was to not only study our region, but to also develop a method that could be reused for future research on other locations through the consolidation of Python and SQL scripts that were created. As explained in Chapter 3, the GHTorrent project offered an offline mirror of public data pulled from the GitHub REST API and allowed us to get started on our investigations immediately. GHTorrent provides two methods to collect data, a MySQL instance which includes structured meta-data and a MongoDB instance which includes in-depth GitHub information in the form of JSON documents. An example of a query and associated results from both MySQL and MongoDB can be found in Figure 5.1 and Figure 5.2 respectively. These figures give an idea of how much more information is stored in the GHTorrent MongoDB versus MySQL instance.

In the beginning of the data collection phase, specific meta-data (which will be discussed further in future sections in this chapter) was pulled on RTP users and repositories by running SQL scripts against the GHTorrent MySQL database. We then used the meta-data collected and queried the GHTorrent MongoDB collections

MySQL Query:

```
select id, login, name, location
        from users
        where login = 'lindseylanier'
```

Results:

id	login	name	location
5641774	lindseylanier	Lindsey Lanier	USR

Figure 5.1: Example of a MySQL query to pull the information on one user and the associated results after executing.

MongoDB Query:

```
> db.users.find({"login":"lindseylanier"}).pretty()
```

Results:


```
{
        "_id" : ObjectId("56c563256480fd331e002493"),
        "login" : "lindseylanier",
        "id" : 8949639,
        "avatar_url" : "https://avatars.githubusercontent.com/u/8949639?v=3",
        "gravatar_id" : "",
        "url" : "https://api.github.com/users/lindseylanier",
        "html_url" : "https://github.com/lindseylanier",
        "followers_url" : "https://api.github.com/users/lindseylanier/followers",
        "following_url" : "https://api.github.com/users/lindseylanier/following{/other_user}",
        "gists_url" : "https://api.github.com/users/lindseylanier/gists{/gist_id}",
        "starred_url" : "https://api.github.com/users/lindseylanier/starred{/owner}{/repo}",
        "subscriptions_url" : "https://api.github.com/users/lindseylanier/subscriptions",
        "organizations_url" : "https://api.github.com/users/lindseylanier/orgs",
        "repos_url" : "https://api.github.com/users/lindseylanier/repos",
        "events_url" : "https://api.github.com/users/lindseylanier/events{/privacy}",
        "received_events_url" : "https://api.github.com/users/lindseylanier/received_events",
        "type" : "User",
        "site_admin" : false,
        "name" : null,
        "company" : null,
        "blog" : null,
        "location" : null,
        "email" : null,
        "hireable" : null,
        "bio" : null,
        "public_repos" : 6,
        "public_gists" : 0,
        "followers" : 0,
        "following" : 0,
        "created_at" : "2014-09-28T18:15:42Z",
        "updated_at" : "2016-01-06T02:33:15Z"
}
```


Figure 5.2: Example of a MongoDB query to pull the information on one user and the associated results after executing.

(see Table 3.1 for the full list of available collections) for further information. The meta-data was needed to meet GHTorrent's various index requirements on their MongoDB instance. Due to the massive data source and heavy loads, indexes are required as GHTorrent's MongoDB instance has a 100 second time limit (non-indexed searches easily surpass this). PyMongo [25], a Python/MongoDB library was used to connect

to the GHTorrent MongoDB instance programatically (although connections using a command prompt were also available and used on occasion for searches). The results from the queries executed against the GHTorrent MongoDB instance were stored in a locally created MongoDB instance for analysis.

Subsequent sections of this chapter will show a common theme around the method we used to both collect and analyze GitHub data in our local MongoDB instance, namely utilizing MongoDB's aggregaton framework [26]. The aggregation framework provides the capability to write queries and perform powerful transformations on collections that surpass the basic MongoDB "find" operation. We will show the ways aggregation was utilized in this project to calculate data points such as the number of original repositories per user, the number of forked repositories per user, the number of pull requests submitted per user, the number of years each user has been a member of GitHub, programming language popularity, etc. Once the data was aggregated (and new collections were formed), we were able to drill into the various bits of information deeper to gain a better understanding of local activity and answer questions about the users and their associated repositories. The basic MongoDB "find" operation was also used in order to search the newly created collections to pull back datasets used to create the various figures presented in this research.

5.1 User Information

5.1.1 Collecting RTP Users

Before starting to look into the Research Triangle Park data, we first had to identify which cities this region consisted of. As discussed in Chapter 2, the Research Triangle Organization [1] identified the RTP region by county and later we found that the NCLM [7] had a list of associated cities per county. There were a total of 90 cities (see

Figure 5.3) that we were then able to query the GHTorrent MySQL database instance with. This initial step was required as we couldn't directly start using the GHTorrent MongoDB instance due to the users collection having an index requirement on login name (we could not query the GHTorrent MongoDB instance directly on the location field alone without timing out). Additionally, the geocoding of user locations is only available in their MySQL instance. From the GHTorrent MySQL instance, we were able to export the list of users to a comma separated value (CSV) file. We then wrote a Python script which would iterate through the CSV file and query the GHTorrent MongoDB instance for users by their login name, saving the results to a new collection within our local MongoDB instance. This newly formed users collection gave us the beginning of what would later become a 25GB database of RTP GitHub data.

5.1.2 Calculating Total Number of Years on GitHub

One of the questions we sought to answer was the average number of years that local users had been GitHub members. Per Wikipedia, GitHub was founded 8 years ago in February of 2008 [18]. We were interested in understanding if there were any local, active users that had been around since the start. In order to do this, we had to run a few different PyMongo scripts. The first thing that had to occur was the conversion of the "created at" date in the users collection to an "ISODate". The date field that we pulled from GHTorrent's MongoDB instance was not a proper date format. We were not able to run any aggregation functions on the date until it was converted (see Figure 5.4 which shows an example of how easily we are able to do this with Javascript). Next, we used aggregation to get the number of years the user had been a member and projected the output to a new collection (see Figure 5.5). Once this collection was created, we were able to query for users that had been a member for any specified number of years.

1. Aberdeen
2. Angier
3. Apex
4. Bailey
5. Benson
6. Black Creek
7. Broadway
8. Bunn
9. Butner
10. Cameron
11. Carrboro
12. Carthage
13. Cary
14. Castalia
15. Centerville
16. Chapel Hill
17. Clayton
18. Coats
19. Conetoe
20. Creedmoor
21. Dortches
22. Dunn
23. Durham
24. Elm City
25. Erwin
26. Four Oaks
27. Foxfire Village
28. Franklinton
29. Fuquay-Varina
30. Garner
31. Goldston
32. Henderson
33. Hillsborough
34. Holly Springs
35. Kenly
36. Kittrell
37. Knightdale
38. Leggett
39. Lillington
40. Louisburg
41. Lucama
42. Macclesfield
43. Macon
44. Mebane
45. Micro
46. Middleburg
47. Middlesex
48. Momeyer
49. Morrisville
50. Nashville
51. Norlina
52. Oxford
53. Pine Level
54. Pinebluff
55. Pinehurst
56. Pinetops
57. Pittsboro
58. Princeton
59. Princeville
60. Raleigh
61. Red Oak
62. Robbins
63. Rocky Mount
64. Rolesville
65. Roxboro
66. Sanford
67. Saratoga
68. Selma
69. Sharpsburg
70. Siler City
71. Sims
72. Smithfield
73. Southern Pines
74. Speed
75. Spring Hope
76. Stantonsburg
77. Stem
78. Stovall
79. Tarboro
80. Taylortown
81. Vass
82. Wake Forest
83. Warrenton
84. Wendell
85. Whispering Pines
86. Whitakers
87. Wilson
88. Wilson's Mill
89. Youngsville
90. Zebulon

Figure 5.3: List of local RTP cities which were collected and used in this research.

```
db.githubRTPUsers.find().
        forEach
    (
        function(element)
        {
                element.created_at = ISODate(element.created_at);
                db.githubRTPUsers.save(element);
        }
    )
```

Figure 5.4: MongoDB function used to convert the created date to a proper ISODate.

```
db.githubRTPUsers.aggregate(
[
        {$project: { _id: 0,
                item: "$login",
                diff_days:{
                $divide:
            [{$subtract: [new ISODate(),"$created_at"]}
                        ,1000 * 60 * 60 * 24 / 365]}}},
        {"$out":"calculated_memberForYears"}
])
```

Figure 5.5: Aggregation in MongoDB used to get the number of years a user has been a member on GitHub. Results are projected to a new collection within our local MongoDB instance.

5.2 Repository Information

5.2.1 Collecting RTP Repositories

Once the local users baseline had been established, we began looking into what these users were working on within the open source world of GitHub. The next baseline that we took was user owned GitHub repositories. Again, due to required indexes (repository name and owner login name) in the GHTorrent MongoDB repositories collection, we needed to pull some meta-data out of the GHTorrent MySQL instance before we could query. We used the previous users CSV file to query the GHTorrent

MySQL instance (projects table) for everything where the owner's login name was equal to our user's login name. We exported this information to a new CSV, imported it into our local MongoDB instance, and used it to pull more information on user's repositories from GHTorrent's MongoDB into our local MongoDB instance. This new collection gave us pertinent information about local GitHub repositories which we later used to answer our research questions in the data analysis phase.

As mentioned in the introduction of this chapter, MongoDB provides an aggregation framework which was used to sort through much of this data. Figure 5.6 shows an example of calculating the total number of repositories by user with the MongoDB aggregation framework (using PyMongo). Here we set up a "pipeline" using the group, sort, project, and out operators. In this particular example, we sought to group the count of repositories by the owner's login name and sort the results in descending order. Each aggregation created a new document that was projected to a new collection in our local MongoDB instance. The results of this aggregation query helped us address questions we had around how many repositories each RTP user had created. As part of this research, we were also interested in understanding which cities in the region had the most repositories. In order to do this, we created a new collection using the same method just described (MongoDB aggregation framework). These results were captured and are discussed in the next chapter on data findings.

5.2.2 Original vs. Forked Projects

In order to determine original versus forked projects from our collection of local RTP projects, we needed to take a look at the "fork" field for each repository. The "fork" field takes a boolean value of True or False. We wrote a Python function that would count the number of original projects per user as well as the number of forked projects per user. Two separate collections were created in our local MongoDB instance and

```
def repo_count_by_user():
    pip = [
        {"$group": {"_id": "$owner.login", "count": {"$sum": 1}}},
        {"$sort": SON([("count", -1), ("_id", -1)])},
        {"$project" : {"_id":1, "count":1}},
        {"$out" : "calculated_repoCountByUser"}
    ]
    userReposCollection.aggregate(pip)
```

Figure 5.6: Sample Python code which pulls the total repository count per user using PyMongo and the MongoDB Aggregation Framework.

the data was later analyzed to help us understand if local users were forking existing projects more often than creating new projects.

5.3 Overall RTP Activity

Once we had established a baseline of GitHub users and associated repositories, we could start calculating RTP users' overall activity. As discussed, one piece of this research that particularly interested us was to find out how many users were actually active open source community members, in other words, how many users are actively contributing towards live projects. For this, we wanted to ignore users that joined for one project or homework assignment, for example, and never came back again.

In order to gain an understanding of the overall GitHub activity in RTP, we investigated from a few different angles. We first aimed to understand trends in overall activity over the lifetime of a local user account and then wanted to understand who is still currently active. We are defining "active" in this research as making a commit to a repository over the last 6 months. In order to determine overall activity, we created the following questions to guide us:

1. Total number of commits by owner

2. Total number of active repositories based off commits

3. Total number of forks

4. Total number of pull requests

5. Total number of stargazers

Question 1-2 are covered in section 5.3.1. Question 3 was covered already in section 5.2.2. Question 4 is covered in section 5.3.2 and question 5 is in section 5.4. The results from each data collection method are captured in Chapter 6.

5.3.1 Collecting User Commits and Active Repositories

As part of the study of overall activity within our local scope, we sought to understand what the total number of commits per project looked like. In order to collect this information, we used the meta-data located in the GHTorrent MySQL instance in combination with our already existing collection of local users in our local MongoDB instance. We queried the GHTorrent MySQL instance for the name of the repository, the owner name, the count of commits as well as the last commit date. Figure 5.7 shows the script in detail. Due to the massive volume of commit data, we let this script run overnight as it took several hours to complete. During the initial run, we ran into errors that were not caught in the code. MongoDB requires UTF-8 encoding and hence any entries which were not UTF-8 that we tried to insert into our local MongoDB instance failed (**Error Message: bson.errors.InvalidStringData: strings in documents must be valid UTF-8**). We altered the script to translate these field names to binary to get past this, logging any future failure details to a new collection. The subsequent run of this script did not give us any failures, however. The total number of commits per project (as well as their associated timestamps)

```
def commit_numbers():

    query = "select p.name, u.login, u.name,
    max(c.created_at), count(c.id)
    from projects p, users u, commits c"\
    " where u.id = p.owner_id"\
    " and c.project_id = p.id"\
    " and u.login = %s"\
    " and p.name = %s"\
    " group by p.name, u.login, u.name"

    reposList = userReposCollection.find(
        {},{"owner.login":1, "name":1})

    i = 0
    try:
        for x in reposList:
            if(i > 16969):
                mySqlCursor = mySqlDB.cursor()
                mySqlCursor.execute(query,(x['owner']['login'],x['name']))
                data = mySqlCursor.fetchall()
                for row in data:
                    myLocalDB.testCollection.insert([
                        {"project_name":row[0],
                        "owner_login": row[1],
                        "owner_name": bson.Binary(str(row[2])),
                        "last_commit_date": row[3],
                        "commit_count":row[4]}])
                    print "done with ", x['owner']['login'], x['name'], i
            i+=1
            # print x['owner']['login'], x['name']
    except Exception, e:
        myLocalDB.rejects.insert({"failed":i})
```

Figure 5.7: This Python script was used to collect the number of commits per local project.

were useful is helping us answer defined questions on RTP user activity and current participation in the open source community.

5.3.2 Collecting Pull Requests

In order to determine overall user activity, we needed to understand how many pull requests had been submitted by each RTP user. For this, we pulled data directly

out of the GHTorrent MySQL database and inserted it into our local MongoDB instance. The SQL query we used performed a union of two select statements. The first statement joins locally owned repositories on the head repository ID in the pull requests table and the second statement joins on the base repository id in the pull requests table. This is done to ensure we collect any repositories that have been forked. The script used for collecting the pull requests is shown in Figure 5.8.

5.4 Social Networking - "Stargazing"

In addition to overall open source activity, we were interested in understanding the popularity of local users and their work. One way to do this was to look into how many users had repositories which were being "starred" or "followed". We wrote a Python script to query the GHTorrent MongoDB instance (watchers collection), using our collected list of RTP users and their owned repositories. We inserted the results from this script into a new collection in our local MongoDB instance. Figure 5.9 shows an example of what one of these entries looks like.

After the list of "stargazers" had been collected, we again used the MongoDB aggregation framework to find out which local repositories were the most popular. Figure 5.10 shows what this query looked like. One thing to note for this collection is that if a user is a contributor on a project then the stars for that project end up in our list - some examples of this are shown in Chapter 6 (Data Findings). Lastly, we utilized aggregation to find out how many followers each local user had (see Figure 5.11).

```
def getPullReqs():

    query = "(select u.login, p.name, count(*)
        as 'prcount', 'head' as 'repotype'"\
            " from projects p, users u, pull_requests pr"\
            " where p.owner_id = u.id"\
            " and pr.head_repo_id = p.id"\
            " and p.deleted is false"\
            " and p.forked_from is null"\
            " and u.login = %s"\
            " group by p.id"\
            " order by count(*) desc)"\
            " UNION"\
            " (select u.login, p.name, count(*)
                as 'prcount', 'base' as 'repotype'"\
            " from projects p, users u, pull_requests pr"\
            " where p.owner_id = u.id"\
            " and pr.base_repo_id = p.id"\
            " and p.deleted is false"\
            " and p.forked_from is null"\
            " and u.login = %s"\
            " group by p.id"\
            " order by count(*) desc)"

    myUsers = githubRTPUsers.find({},{"login":1}) # first find all usernames

    i = 0
    try:
        for x in myUsers:
            mySqlCursor = mySqlDB.cursor()
            mySqlCursor.execute(query,(x['login'],x['login']))
            data = mySqlCursor.fetchall()
            for row in data:
                myLocalDB.calculated_pullReqByUser.insert([
                    {"login":row[0],
                    "name": row[1],
                    "prcount": row[2],
                    "repotype": row[3]}])
                print "done with ", x['login'], i
            i+=1
    except Exception, e:
        print e
        myLocalDB.rejects.insert({"failed":i})
```

Figure 5.8: Python script used to query for pull requests from the GHTorrent MySQL database and then insert them into our local MongoDB instance.

```
> db.githubRTPStargazers.findOne()
{
        "_id" : ObjectId("539eb518bd35432a2004564c"),
        "following_url" : "https://api.github.com/users/HagamosVideojuegos/following{/other_user}",
        "events_url" : "https://api.github.com/users/HagamosVideojuegos/events{/privacy}",
        "organizations_url" : "https://api.github.com/users/HagamosVideojuegos/orgs",
        "url" : "https://api.github.com/users/HagamosVideojuegos",
        "gists_url" : "https://api.github.com/users/HagamosVideojuegos/gists{/gist_id}",
        "html_url" : "https://github.com/HagamosVideojuegos",
        "subscriptions_url" : "https://api.github.com/users/HagamosVideojuegos/subscriptions",
        "repo" : "BombaFiesta-Unity3D-Futile",
        "owner" : "edbartley",
        "avatar_url" : "https://avatars.githubusercontent.com/u/6969130?",
        "repos_url" : "https://api.github.com/users/HagamosVideojuegos/repos",
        "received_events_url" : "https://api.github.com/users/HagamosVideojuegos/received_events",
        "gravatar_id" : "028308b2ed248bbd76fa686f0855006e",
        "starred_url" : "https://api.github.com/users/HagamosVideojuegos/starred{/owner}{/repo}",
        "site_admin" : false,
        "login" : "HagamosVideojuegos",
        "type" : "User",
        "id" : 6969130,
        "followers_url" : "https://api.github.com/users/HagamosVideojuegos/followers"
}
```

Figure 5.9: A view into the stargazers collection.

```
db.githubRTPStargazers.aggregate(
[
        {$group: {
        _id:"$repo",
        total: {$sum: 1}}},{$sort:{total:-1}
        }
])
```

Figure 5.10: Aggregation used to find out how many followers each local repository contained.

```
def stargazing():
    pip = [
        {"$group": {"_id": "$owner", "count": {"$sum": 1}}},
        {"$sort": SON([("count", -1), ("_id", -1)])},
        {"$project" : {"_id":1, "count":1, "owner":1,"repo":1}},
        {"$out" : "calculated_totalStargazers"}
    ]
    stargazersCollection.aggregate(pip)
```

Figure 5.11: Python script used to calculate the number of followers that each RTP user has.

Chapter 6

Data Findings

Once the data collection method was developed and implemented, we started mining through the information collected in an attempt to answer the key questions described in chapter 1 for the RTP region. We broke down those high level questions into more specific items described in section 6.1. To find out about GitHub users in the RTP region, we provide answers to questions one through five. In order to see what the overall numbers look like in terms of projects developed out of the RTP area as well as their associated activity and current relevance, we answer questions six through twelve. Finally, in order to gain a better understanding of how RTP users are taking advantage of GitHub social networking features, as well as their overall popularity, we answer questions thirteen through fifteen.

We found that the easiest way to illustrate a lot of these findings was to create visual representations of the data collected. For this, we used Plot.ly, an open source framework for developing data visualizations in various languages [27]. Figure 6.1 shows an example of the creation of a bar chart using this framework. The remainder of this chapter is broken down into specific sections dedicated to answering the key research questions. We first go into an overview of RTP users, then drill into RTP specific repositories, overall open source contribution activity and lastly look into the usage of social networking features available on GitHub.

6.1 Key Research Questions

1. How many users does the RTP area have overall?
2. How many users does the RTP area have by type (User vs. Organization)?
3. What is the average number of repositories per user in RTP (overall)?
4. How long have local users been GitHub members? How many users have been around since GitHub started?
5. What city has the most users?
6. What is the repository count by User?
7. What is the repository count by Organization?
8. Who has the most repositories overall in the RTP area?
9. Are users creating original repositories or forking existing repositories or collaborating more often?
10. What do the commit numbers look like for local RTP repositories?
11. How many of the RTP repositories are active (for this research, active is defined as repositories which contain changes that have been committed at some point over the last 6 months)?
12. What is the average life of a local project?
13. Are many local users being "followed"?
14. Are local repositories being starred often?
15. What are the most popular programming languages?

```
def userCountByCity_BarChart():
    results = userCountByCity.find({"count":{"$gte" : 20}})

    x = []
    y = []

    for i in results:
        x.append(i['_id'])
        y.append(i['count'])

    plotly.offline.plot({
    "data": [
        Bar(x=x, y=y,
            marker=dict(
            color='rgb(158,202,225)',
            line=dict(
                color='rgb(8,48,107)',
                width=1.5
            ),
        ),
        opacity=0.6)
    ],
    "layout": Layout(
        title="Number Of Users By City (User Count Greater Than 20)",
        annotations=[
        dict(
            x=xi,
            y=yi,
            text=str(yi),
            xanchor='center',
            yanchor='bottom',
            showarrow=False,
        ) for xi, yi in zip(x, y)]
    )
    })
```

Figure 6.1: Sample Python code which pulls the total repository count per user using PyMongo and the Aggregation Framework.

6.2 Users Overview

In this section, we will focus on answering questions one through five from section 6.1. We will find out how many users the RTP area has overall, by both User and Organization type. We will also find out how many repositories each user has on average, how long they have been members and which city in the region has the most

users. The collection of RTP users was the most important dataset in this research as everything else was built upon them. As discussed in chapter 2, RTP is a very prominent location for the information technology industry as a whole. We suspected we would find open source activity in the area but did not have any baselines from other locations to compare to. We managed to find 3,234 total RTP users (again, only being able to find users that had their location listed on GitHub). This means only .1% of the RTP population (as also discussed in chapter 2, there are 3 million people living in a 60 mile radius of the park) are members of GitHub. Of these, 47 percent of them were located in Raleigh. Figure 6.2 shows a bar chart created with Plot.ly with the number of RTP users by city. This chart only shows 8 cities as we filtered out all cities that had less than 20 users. Out of the 90 cities identified through NCLM (see 5.3), only 33 contained users contributing on GitHub. Raleigh, Durham and Chapel Hill had the highest number of users in the area which is understandable given these locations are home to the major research universities as well as many of the large technology companies referenced in chapter 2. We also looked into the total number of RTP users by user type, finding that the majority of accounts were "User" - 2,982 with the remaining 252 being of type "Organization".

Another metric that we were interested in gathering was how long the local RTP users had been members on GitHub. As mentioned in the previous chapter, GitHub has been around for 8 years now. We were curious to find out how many local users had created their accounts when the service went live. There ended up being around 130 users that have had their accounts for 7-8 years. Out of these accounts, 61% (80 users) have been active (or submitted a commit) over the last 6 months. Figure 6.3 shows the script used to determine active users who have been members on GitHub for over 7 years. It was interesting to note that over half of the oldest users were still active on GitHub.

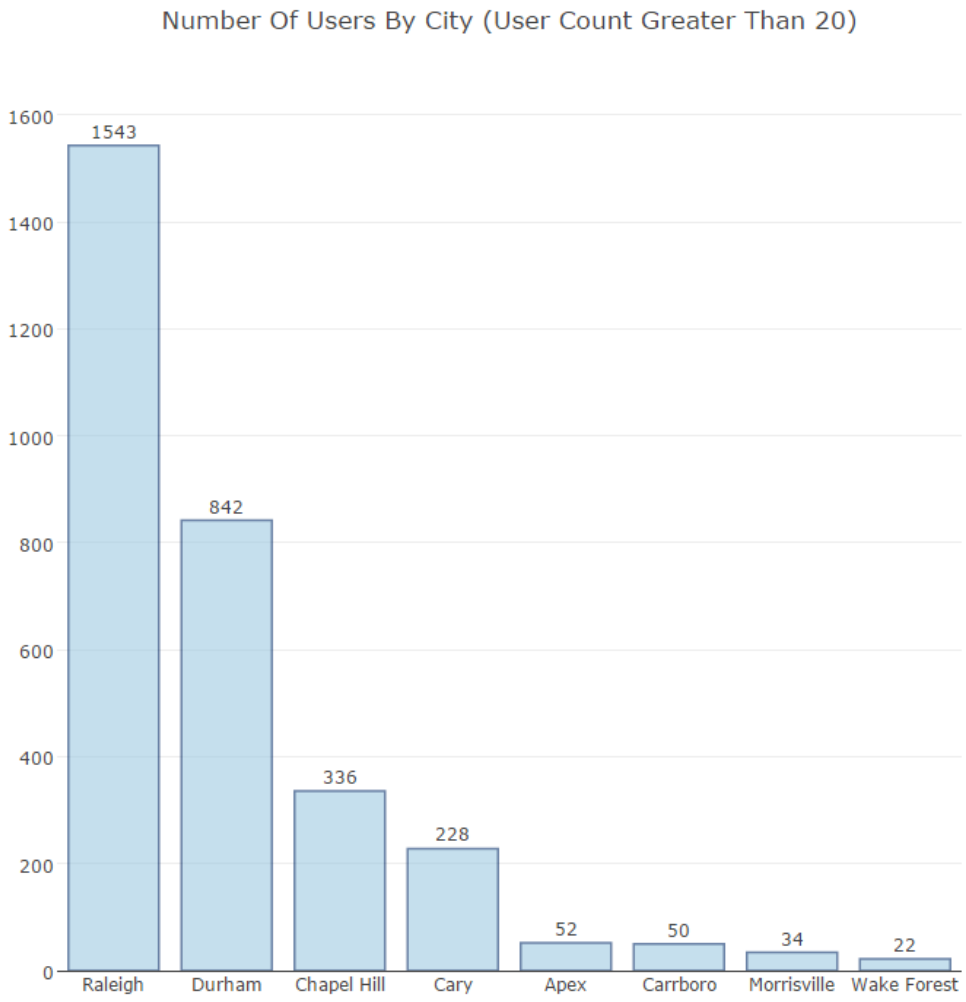

Figure 6.2: This figure shows all RTP cities that have more than 20 users.

6.3 Repositories Overview

In this section, we will present an overview of the RTP repositories found during the data collection phase through answering questions six through nine from section 6.1. We will learn how many repositories have been created out of RTP (overall and by user type), as well as which RTP user has the most repositories. In total, we were able to find 34,825 total RTP user repositories. Of these, 31,843 were owned by a

```
def active_oldUsers():
  d = datetime.datetime(2015, 8, 10, 12)
  results = memberForYears.find({"diff_days":{"$gte" : 7}})
                          .sort("diff_days",-1)

  count = 0
  for i in results:
    totalCommits = activeUsersCollection.find({"owner_login":i['item']})
    for x in totalCommits:
        if x['last_commit_date'] > d:
            print 'Active: ', x['owner_login'], ' ',
                               x['project_name'], ' ',
                               x['last_commit_date']
            count += 1
```

Figure 6.3: This figure shows the Python script used to find RTP users that have been members on GitHub for more than 7 years and are still active today.

"User" type, while 2,982 were owned by an "Organization" type. There was a large amount of information available about each one of these repositories. Each repository contained an entry in a collection within our local MongoDB instance. These entries specified a number of items, including the owner details, the number of forks, whether or not the repository was a fork itself, the programming language used, the number of watchers, the create date, last updated date, as well as several API URLs that could be used to perform a number of actions on GitHub (view comment, view issues, view events, download the repository, etc.).

We calculated that RTP users have created an average of 12 repositories. The user with the most repositories had created 351, while there were a large number of users that had only created one repository. Figure 6.4 is a chart created to show the users with the most repositories (breaking it down to users with more than 120 owned repositories). This does not necessarily mean they are active repositories and in fact could have been created and only used as a means of storage and never opened up for collaboration nor as a source code repository. The user with the most number of repositories was created as an "Organization", the account is still active today but

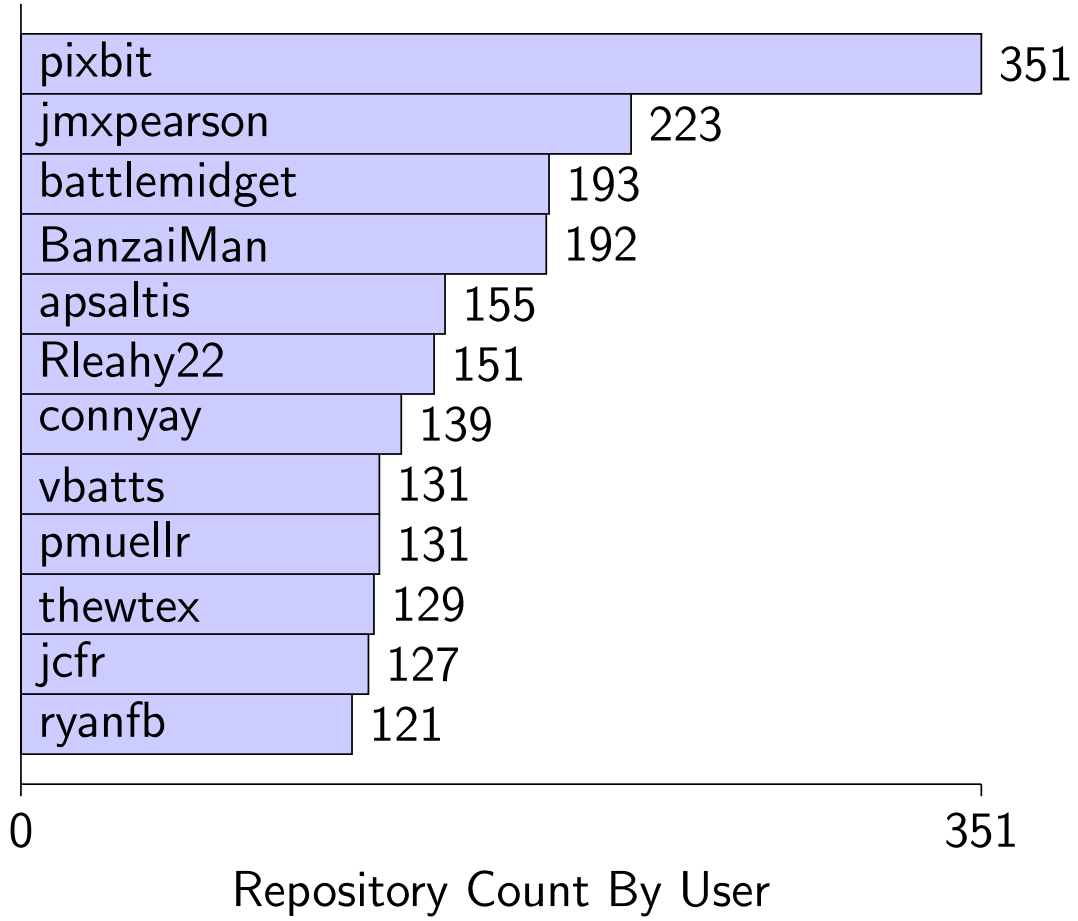

Figure 6.4: This bar chart shows users with the most overall repositories in the RTP region.

not all 351 repositories are currently being contributed to. We will assess the activity of all repositories in section 6.4.

The next piece of information gathered about overall repositories in the RTP region was the total number of original versus forked projects. We were interested in understanding if users created original projects more often than forking from already existing projects. Table 6.1 and 6.2 show the findings from these calculations. After sampling this data, we found out that around 40% of the projects in RTP are forked and 59% are original. This does not exclude inactive repositories. What this tells us is that while more than half of the repositories are original projects created by

Forked Repositories	Total Users
15,244	2099

Table 6.1: Total Original Repositories in RTP

Original Repositories	Total Users
19,745	2518

Table 6.2: Total Forked Repositories in RTP

the RTP owner, there are still a large number of projects that have been forked from already existing projects with a possible intent to collaborate.

6.4 Activity Overview

We did some investigating into how many local users were active while seeking to answer questions ten and eleven. We find out what the commit numbers look like for RTP repositories, as well as the overall activity of RTP users. As discussed previously, we defined a user as being "active" if they have made a commit sometime over the last 6 months. We found that 2,559 unique users had made a commit over the last 6 months on GitHub which we seemed quite high. We can't necessarily say that 79% of RTP users are active on GitHub, however, due to the fact that was discussed in chapter 3, when committing a change in GitHub, the author field is free text in many cases. We can see this quite clearly in the collected dataset on commits. There are unfortunately many users who lazily entered data such as their first name only (i.e. "Ben") instead of their full name, e-mail address or login name as an identifier. As part of future research, it would be useful to programatically attempt to match authors to their rightful GitHub account, similar to what the GHTorrent project does (as discussed already in section 3.2). This way we would be able to calculate a more accurate number on the total number of active users in the area.

We were able to say for certain that there are 6,756 unique RTP owned repositories that are active on GitHub, since this field cannot be free text. This means that of all the repositories owned by RTP users, 19% of them are active. Figure 6.5 shows all

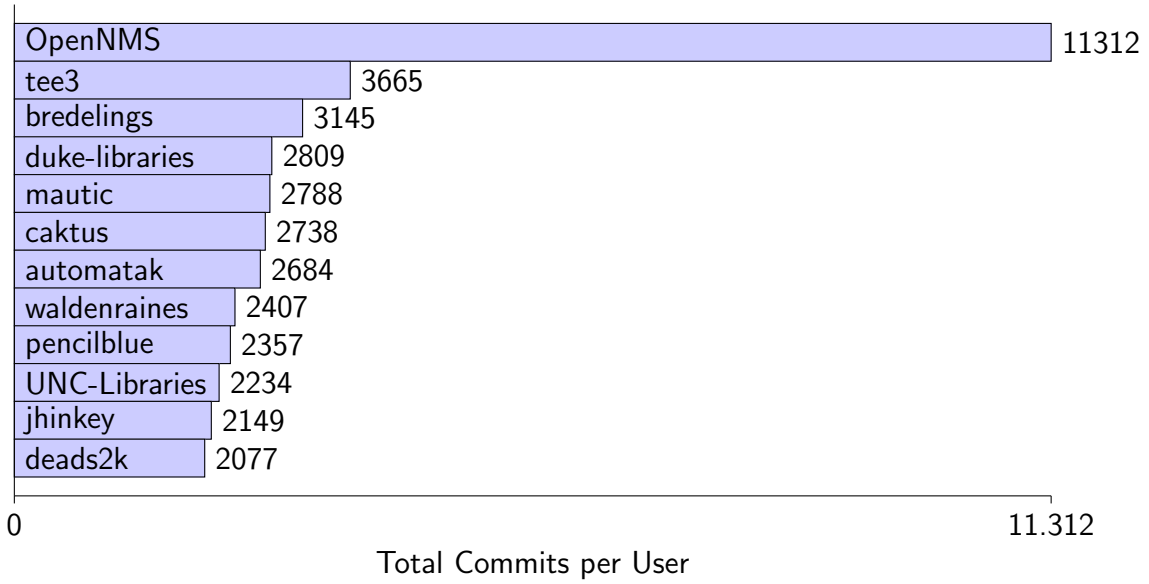

Figure 6.5: This figure shows all users with more than 1000 commits over the last 6 months.

users with more than 2,000 commits over the last 6 months. We looked a bit further into the top 10 active users (by login ID) to find out more information about them. All 10 of these users have made more than 2,000 commits to projects over the last 6 months (see Table 6.3). As we can see, 70% of these users are set up as Organizations on GitHub, which makes sense in that there would be more commits than "User" projects where there may be only one contributor. It was hard to identify a specific trend between these numbers. For example, these project owners either heavily utilize pull requests, or they don't. This observation could be explained by the fact that users and organizations utilize different workflows as already discussed in section 3.1.

6.5 Popularity Overview

Finally, after reviewing metrics on RTP users, their repositories and overall activity on GitHub, we decided to take a look into the data to find out how "popular" our local users and projects were. In this section, we will discuss questions twelve through

User ID	User Type	Member Since	City	# Projects	# Pull Reqs	# Stargazers
OpenNMS	Organization	Dec-12	Pittsboro	66	864	254
tee3	User	Mar-10	Raleigh	18	2	5
bredelings	User	Oct-09	Durham	8	2	15
duke-libraries	Organization	Oct-12	Durham	52	2310	40
mautic	Organization	Aug-13	Raleigh	14	718	479
caktus	Organization	Apr-10	Durham	89	1602	856
automatak	Organization	Jan-13	Raleigh	10	42	81
waldenraines	User	Apr-13	Raleigh	17	0	0
pencilblue	Organization	Feb-14	Raleigh	15	751	1257
UNC-Libraries	Organization	Jan-11	Chapel Hill	28	766	157

Table 6.3: More information on the top 10 active RTP users (have pushed commits in the last 6 months). This chart shows their total number of projects, pull requests and stargazers.

fifteen. We will seek to find out if many RTP users are being "followed" or "starred". We will also find out what the most popular programming language is. As mentioned in chapter 3, GitHub gives us an easy way to do this with their built in social media features such as "stargazing". We were able to dig into the collected data to find out that some of our users are actually somewhat popular. 1,253 users (39%) contain at least one follower. We have 2 local users that have more than 10,000 followers. Figure 6.6 gives us a view of the number of stargazers that each user has (where the total number of followers is greater than 1000). The RTP user with the most stargazers is an author of a book on PHP, who lives and works in Chapel Hill.

Another metric we collected was the overall programming language popularity in this region. We aggregated the repository collections and counted the number of occurrences for each language, ignoring any projects that did not have the programming language listed (8,459 or 24% did not have one listed). Figure 6.7 shows the number of projects that use each language in a bar chart where usage count is greater than 1,000, with JavaScript leading the way. We note that this information is what comes back from GitHub, it is important to remember that many projects will likely have more than one associated programming language.

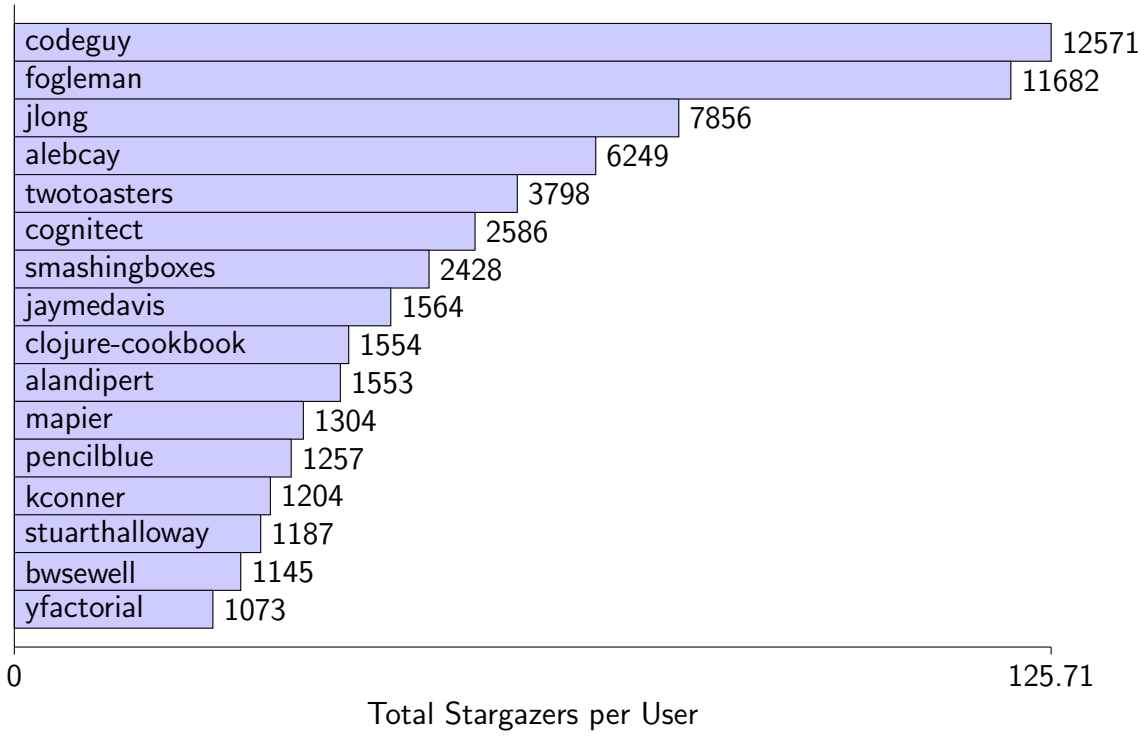

Figure 6.6: This figure shows the number of followers that each user has (where the count of followers is greater than 1000).

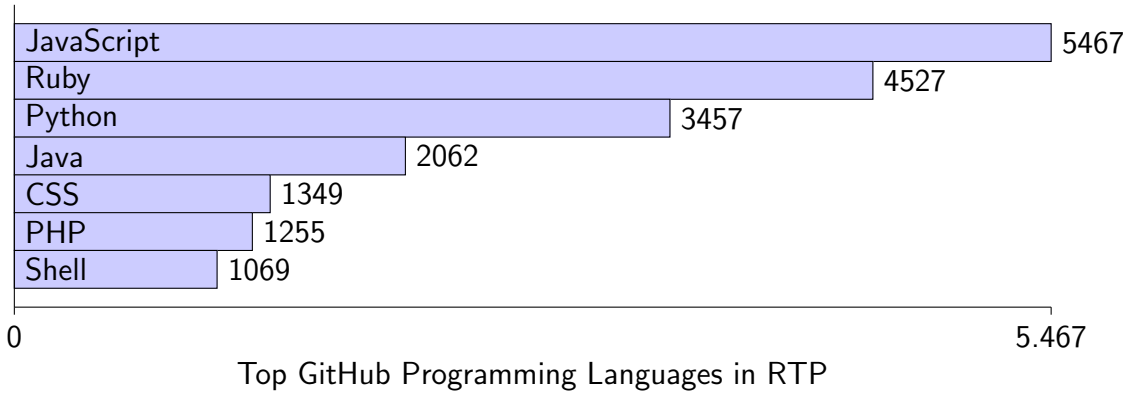

Figure 6.7: This figure shows the most popular programming languages in the RTP region, where the usage count is greater than 1000.

6.6 Average Life of a RTP Project

One of the additional data points that we decided to collect was what the average life of a GitHub project owned by a user in RTP looked like (question fifteen). We found

```
> db.githubRTPUsersRepos
        .find({},
        {"owner.login":1,"created_at":"1","name":1})
    .sort({"created_at":1})
    .limit(1)
> {
        "_id" : ObjectId("52bedd67bd3543677400823a"),
        "owner" : { "login" : "bscofield" },
        "name" : "depth-charge",
        "created_at" : ISODate("2008-02-27T12:42:10Z")
  }
```

Figure 6.8: Finding the oldest RTP repository from MongoDB's shell (query and result).

out that the oldest RTP repository was created less than a month after GitHub was founded, "depth-charge". Figure 6.6 shows the query used to find this information in our local MongoDB database. This particular repository was created 8 years ago and hasn't been updated since.

Finally, we also found that the average life of a RTP project is 102 days. This was calculated by subtracting the created date from the last commit date and then averaging the entire gathered list. We found that the shortest project was 1 day and that there are plenty of projects that are still ongoing and active.

6.7 Threats to Validity

Due to the quality of data that we were able to retrieve for this project, there are several threats to validity that must be mentioned. As already discussed, it is important to remember that the location field in GitHub is not required, hence there is likely a large population of users that are left out of our research because they haven't given a location or have possibly given an inaccurate one. This was a limitation with our research due to the fact that we were heavily interested in querying for users

that lived in such a specific location. Additionally, it was difficult for us to find out the exact number of commits per user. This is because users are able to enter free text into the authors field when committing a change in the GitHub shell (command line). This impacted the results from the mining for overall activity in the RTP region. We looked into pulling data from other sources to supplement the information from GitHub. An example source being LinkedIn, which was unsuccessful due to restrictions with the licensing of their REST API.

Chapter 7

Conclusion

The purpose of this research was to shed light on the open source activities in the Research Triangle Park region using information mined from GitHub. We began the data collection phase by gathering metrics around how many people in the RTP area were involved in open source projects (from both personal and organizational levels). This information was then used to explore repositories developed out of the RTP area, looking into characteristics such as the programming languages used, their associated activity, and current relevance of the various projects. Lastly, we explored how RTP users were involved in the social media aspects of GitHub. In summary, we found that, as a whole, the RTP region is not heavily involved in the open source community, but we were able to identify a number of users that were prominent on the platform both from the single user and organizational perspective.

As part of future research, it would be interesting to extend these experiments to cover other locations with a similar technology industry profile, providing additional context for the current results and providing insight into how open source development differs in different parts of the United States and in technology hubs in other countries. It would also be interesting to add a temporal aspect to this research, exploring how open source development activities change over time. Teams and individual developers could also use this work to find out which cities have the most users,

repositories, open source activity, and overall popularity, potentially by providing this data through a portal. Lastly, it would also be interesting to compare findings such as the most popular programming languages to the job requirements in the local markets to see if any trends can be identified. The scripts developed for this thesis, used as part of the data collection and analysis phases, are themselves available on GitHub under an open-source license.

BIBLIOGRAPHY

[1] "Research Triangle Region - NC." [Online]. Available: http://www.researchtriangle.org/about-rtrp

[2] "GitHub," Jan. 2016. [Online]. Available: https://github.com

[3] G. Gousios and D. Spinellis, "GHTorrent: Github's data from a firehose," in *2012 9th IEEE Working Conference on Mining Software Repositories (MSR)*, Jun. 2012, pp. 12–21.

[4] G. Gousios, "The GHTorrent dataset and tool suite," in *2013 10th IEEE Working Conference on Mining Software Repositories (MSR)*, May 2013, pp. 233–236.

[5] "Masters thesis scripts - lindseyklanier." [Online]. Available: https://github.com/lindseyklanier/masters_thesis

[6] R. Weddle, "Research Triangle Park - North Carolina Digital History." [Online]. Available: http://www.learnnc.org/lp/editions/nchist-recent/6177

[7] "North Carolina League of Municipalities." [Online]. Available: http://www.nclm.org/resource-center/municipalities/Pages/By%20County.aspx

[8] "Durhams American Underground Leads Nation on Entrepreneurial Diversity, Sees Major Increase in Funding | American Underground." [Online]. Available: http://americanunderground.com/durhams-american-underground-leads-nation-on-entrepreneurial-diversity-sees-major-increase-in-funding/

[9] "4 U.S. regions rivaling Silicon Valley." [Online]. Available: http://mashable.com/2015/10/05/next-silicon-valley-us-cities/#NseaXjYYWSqi

[10] J. Kotkin, "America's Fastest- And Slowest-Growing Cities - Forbes." [Online]. Available: http://www.forbes.com/sites/joelkotkin/2013/03/18/americas-fastest-and-slowest-growing-cities/#4d4e74961acb

[11] "American Underground." [Online]. Available: http://americanunderground.com/

[12] "The Research Triangle Park Organization," Jan. 2016. [Online]. Available: http://www.rtp.org/about-us/

[13] "The Frontier." [Online]. Available: http://www.rtp.org/about-us/the-frontier/

[14] "NC State of Technology - 2016 Industry Report." [Online]. Available: http://www.nctechnology.org/resources/sotir.aspx

[15] "Why are my commits linked to the wrong user? - User Documentation." [Online]. Available: https://help.github.com/articles/why-are-my-commits-linked-to-the-wrong-user/

[16] "MongoDB for GIANT Ideas." [Online]. Available: https://www.mongodb.com/

[17] "GHTorrent (@ghtorrent) | Twitter." [Online]. Available: https://twitter.com/ghtorrent

[18] "GitHub - Wikipedia, the free encyclopedia." [Online]. Available: https://en.wikipedia.org/wiki/GitHub

[19] J. Cabot, "Should developers believe reports based on GitHub mining results?" Mar. 2016. [Online]. Available: http://modeling-languages.com/believe-research-github-mining/

[20] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian, "The Promises and Perils of Mining GitHub," in *Proceedings of the 11th Working Conference on Mining Software Repositories*. ACM, 2014, pp. 92–101. [Online]. Available: http://dl.acm.org/citation.cfm?id=2597074

[21] Y. Takhteyev and A. Hilts, *Investigating the geography of open source software through GitHub*. Working Paper, 2010. [Online]. Available: http://takhteyev.org/papers/Takhteyev-Hilts-2010.pdf

[22] D. Rusk and Y. Coady, "Location-Based Analysis of Developers and Technologies on GitHub," in *2014 28th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, May 2014, pp. 681–685.

[23] A. Begel, J. Bosch, and M. A. Storey, "Social Networking Meets Software Development: Perspectives from GitHub, MSDN, Stack Exchange, and TopCoder," *IEEE Software*, vol. 30, no. 1, pp. 52–66, Jan. 2013.

[24] "The Issue 32 incident An update." [Online]. Available: http://gousios.gr/blog/Issue-thirty-two/

[25] "PyMongo 3.2.1 Documentation PyMongo 3.2.1 documentation." [Online]. Available: https://api.mongodb.org/python/current/

[26] “Aggregation MongoDB Manual 3.2.” [Online]. Available: https://docs.mongodb.org/manual/aggregation/

[27] “plotly.” [Online]. Available: https://plot.ly/python/