Towards Understanding the Public Gists on GitHub

by

Weiliang Wang
B.Sc., Southeast University, 2013

A Master Project Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Towards Understanding the Public Gists on GitHub

by

Weiliang Wang
B.Sc., Southeast University, 2013

Supervisory Committee

Dr. Daniel M. German, Supervisor
(Department of Computer Science)

Dr. Alex Thomo, Departmental Member
(Department of Computer Science)

Supervisory Committee

Dr. Daniel M. German, Supervisor
(Department of Computer Science)

Dr. Alex Thomo, Departmental Member
(Department of Computer Science)

ABSTRACT

GitHub is a popular source code hosting site which serves as not only a collaborative coding platform but also a social network. Various unique features of GitHub have greatly facilitated developers' collaboration, communication, and coordination, which has given rise to many research in different angles. One of GitHub features is Gists, which is expected to be a tool for instantly sharing code, notes, and snippets. However, it hasn't been paid much attention by researchers.

The goal of this study is to understand what GitHub Gists look like and how they are used. It was composed of three parts. The first part was a quantitative analysis of the metadata of 562,993 GitHub users, 618,393 Gists and 793,891 files contained in those Gists. For the second part, manual coding was conducted for the content of 400 randomly selected Gists. The third part was qualitative analysis of the searched results from Google and Twitter about how individuals use Gists.

The study result shows that only a small portion of GitHub users use Gists. Most of Gists contain small snippets of source code, but a great variety of Gists content was also discovered. The majority of Gists were not updated frequently, and users rarely fork or comment other users' Gists. Gists are being used in lots of ways, from saving snippets of code or notes, sharing files, to drafting their writings.

Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **iv**

List of Tables **vi**

List of Figures **vii**

Acknowledgments **viii**

Dedication **ix**

1 Introduction **1**

2 Related Works **4**

2.1 Version Control History and GitHub 4

2.2 Recent Research on GitHub 5

2.3 GitHub Gists 6

2.3.1 Related Tools 6

2.3.2 The Features of Gists 6

3 Methodology **7**

3.1 Research Questions 7

3.2 Data Collection 8

3.2.1 GitHub REST API 8

3.2.2 GHTorrent 8

3.2.3 Data Retrieving 8

3.3 Quantitative Analysis of Gists Metadata and Content 9

3.3.1 Analysis of Gists Metadata 10
3.3.2 Analysis of Gists Files 11
3.4 Qualitative Analysis of Gists Content 11
3.5 Qualitative Analysis of Users' Discussion Regarding Gists 12

4 Results 14
4.1 RQ1. What do Gists look like? 14
4.1.1 Users and Their Gists 14
4.1.2 Contents of Gists 14
4.1.3 Activity 21
4.1.4 Manual Inspection of Gists Content 21
4.2 RQ2. How are Gists being used? 24
4.3 Summary 26

5 Limitations and Future Work 28
5.1 Limitations 28
5.2 Future Work 28

6 Conclusion 30

A Published Paper Based on This Project 31

Bibliography 42

List of Tables

Table 3.1 Data sample used in this study. 9
Table 3.2 Description of Gist metadata. 10
Table 3.3 Description of Gist file metadata. 10
Table 3.4 Description of Gist labels in terms of content. 12
Table 3.5 Description of Gist labels in terms of files relationships. 12

Table 4.1 Comparison of GitHub users having at least one public Gist and those having not any. 14
Table 4.2 Distribution of Gists by number of files contained. 15
Table 4.3 Gist files recognized by CLOC 20
Table 4.4 Distribution of number of forks per Gist. 22
Table 4.5 Distribution of number of comments per Gist. 22
Table 4.6 Distribution of content labels of Gists. 23
Table 4.7 Distribution of relationships between files in each Gist. 24

List of Figures

Figure 4.1 Cumulative distribution of the number of Gists owned by user. 15
Figure 4.2 Distribution of MIME types by Gist files. 17
Figure 4.3 Distribution of languages by Gist files. 18
Figure 4.4 Cumulative distribution of size of files in Gists. 19
Figure 4.5 Cumulative distribution of number of lines per Gist. 19
Figure 4.6 Distribution of number of SLOCs of source files in Gists. 20
Figure 4.7 Distribution of number of commits per Gist. 21
Figure 4.8 Distribution of days between creation and latest update per Gist. 22

ACKNOWLEDGEMENTS

I would like to thank:

my supervisor, **Dr. Daniel M. German**, for mentoring, encouragement, patience and understanding;

my parents, **Ruby**, **Tony**, for your love and understanding about all of my decisions I've made;

my best friends, **Yingsi** and **Hanbin**, for making me feel not alone;

the companies, **Isolation Network** and **CopperLeaf Technologies**, for giving me a chance of experiencing the industrial world before graduation;

some awesome movies and TV shows such as **True Detective**, **House of Cards**, **Breaking Bad**, for expanding my boundaries of thinking about everything;

and everyone else who has ever helped me.

DEDICATION

I dedicate this report to my parents.

Chapter 1

Introduction

GitHub[1] is a popular source code hosting site. It enhanced the version control tool Git[2] by combining the features of both software engineering and social networking, including event feeds, pull requests, code reviews, and an issue tracking mechanism[3], greatly facilitating developers' collaboration, communication and coordination.

Gist is one of the many features of GitHub, which allows users to instantly share code, notes, and snippets. GitHub illustrates[4] Gist use as follows:

"Gists are a great way to share your work. You can share single files, parts of files, or full applications...Every gist is a Git repository, which means that it can be forked, cloned, and manipulated in every way."

Since how a technology is supposed to be used might differ from how it is actually used, it would be interesting to know how Gist is actually being used, and how that compares to the way that GitHub expects it to be used. However, Gist has been paid little attention by researchers, and there's no existing Gists dataset available for research. Thus, a lot of questions about Gists remain unanswered. For example, how many GitHub users are using gists? Do users collaborate on Gists? What are the contents of Gists about? How large are Gists?

A noteworthy trend is that several third-party applications have been developed to support creation, management and sharing of Gists. For example, both *GistBox*[5] and *Gisto*[6] are applications that help users better organize their Gists by adding ad-

[1] `https://www.github.com`

[2] `https://en.wikipedia.org/wiki/Git_(software)`

[3] `https://github.com/features`

[4] `https://help.github.com/articles/about-gists/`

[5] `http://www.gistboxapp.com/`

[6] `http://www.gistoapp.com/`

ditional features such as searching, tagging and sharing Gists; *GistBox Clipper* allows users to create Gists from any web page; many blog sites like *WordPress*[7] support Gists embedding either by its URL or its ID[8]; many popular IDEs or text editors support Gists creation through plugins, such as Visual Studio[9], *IntelliJ IDEA*[10], *sublime text*[11], etc. All these threads imply that gists are gaining popularity among GitHub users. It's worth to explore Gists and how developers are using them.

In this report, an empirical study of GitHub Gists is presented. With the goal of getting a picture of what Gists look like and how they are being used, the following two research questions were addressed:

- **RQ1. What do Gists look like?**

  The purpose of this research question is to provide basic information on the appearance of Gists in terms of different aspects, such as the size, files count, Gists collaboration, and so on.

- **RQ2. How are Gists being used?**

  This research question is to find out how individuals use Gists whether or not in the way that GitHub expects.

To answer **RQ1**, both qualitative and quantitative analyses were performed of Gists contents and metadata. For the quantitative part, 618,393 Gists and 793,891 files contained in those Gists were downloaded. Also, 562,993 GitHub users randomly selected were analyzed to know the proportion of GitHub users that are using Gists. For the qualitative part of the study, a manual inspection was performed on 400 randomly selected Gists.

To answer **RQ2**, the Web was searched for evidence of how users described their use of Gists, which included a search of Websites (including blog posts) and Twitter feeds.

Our results can be summarized as follows. Only a small portion of GitHub users use Gists. Gists are usually small snippets of source code, and they usually contain only one file. Gists content has a wide variety including data files (such as JSON, XML), binary files (such as images, audios, videos), and text files of various kinds

[7] `http://www.wordpress.com/`

[8] `http://en.support.wordpress.com/gist/`

[9] `https://marketplace.visualstudio.com/items/dbankier.vscode-gist`

[10] `https://www.jetbrains.com/idea/help/creating-gists.html`

[11] `https://github.com/bgreenlee/sublime-github`

(such as blogs, logs, letters, restaurant menus). The majority of Gists are not updated frequently, and users rarely fork or comment other users' Gists. People are using Gists for various purposes beyond source code, from creating reusable components, sharing files, saving notes or tops, to drafting writings.

The main contribution of this study is that it provided an initial insight into what Gists are and how they are used through exploratory data analysis. Though it's just scratching the surface of Gists, the study result can definitely inspire other researchers to conduct further research on Gists.

Chapter 2

Related Works

2.1 Version Control History and GitHub

Version Control Systems (VCS) were developed to record changes to a file or set of files over time so that a specific version can be recalled later. [1] The traditional Centralized Version Control Systems (CVCS) require developers to commit a change to a central repository, merge and resolve the conflicts. This mechanism leads to the downside where the entire history of the project lies in the central database, and if the central database becomes corrupted, all the history data would be lost. [1][2]

Distributed Version Control Systems (DVCS) relaxed the requirement of CVCSs to have a central, master repository. In a DVCS, each developer owns "first-class access" to the project repository so they can access the entire history data of the repository.[2]. DVCS has the potential to "make releasing, developing, and coordinating large software projects much less rigid than its exclusively centralized predecessor".[3]

Among all these DVCSs, Git has gained the most momentum. It began in 2005 as a revision management system used for coordinating the Linux kernel's development. Over the years, Git has evolved by "leaps and bounds" due to its functionality, portability, efficiency, and rich third-party adoption.[4]

Using Git as a back end to host open source projects, many Web-based applications enhanced project management functionality by adding rich user-friendly interfaces, which provide a convenient way for developers to set up repositories, clone existing projects and commit their contributions.[4] These applications also lay more emphasis on the social aspect of software engineering.

Among all these applications, GitHub is the most popular one with more than

12.2 million users[1]. GitHub not only allows users to *star*[2] or *watch*[3] repositories to keep track of projects they find interesting, but also allows users to follow other users to see what other people are working on and who they are connecting with. GitHub also supports team management using the *organizations* feature, projects discovery using *explore* feature, bugs tracing using *issues*, etc.[4] By integrating these social features into the version control system, the communication and coordination among developers is greatly enhanced.[5]

2.2 Recent Research on GitHub

GitHub introduces a new open source environment that has given rise to research from many different angles. GitHub is a huge data pool that contains not only numerous projects but also developers' profiles and developers' activities such as their contributions to projects and their interactions with other developers. More and more researchers have jumped into this data pool, trying to discover some interesting patterns or good stories in either software engineering or social networking.

Some researchers have been working on helping employers find technical experts by analyzing into the developers' profile and their activities.[6][7][8][9] Others have focused on the source code in project repositories. For example, Bissyande et al. (2013) took advantage of the rich data on GitHub by examining the "popularity", "interoperability", and "impact" of various programming languages measured in different ways, such as lines of code, development teams, issues, etc.[10] There are also some researchers who tried to discover patterns in developers' collaboration and interaction, such as how developers assess each other and find proper collaborators[11][12][13], the herding phenomena on GitHub[14], the relation between developers' behavior on GitHub and other Q&A websites like StackOverflow[5][15], etc.

However, no scientific study that focused on GitHub Gists have been found.

[1] `https://github.com/about/press`

[2] `https://help.github.com/articles/about-stars/`

[3] `https://help.github.com/articles/watching-repositories/`

[4] `https://help.github.com/articles/be-social`

[5] `http://www.stackoverflow.com/`

2.3 GitHub Gists

2.3.1 Related Tools

There are some other snippets management tools similar to GitHub Gists such as *pastebin*[6], *snipt*[7], *codepen*[8], *dabblet*[9], etc. However, GitHub Gists is the only one that manages snippets using version control. Over these years, the features of GitHub Gists are being improved and updated to be more user friendly[10] [11].

2.3.2 The Features of Gists

Gists are expected to be small snippets of code or text that utilize the version control tool Git for creation and management. Wikipedia provides a good explanation of the benefits of Gists as follows[12]:

"Gist builds upon that idea by adding version control for code snippets, easy forking, and SSL encryption for private pastes. Because each 'gist' is its own Git repository, multiple code snippets can be contained in a single paste and they and be pushed and pulled using Git. Further, forked code can be pushed back to the original author in the form of a patch, so pastes can become more like mini-projects."

In addition, GitHub also provides a powerful Web-based editor to create or modify Gists, which makes it possible to work on Gists without Git. It also supports comments on Gists and provides a Web service that makes Gists embeddable in a Web page. All these features make Gists very flexible, functional and user friendly, helping users better manage their gists and share them.

[6] http://pastebin.com/

[7] https://snipt.net/

[8] http://codepen.io/

[9] http://dabblet.com/

[10] https://github.com/blog/1837-change-the-visibility-of-your-gists

[11] https://github.com/blog/1850-gist-design-update

[12] http://en.wikipedia.org/wiki/GitHub#Gist

Chapter 3

Methodology

3.1 Research Questions

There are two main research questions to be addressed in this study.

- **RQ1. What do Gists look like?**

  The purpose of this research question is to provide basic information of Gists using different metrics, such as Gists size, files count in Gists, users' collaboration on Gists, and so on. We retrieved such information from the metadata and contents of Gists, and statistically analyzed the distribution of Gists in terms of these different metrics.

- **RQ2. How are Gists being used?**

  We used Google to search Web pages and tweets on Twitter for people's description about how they used Gists. We combined these information with the results of the previous question to answer this research question.

Our methodology was a mix of quantitative and qualitative analysis. The study involved three main parts with the first two parts to answer **RQ1** and the third to answer **RQ2**:

1. A quantitative analysis of the metadata and contents of sampled Gists that were created by a large sample of GitHub users.

2. A qualitative analysis of a small sample of Gists. The Gists files were manually inspected to infer the purpose of each Gist.

3. A qualitative analysis of searched results of people's description about how they used Gists.

3.2 Data Collection

3.2.1 GitHub REST API

The data on GitHub can be accessed through GitHub REST API[1] which is based on the REST architecture[2]. Despite the adaptability of GitHub API, it has some limitations on the number of requests the users can make. GitHub describes the rate limiting as follows: "For requests using Basic Authentication or OAuth, you can make up to 5,000 requests per hour. For unauthenticated requests, the rate limit allows you to make up to 60 requests per hour."[3] Thus, in order to get a higher query speed, each request for data retrieval must be signed with valid GitHub user tokens.

3.2.2 GHTorrent

In order to make it convenient enough for developers and researchers to acquire and analyze GitHub data, Gousios et al. developed *GHTorrent*[4], which is a project that aims to offer a mirror of GitHub's data and event streams to the research community as a service in a scalable manner.[16] It provides data dumps of GitHub users, repositories, commits, comments, etc. in both MySQL and MongoDB format. Unfortunately, GHTorrent does not provide dataset of gists, so for this research Gists had to be retrieved through GitHub's REST API.

3.2.3 Data Retrieving

Since the goal is to get a whole picture of the use of Gists, any Gist from any GitHub user would of interest. The Gists data were collected by leveraging GHTorrent and GitHub REST API in the following approach.

First GHTorrent was used to obtain the most recent dataset of GitHub users at the time of this study (the MongoDB dump with date 2014-03-29[5]) which contained

[1] `https://developer.github.com/v3/`

[2] `https://en.wikipedia.org/wiki/Representational_state_transfer`

[3] `https://developer.github.com/v3/#rate/limiting`

[4] `http://www.ghtorrent.org/`

[5] `http://ghtorrent.org/downloads.html`

the metadata of all GitHub users. These user data had to be filtered because of two main concerns. First, there may exist duplicated users in the dataset. Second, there exist non-individual user accounts in the dataset (Organizations are included as users in the GHTorrent user dataset[17]). A user was determined as non-individual when the value of the *ext_ref_id* attribute for that user row in MySQL dump was NULL or empty. After filtering out the non-individual users and duplicates, 2,572,370 valid users were left.

Then I started querying the metadata and content of the Gists from these users by using GitHub's API. I successfully downloaded the metadata of 618,393 Gists as well as 793,891 files contained in these Gists. (Note that it was not recorded how many users these 618,393 Gists were from.)

Also, in order to know the percentage of GitHub users who used Gists, I randomly selected 562,993 users from all valid users. Among all these users, 32,786 (5.8%) of them were found to have at least one Gist and these 32,786 users had 144,073 Gists in total. Table 3.1 lists all the data used in this study.

Description	**Count**
Total valid GitHub users	2,572,370
Downloaded Gists	618,393
Files of downloaded Gists	793,891
Sampled users	562,993
Sampled users having Gists	32,786
Gists of sample users	144,073

Table 3.1: Data sample used in this study.

Table 3.2 lists the metadata associated with each Gist that was downloaded.

One Gist can contain one or more files. The metadata of each individual file is also included in the Gist metadata, and it's listed in Table 3.3.

3.3 Quantitative Analysis of Gists Metadata and Content

With the goal of finding some patterns and characteristics of Gists so as to get a overview of what Gists look like. I quantitatively analyzed the Gists data that were downloaded. Not only did I analyze the metadata of the sampled 618,393 Gists but

Gist Metadata	**Description**
Gist identifier	The identifier of the Gist that is unique in the system
Description	How the owner described the Gist
Create date	The date when the Gist was created
Last update date	The most recent date when the Gists was updated
Forks count	Number of forks for the Gist made by other users
Commits count	Number of commits pushed to the Gist
Commits history	Number of additions and deletions in the commits
Comments count	Number of comments for the Gist
Files count	Number of files contained in the Gist

Table 3.2: Description of Gist metadata.

Gist File Metadata	**Description**
Filename	The name of the Gist file which includes the extension.
Language	The languages used in the file
MIME Type	MIME Type of the file such as application/json etc.
File size	Size of the file in Bytes

Table 3.3: Description of Gist file metadata.

also inspected the actual content of the 793,891 Gist files.

3.3.1 Analysis of Gists Metadata

The quantitative analysis for Gists metadata was based on the a series of sub-questions:

- How many Gists does each user own?
- How many files are there in a Gist?
- How large is a Gist?
- What languages are used in Gists?
- How many commits/forks/comments are there per Gist?
- How often do users update their Gists?

3.3.2 Analysis of Gists Files

I first analyzed the size metadata of all Gists files downloaded. Then for each file whose MIME type is "text" or "application", I calculated its number of lines. Further analysis was made by taking advantage of the source code analyzing tool *CLOC*[6]. *CLOC* is able to detect uniqueness of text files based on *Digest::MD5*[7], as well as calculate source lines of code (SLOC[8]) for source files in many widely used programming languages[9].

3.4 Qualitative Analysis of Gists Content

In order to make the qualitative analysis of actual contents of Gist files, I randomly chose 400 from those 618,393 Gists and tried to manually analyze them by coding them. For each of these 400 Gists, I started reading through it to get a general idea of its content and then labeled it. Since a Gist is composed of one or more files, these Gists were manually labeled in terms of two aspects: the content of the Gist as a whole and the relationships between all the files of the Gist.

- The content of the whole Gist

  I manually assigned labels to each Gist based on its description, key words in Gist files and personal experience. A Gist could have more than one labels and these labels are not necessarily exclusive as are shown in Table 3.4.

- Relationships between files in a Gist

  Since a Gist may contain more than one file, I assumed there could exist some relationships between the files in each Gist. These relationships are listed in Table 3.5.

After finishing the data coding, statistical analysis was made against these labels.

[6] `https://github.com/AlDanial/cloc`

[7] `http://search.cpan.org/dist/Digest-MD5/MD5.pm`

[8] `https://en.wikipedia.org/wiki/Source_lines_of_code`

[9] `https://github.com/AlDanial/cloc#Languages`

Types of Gists Content	Description
Code	Source code.
Test	Test code.
Class	Only a class is defined.
Template	Coding example/pattern.
Command	Commands used in shell.
Function	Only one or several functions are defined.
Fragment	Several lines of code without complete functions/classes.
Note	Script without source code.
Log	Log files.
Configuration	Configure files used for executing code.
Diff	Diff files used for visualizing the changes in a file.
Documentation	Text tutorial documentations.
Data	Data stored in json, csv or other forms.
Blog	Technical blog in narrative format.
Non-technical	Notes without any technical content.

Table 3.4: Description of Gist labels in terms of content.

Types of Gist Files Relationships	Description
Single File	The Gist contains one file.
Independent	Files in a Gist are independent.
Reference	One file refers to classes, functions, etc. in another file.
Generation	One file is the input/output of another file.
Test	One file is to test another file.
Attachment	One file is the configuration or information of another file.

Table 3.5: Description of Gist labels in terms of files relationships.

3.5 Qualitative Analysis of Users' Discussion Regarding Gists

I also searched Web pages and tweets on Twitter for users' description about how Gists were used. All relevant Web pages, either official or unofficial, as well as the most recent comments were considered at the time this study was conducted.

1. Web Pages. In order to find relevant Web pages, Google Search Engine was used to perform the searching using the keyword "GitHub Gist". For each of the search results in the first 10 pages, I manually read through the Web page,

extracted the information about how Gists were used, and took notes.

2. Twitter Postings. Recently, Twitter has become a popular platform for developers to learn, share, discuss technical questions, so I chose Twitter as a information source to help understand how people are using gists. I performed the query using “GitHub Gist” as the searching keywords on Twitter Web search interface, and read all the searched tweets one by one. While reading these data, I took notes in terms of how they used Gists suggested in the posting.

Chapter 4

Results

4.1 RQ1. What do Gists look like?

4.1.1 Users and Their Gists

As Table 4.1 shows, among the randomly sampled 562,993 GitHub users, only 32,786 (5.82%) of them had at least one public Gist.

Users	**Count**	**Percentage**
Users having at least one public Gist	32,786	5.82%
Users having no public Gists	530,207	94.18%

Table 4.1: Comparison of GitHub users having at least one public Gist and those having not any.

For those 32,786 users having at least one public Gist, Figure 4.1 shows their distribution by the number of Gists per user. We can see that majority of users just have a small number of Gists. Specifically, 50% of the users have less than 5 public Gists. 70% have less than 10 public Gists. Only 4% of users have 30 or more public Gists.

4.1.2 Contents of Gists

The content of Gists can be analyzed in several different aspects, such as the number of files per Gist, their languages, their MIME type, their size. The results in this section are from the sampled 618,393 Gists which contained 793,891 files.

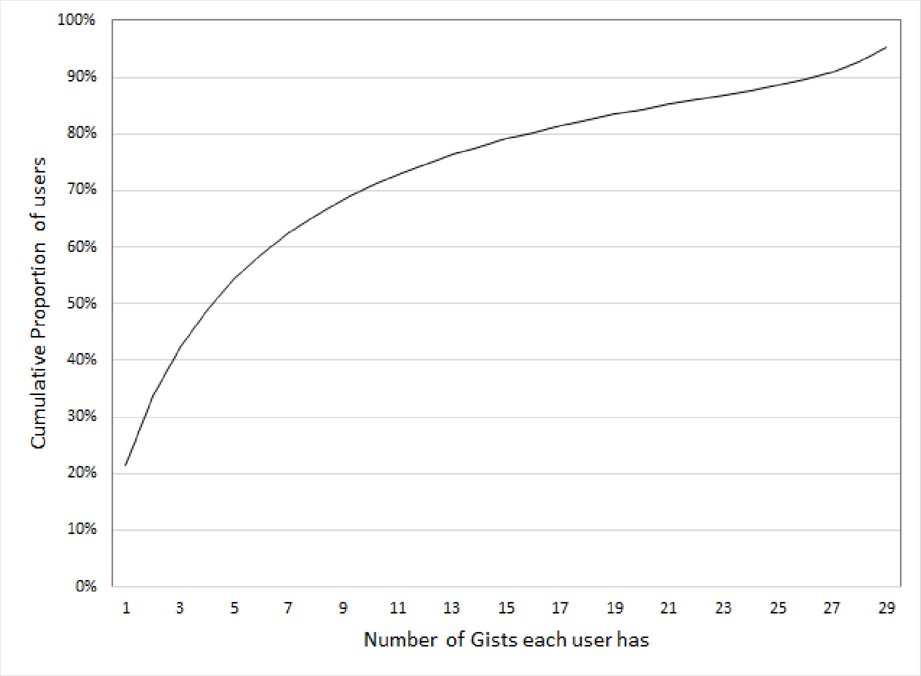

Figure 4.1: Cumulative distribution of the number of Gists owned by user.

Number of Files per Gist

GitHub doesn't limit the number of files in each Gist. Table 4.2 shows the distribution of all sampled Gists based on the number of files they contain. It shows that the majority of Gists (86.79%) contain only one file.

Number of Files Each Gist contains	Percentage of Gists
0	0.02%
1	86.79%
2	7.16%
3	3.16%
4	1.53%
≥ 5	1.35%

Table 4.2: Distribution of Gists by number of files contained.

It is expected that the majority of Gists contain only a few files, since Gists are supposed to record and share small code snippets. However, one surprising observation was that there also existed Gists containing a lot of files (for example, one with 1001 files).

Gist Files Types

I analyzed the types of Gist files by their MIME types and their languages.

Figure 4.2 shows the distribution of top 20 MIME types of all Gist files (there are other 93 falling into the "Others" category which is 0.4%). Gist files of MIME type "text/plain" dominates with 53.5% which actually includes a variety of file types such as .text files, .c files, and so on. Some programming languages are very common (indicated by the MIME type), such as application/javascript (9.7%), application/x-ruby (9.4%), application/python (5.6%), but we need to note that lots of programming languages fall into the text/plain category. Other common types include images (png 0.5%, gif 0.4%, jpeg 0.1%), markup language text (html 3.7%, xml 1.4%, yaml 0.5%), JSON data (1.7%), etc.

The Gist file metadata provides the language attribute for each file. 257 languages were discovered in total from our sample. Figure 4.3 shows the distribution of top 30. We can see 27.9% of Gist files don't have any languages. The most popular programming languages are Ruby (10.2%), JavaScript (9.7%), Python (5.6%), Shell script (5.3%). Lots of markup languages are very common too, which includes Markdown (4.6%), HTML (3.7%), JSON (1.9%), XML (1.5%), Diff (1.3%). The "Others" category represents the other 227 languages that are not shown in the chart and they account for 5.8%.

Size of Gist Files

As for the size of Gist files, I looked into three metrics: size in bytes, number of lines for files of text or application MIME type, and SLOCs for source files.

Figure 4.4 shows the distribution of Gist files in terms of their size in bytes. We can see that the majority of Gist files are around 1KB. The median value is 714 Bytes (quartiles are 287 and 1830 bytes). There exist outliers though: 0.17% of Gist files are over 1MB.

We also counted the number of lines for all the text files (783,326 in total). Figure 4.5 shows the distribution of these files by their number of lines. 87.1% of them have less than 100 lines. The median value is 23 lines (quartiles are 10 and 55 lines).

I ran *CLOC* script against all Gist files to calculate their SLOC (binary or empty files would be ignored by *CLOC*). *CLOC* recognizes a source file by its extension or its first line of code. If the file does not have a recognized extension or is not a

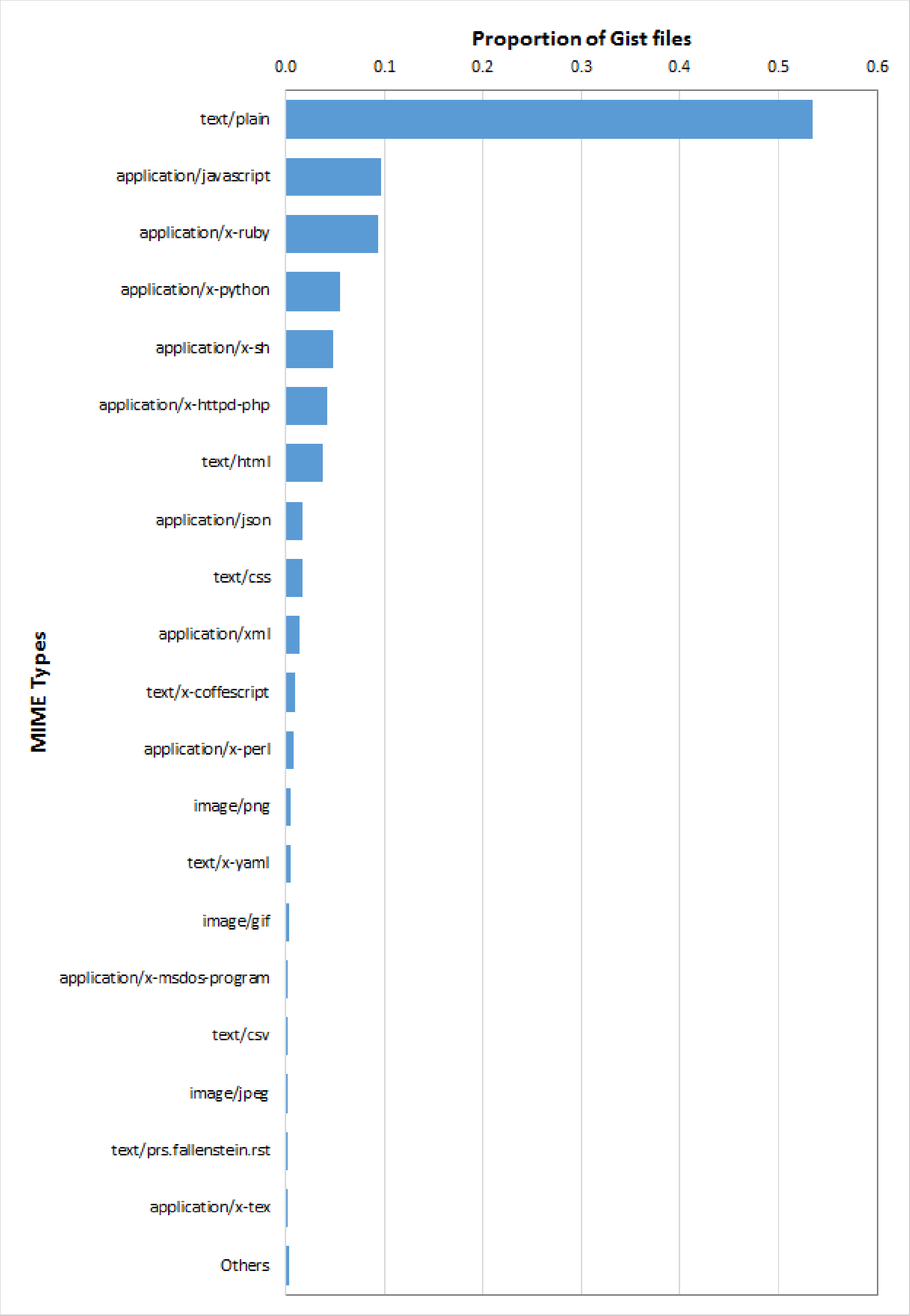

Figure 4.2: Distribution of MIME types by Gist files.

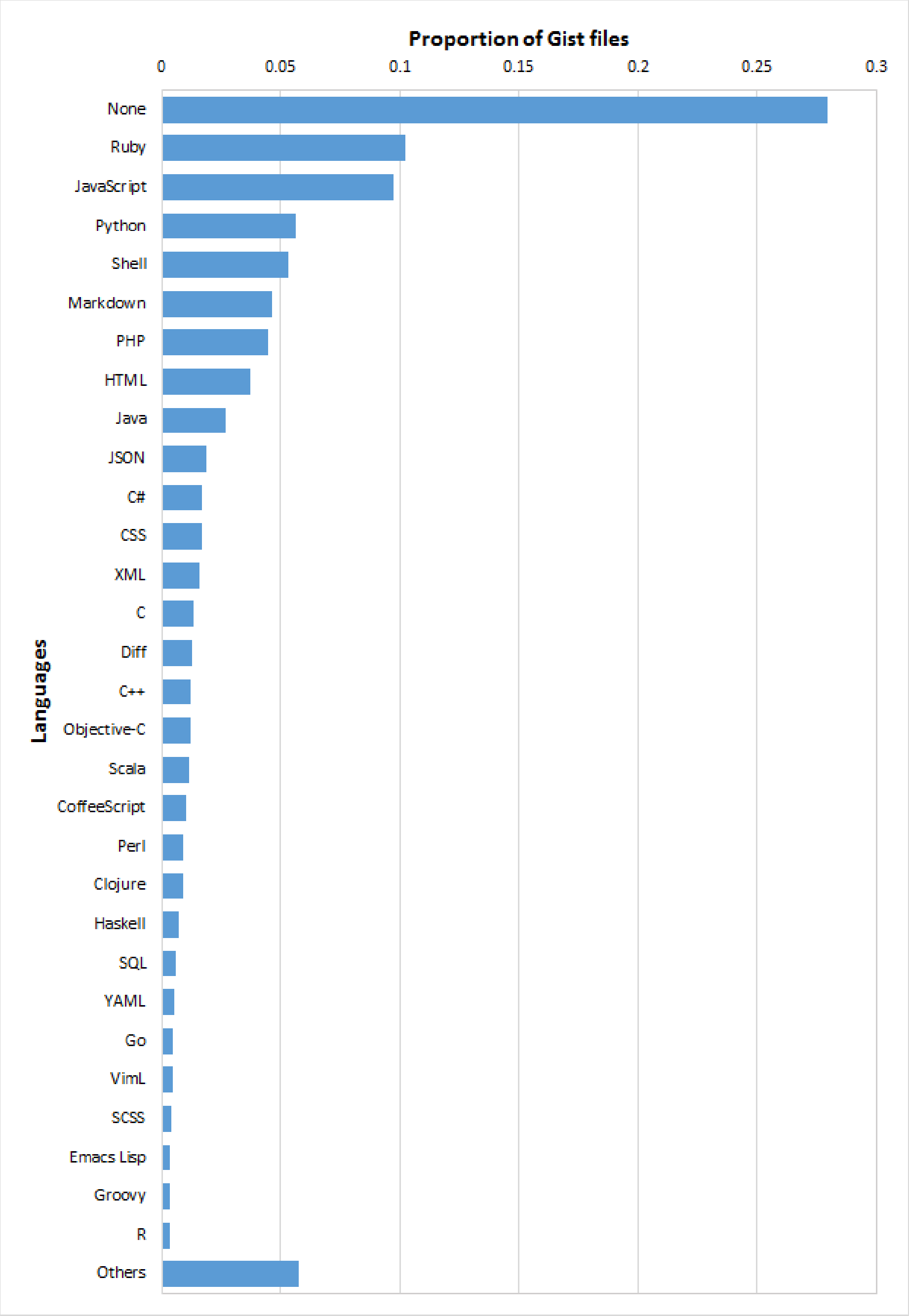

Figure 4.3: Distribution of languages by Gist files.

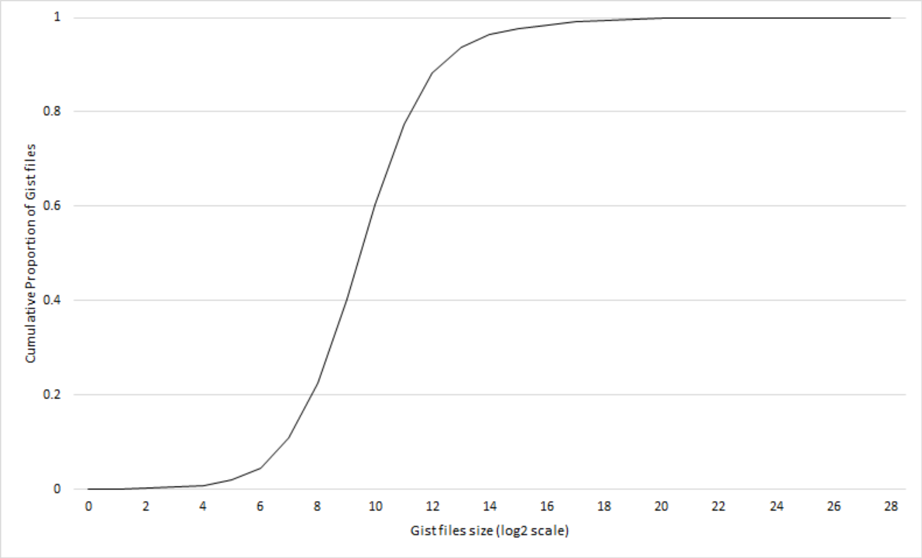

Figure 4.4: Cumulative distribution of size of files in Gists.

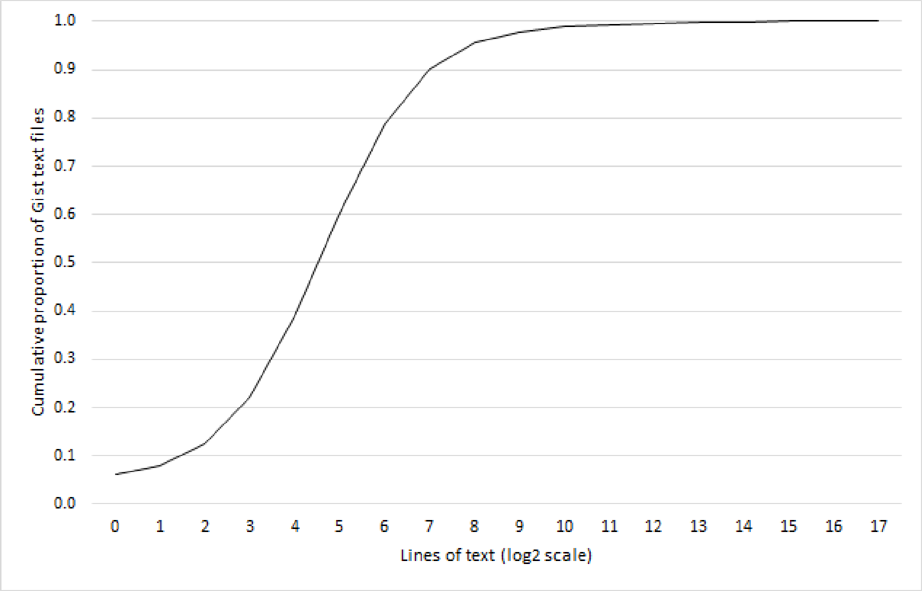

Figure 4.5: Cumulative distribution of number of lines per Gist.

recognized scripting language, it is ignored.[1]. Finally 778,806 text files (98.1% of all Gist files) were detected, from which 502,242 source files (64.5% of all Gist text files, 63.3% of all Gist files) were recognized as Table 4.3 shows. The distribution of source files by SLOC is shown in Figure 4.6. 54.8% of them have less than 20 SLOCs, and 92.1% have less than 100. The median value is 18 SLOCs (quartiles are 8 and 39

[1] https://github.com/AlDanial/cloc#How_it_works

SLOCs).

Files	**Count**
All files of 618,393 Gists downloaded	793,891
— text files	778,806
——— source files	502,242

Table 4.3: Gist files recognized by CLOC

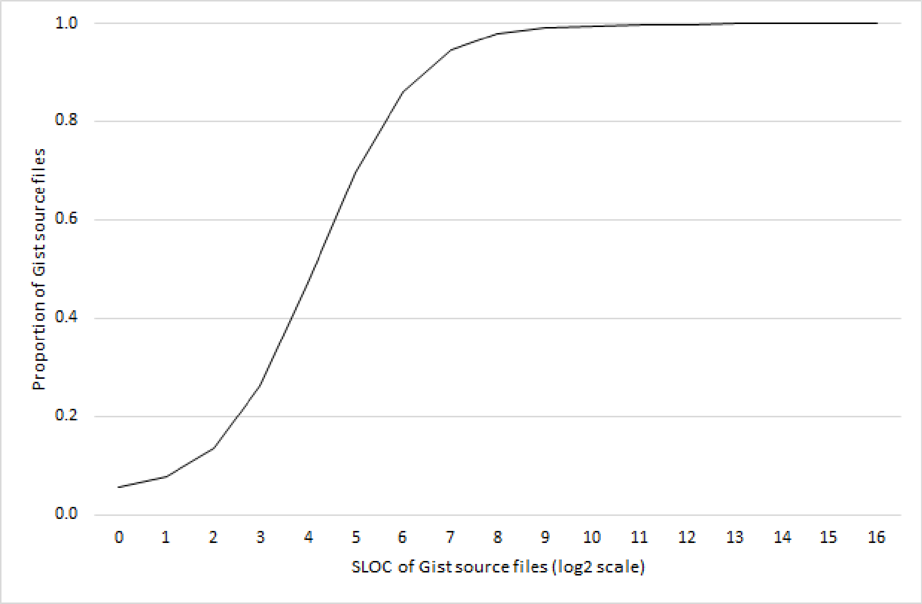

Figure 4.6: Distribution of number of SLOCs of source files in Gists.

Uniqueness of Gist Files

CLOC is able to detect uniqueness for text files based on *Digest:MD5*[2]. By taking advantage of that, I performed duplicate detection against all Gist files. Since *CLOC* does not detect duplicates for binary files, it only analyzed all the 778,806 text files from which 759,586 (97.5% of all text files) files were detected as unique. It means that 2.5% of text Gist files were duplicates.

[2] `https://github.com/gisle/digest-md5`

4.1.3 Activity

Since a Gist is like a GitHub repository, we can trace its user activities, such as number of commits, number of comments, times of being forked, etc. As Figure 4.7 shows, 62.9% of Gists had a single commit. 32.1% had 2 to 5 commits, and only 1.85% of Gists had more than 10 commits. This result is also verified by the distribution of days between the creation the latest update for each Gist, which is shown in Figure 4.8. We can see the creation and the latest update of 83.7% of Gists happened at the same day.

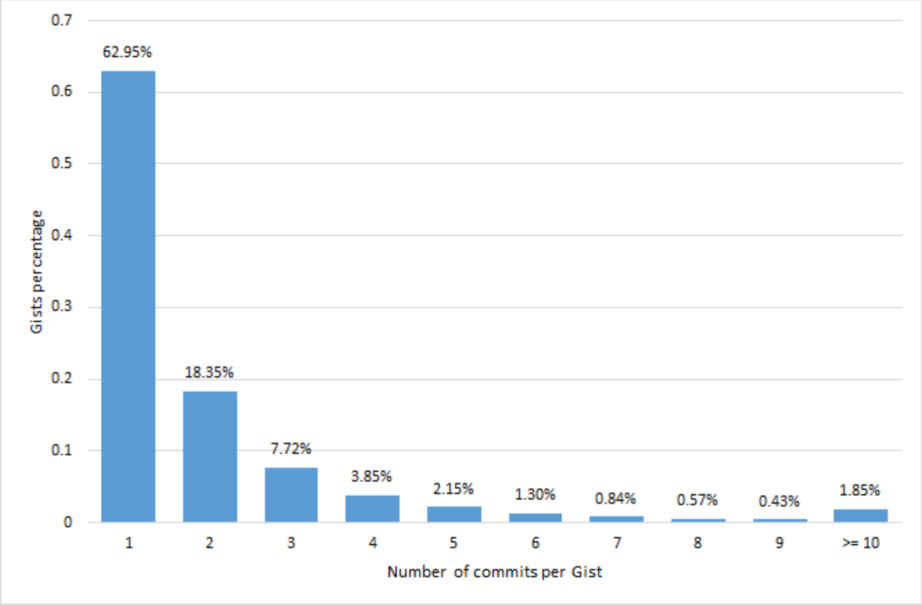

Figure 4.7: Distribution of number of commits per Gist.

In terms of forks, 94.6% of Gists have never been forked, and 4.0% of Gists have only been forked once. Only 0.7% of Gists have been forked more than 3 times (shown in Table 4.4). Table 4.5 shows the distribution of comments per Gist, from which we can see only 7.3% of Gists have been commented once or more. All of these observations show that majority of Gists are not active.

4.1.4 Manual Inspection of Gists Content

I manually inspected 400 randomly sampled Gists by assigning non-mutually-exclusive labels to each of them based on two aspects: the content of a Gist as a whole, and the relationships between files within a Gist.

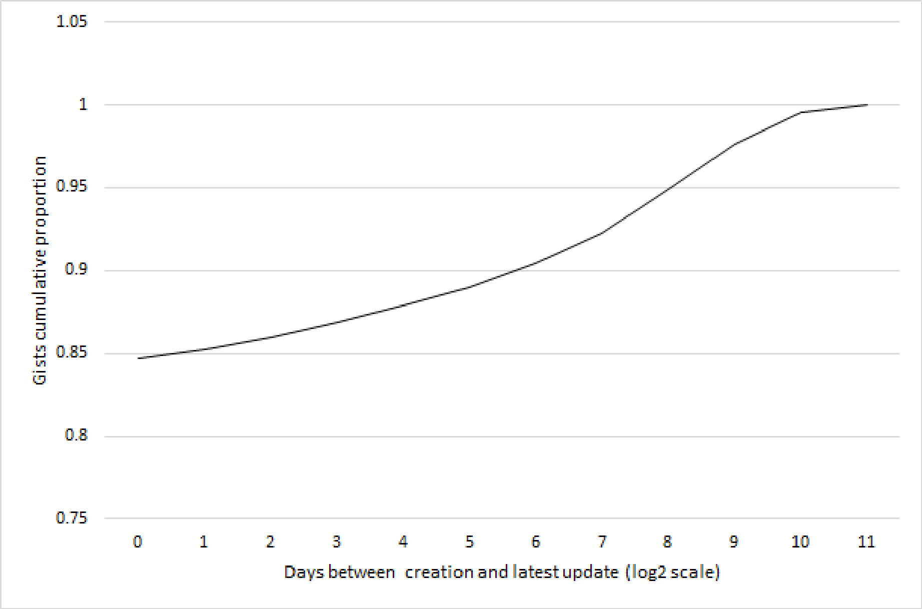

Figure 4.8: Distribution of days between creation and latest update per Gist.

Number of forks	Gists proportion
0	94.6%
1	4.0%
2	0.7%
≥ 3	0.7%

Table 4.4: Distribution of number of forks per Gist.

Number of comments	Gists proportion
0	92.7%
1	4.6%
2	1.3%
3	0.5%
≥ 4	1.0%

Table 4.5: Distribution of number of comments per Gist.

Gists Content

During categorizing these 400 Gists based on content with the labels shown in Table 3.4, It was observed that the contents of Gists were usually very short and demonstrated a big variety of uses. Code is as expected the most dominant use, but I also found lots of other uses, such as blogs, configuration information, logs, data, letters, and even a restaurant menu. I also found Gist written in several other languages besides English. Moreover, I observed that the MIME type or language of a Gist file might not be enough to understand a Gist. For example, source code and plain text are often mixed together within the same file.

The result of this categorizing is shown in Table 4.6. Note that the sum of all labels are more than 100% because a Gist can be assigned several labels. It's not surprising that most Gists (79.25%) contain code, including complete programs, functions, code fragments, classes, test code. However, other types of Gists content cannot be dismissed: Log (7.5%), Blog (1.5%), Non-technical (1.5%), etc.

Labels in terms of content	**Percentage of Gists**
Code	79.25%
Note	24.5%
Function	10.0%
Log	7.5%
Class	6.5%
Data	5.25%
Fragment	4.75%
Command	4.75%
Template	4.0%
Test	3.25%
Configuration	3.25%
Diff	1.75%
Blog	1.5%
Non-technical	1.5%
Documentation	0.5%
Empty	0.5%

Table 4.6: Distribution of content labels of Gists.

Files Relationships Within One Gist

A Gist can have more than one file (Among the 400 Gists sampled, 51 Gists contained multiple files). When a Gist has multiple files, I observed that they were usually related. The 400 Gists were categorized with labels shown in Table 3.5 to show the relationships between the files within each Gist, and the result is shown in Table 4.7. We can see most Gists contained only one file. For those having multiple files, they spread approximately uniformly among different categories. Only 3.25% of Gists have multiple files that are not related to each other.

Relationship between files of each Gist	**Percentage**
Single File	87.25%
Reference	4.25%
Generation	3.25%
Independent	3.25%
Test	1.25%
Attachment	0.75%

Table 4.7: Distribution of relationships between files in each Gist.

4.2 RQ2. How are Gists being used?

This question was answered based on the searched result of two sources: Websites and Twitter. I used "Github Gist" as keywords to search for related Web pages or tweets. While reading these content, I took notes in terms of suggested Gist uses. Finally I combined the results from Web pages and tweets, and classified them into several categories. Where appropriate, some quotations from the sampled Websites and Tweets are provided to illustrate our findings which are as follows.

Being embedded in blogs

One of the biggest blog websites, WordPress[3] has supported embedding Gists in the blogs[4]. Apart from that, Many other blogging platforms like Medium[5] also supporting such embedding soon.[6] This makes it easy to manage the code in blogs by saving it in Gists and simply referencing it in blogs by ID or URL.

"I think I might start using GitHub Gists in my blog posts. Thinking about how to integrate it smoothly."[7]

Version-controlled list

Gists can serve as version-controlled lists by taking advantage of the markdown rendering. Also the version control system can help to keep track of the change history of the lists, adding functionality at no cost. One good example is a popular blog that

[3] https://wordpress.com/

[4] http://crunchify.com/how-to-embed-and-share-github-gists-on-your-wordpress-blog

[5] https://medium.com/

[6] https://medium.com/the-story/yes-we-get-the-gist-1c2a27cdfc22

[7] https://twitter.com/BenNadel/status/157495925231714304

has been widely tweeted on Twitter. The blog illustrates how to maintain a to-do list using a private GitHub Gist authored by Carl Sednaoui[8].

"GitHub Gists are a great way to keep version-controlled lists (in this case, US states I've visited) `https://gist.github.com/dliggat/11003570`*"*[9]

"Im a HUGE fan of todo lists. They help me stay organized, prioritize my day and add structure to an otherwise chaotic day. I recently discovered what appears to be the best yet simplest way to keep a to-do list: a GitHub Gist."[10]

Saving notes/tips

Some people use Gists to save notes or tips such as technical information, learning outcomes tips, etc.

"I've been saving my learning as gists because blogging is a barrier: `https://gist.github.com/Greg-Boggs`*"*[11]

"Useful fiddles and gists collected from #AngularJS forum discussions `https://github.com/angular/angular.js/wiki/JSFiddle-Examples...`*"*[12]

Sharing files

Gists can also be used to share files with other people. Even if a Gist is private, it can still be visited by whoever has its URL. Sometimes private Gists can be a perfect way to share files. The following conversation on Twitter is a good example of this use.

User A to user B: *"Can you show me what your application.xml looks like for your ios+android game? Need to know what info is needed to compile."*

User B replied user A: *"I can put it on GitHub Gists sometime soon, remind me in a few days if I forget..."*[13]

[8] http://carlsednaoui.com/post/70299468325/the-best-to-do-list-a-private-gist

[9] https://twitter.com/dliggat/status/458090816930848768

[10] http://www.carlsednaoui.com/post/70299468325/the-best-to-do-list-a-private-gist

[11] https://twitter.com/gregory_boggs/status/483111455550877697

[12] https://twitter.com/AppsHybrid/status/481901294563901440

[13] https://twitter.com/stvr_tweets/status/483092598639185920

Drafting

Gists are also a perfect place to write a draft. For example, a user uses a Gist[14] to draft the wiki for a Google project *Cayley*[15]. This use can also be suggested by the following tweet:

"Every once in a while I think I wish I could "draft" a pull-request, issue, or comment on GitHub, then I remember that private Gists exist." [16]

4.3 Summary

Based on all the analysis above, our observations can be summarized as the following themes.

Majority of GitHub users don't use public Gists.

For those randomly sampled GitHub users, only 5.82% of them have at least one public Gist. For those who have public Gists, half of them have less than 5 Gists. Also, 86.79% of those sampled Gists only contain 1 file. All these results show that GitHub users generally don't use public Gists much.

Gists usually contain small files.

According to the metadata provided by GitHub, I found that the majority of Gist files were small. The median size was 714 Bytes. As for all the text files, 87.1% of them have less than 100 lines, and the median number of lines is 23. I also calculated the SLOC for all the source files, and 92.1% of them have less than 100 SLOCs with median as 18 SLOCs.

Gists content shows a great variety.

There are 113 MIME types and 257 programming languages discovered from all the Gist files downloaded. Although majority of Gist files are source files, I also observed data files (such as JSON, CSV, XML) and binary files (such as images, audios and even videos) in Gists. During our manual inspection against Gists content, I also found blogs, letters, and even restaurant menus.

[14] `https://gist.github.com/danbri/58db1297f3b488da9f86`

[15] `https://github.com/google/cayley/wiki`

[16] `https://twitter.com/nuclearsandwich/status/249213040610910209`

Users don't collaborate on Gists.

Gists are found mostly personal artifacts. Among all of the sampled Gists, 62.9% of them had a single commit (by the owner obviously), and only 1.85% had more than 10 commits. The creation and the latest update of 83.7% of Gists happened at the same day. 94.6% of Gists have never been forked by other users, and only 7.3% of Gists have been commented once or more.

Gists are being used for a variety of purposes.

Based on our searched results from Websites and Twitter, a great variety was discovered in terms of what people use Gists for: some people use it to create components that can be embedded in their blogs; some use it to keep track of some activities by a version controlled list; some save their notes such as learning outcomes in Gists; some share content with others simply by Gist URLs; some use it to draft their writings.

Chapter 5

Limitations and Future Work

5.1 Limitations

There exist threats to the validity of our results.

First of all, though private Gists are available to whoever has the Gist URL, they cannot be queried through the APIs GitHub provides. Therefore, we can only perform the analysis against public Gists, which introduces limitations to our results. The answer to our research questions might differ for private Gists from public ones.

Besides, the manual labeling in our qualitative analysis may introduce errors into our results. One researcher conducted the labeling by manually reading and interpreting the Gist content. The misinterpretation can introduce errors.

Also, since we have no control of how the search engines of Google and Twitter work, our searched results really depend on when the search was performed. This dependence makes the results transient and likely to change.

5.2 Future Work

This study is exploratory. I was just scratching the surface of GitHub Gists. One aspect I didn't reach is what users think about Gists. In the future, interviews of users should be involved. Some interview questions could be: what motivates a user to create a Gist? When do users choose to use Gists instead of GitHub repositories? What determines if a user create a public Gist or a private one? Do they collaborate much on Gists?

Lots of users use Gists to save reusable components, but where do these reusable components from? Are they from other people's blogs? Or the answers of Q&A websites like StackOverflow? Or GitHub repositories?

There also exist other tools for users to save and share code snippets. A comparison between Gists and these tools could also be an interesting research topic.

Chapter 6

Conclusion

In this study, with the goal of understanding what GitHub Gists look like and how they are used, I performed quantitative analysis against 618k Gists and 562k GitHub user, manual coding for the content of 400 randomly selected Gists, and manual analysis against the searched results from Google and Twitter about users' description of how they use Gists.

Our results can be summarized as follows:

What do Gists look like? Only a small portion of GitHub users use Gists. Gists are usually small snippets of source code, but a wide variety of Gists content was also discovered including data files (such as JSON, XML), binary files (such as images, audios, videos), and text files of various kinds (such as blogs, logs, letters, restaurant menus). Majority of Gists are not updated frequently, and users rarely fork or comment other users' Gists.

How are Gists being used? Although GitHub expects users to use Gists to instantly share code snippets, people are actually using Gists for various purposes: some people use it to create components that can be embedded in their blogs; some use it to keep track of some activities by a version controlled list; some save their notes such as learning outcomes in Gists; some share content with others simply by Gist URLs; some use it to draft their writings.

Gists can serve for various purposes by taking advantage of either its convenient editor or version control support. I hope it will become more and more popular in the future.

It's noteworthy that this study has been improved by my supervisor Dr. German and his other students, and published[18]. The published paper is attached in appendixA of this report.

Appendix A

Published Paper Based on This Project

What is the Gist? Understanding the Use of Public Gists on GitHub

Weiliang Wang, Germán Poo-Caamaño, Evan Wilde and Daniel M. German
Department of Computer Science, University of Victoria, Canada.
Email: {weiliang,gpoo,etcwilde,dmg}@uvic.ca

***Abstract*—GitHub is a popular source code hosting site which serves as a collaborative coding platform. The many features of GitHub have greatly facilitated developers' collaboration, communication, and coordination. Gists are one feature of GitHub, which defines them as "a simple way to share snippets and pastes with others." This three-part study explores how users are using Gists. The first part is a quantitative analysis of Gist metadata and contents. The second part investigates the information contained in a Gist: We sampled 750k users and their Gists (totalling 762k Gists), then manually categorized the contents of 398. The third part of the study investigates what users are saying Gists are for by reading the contents of web pages and twitter feeds. The results indicate that Gists are used by a small portion of GitHub users, and those that use them typically only have a few. We found that Gists are usually small and composed of a single file. However, Gists serve a wide variety of uses, from saving snippets of code, to creating reusable components for web pages.**

I. Introduction

GitHub has become one of the most important forges for software development. It provides an environment in which developers can collaboratively develop software using the Git version control system. GitHub enhances Git with features that greatly improve collaboration and communication between developers, including event feeds, pull requests, code reviews, and an issue tracking mechanism[1].

Gist is one of the many features GitHub provides to its users. GitHub defines[2] a Gist as follows:

> *"Gist is a simple way to share snippets and pastes with others. All Gists are Git repositories, so they are automatically versioned, forkable and usable from Git."*

How a technology is supposed to be used may differ from how it is actually used. This is especially true for disruptive technologies where people find innovative uses that were likely not envisioned by the creators. In a recent study [9], we described how GitHub repositories are being used for a variety of purposes, not just software engineering. For example, we found that repositories are being used to share data (the Tate Gallery in London uses GitHub to share the metadata of its entire collection[3]), and as data storage (the Boston Globe uses GitHub to host a historical mirror of its newspaper[4]). Building on this work, we would like to know how users are taking advantage of Gists, and if their use is similar to what GitHub intended, or if they are finding innovative ways to use them.

There has been a recent surge in third-party applications that support the creation and management of Gists. For example: the Chrome App *GistBox*[5] helps users save snippets of Web page text as Gists, as well as organize their collections of Gists; the *sublime-github plugin*[6] allows users to save snippets of code as Gists and share their Gists from within the Sublime editor—*gist.el*[7] is a similar module for Emacs and *gist-it*[8] for Atom; and *jist* is a command-line utility for managing multi-file Gists[9]. This activity seems to imply that Gists are gaining popularity among GitHub users, which motivated us to explore Gists and how developers used them.

In this paper, we present an empirical study of Gists in GitHub. The goal of this study was to understand what Gists are and how they are used. To do this, we attempted to address the following two research questions:

RQ_1. What do gists look like?
RQ_2. How are users using gists?

To answer *RQ_1*, we searched the Web for evidence of how users are describing their use of Gists. This included a search of Websites (including blog posts) and Twitter. To answer *RQ_2*, we performed a qualitative and quantitative exploratory study of Gists, including their contents and metadata. For the quantitative part of the study, we sampled 750k GitHub users and identified 762k Gists. For the qualitative part of the study, we performed a manual inspection of a sample of 398 Gists.

II. Related Work and Background

A. GitHub

Among the various distributed version control systems (DVCS) available, Git has gained the most momentum. The Git project began in 2005 as a version control system to coordinate the development of the Linux kernel. Due to its functionality, portability, efficiency, and rich third-party adoption, Git has evolved by "leaps and bounds" [14].

There are many Web-based applications that use Git as a back end to host free and open source projects. These

[1] https://github.com/features
[2] https://gist.github.com/
[3] https://github.com/tategallery/collection
[4] https://github.com/bcomdlc/bcom-homepage-archive
[5] http://www.gistboxapp.com/
[6] https://github.com/bgreenlee/sublime-github
[7] https://github.com/defunkt/gist.el
[8] https://atom.io/packages/gist-it
[9] http://charlesleifer.com/blog/jist-a-command-line-utility-for-managing-multi-file-multi-directory-private-gists/

978-0-7695-5594-2/15 $31.00 © 2015 IEEE
DOI 10.1109/MSR.2015.36

applications provide a convenient way for developers to create repositories, clone existing projects, and commit their contributions [14]. These applications also emphasize the social aspects of software engineering.

With more than 8.5 million users[10], GitHub has become one of the most popular of these applications. GitHub is an environment that combines social networking with distributed version control to enhance communication and coordination among software developers [5].

B. Recent Research on GitHub

The development of GitHub has prompted research from many different angles. GitHub stores data on developers' projects, their contributions to other projects, and their interactions with other developers. Some researchers are trying to help employers by analyzing the profiles and activities of developers on GitHub [2,11,16].

Other researchers have focused on the source code in project repositories. Bissyande et al. [1] took advantage of the rich data on GitHub, using lines of code, development teams, and issues that arose as measurements of popularity, interoperability, and impact of programming languages. Other researchers have tried to discover patterns in how developers asses each other and find collaborators [10,12,13], herd behaviour [3], and the relations between behaviour on GitHub and other Q&A Websites such as StackOverflow[11] [15]. In [9], we empirically analyzed the contents of repositories and discovered that GitHub is used for more than software engineering. While GitHub is meant to be a social coding platform, we discovered that a lot of activity within it is driven by developers who are working on their own. However, we have not found any scientific studies focusing on GitHub Gists.

C. Related Tools

There are other snippet management tools for software developers similar to GitHub's Gists, such as *pastebin*[12] and *snipt*[13]. Their existence seems to imply that the sharing and management of snippets is a growing concern among software developers. However, GitHub is the only one that stores the snippets using version control.

D. The Features of Gists

Gists are small snippets of code or text that are stored using Git. Wikipedia provides a good summary of the benefits of such an integration:[14]

> *"Gist builds upon that idea by adding version control for code snippets, easy forking, and SSL encryption for private pastes. Because each 'Gist' is its own Git repository, multiple code snippets can be contained in a single paste and they and be pushed and pulled using Git. Further, forked code can be pushed back to the original author in the form of a patch, so pastes can become more like mini-projects."*

In addition, GitHub makes it easy to create Gists. It also includes a powerful Web-based editor to modify them—it can completely isolate users who may not want to use Git. It also supports the ability to post comments on a Gist, and provides a Web service to retrieve Gists that typesets the contents so that they are ready to be embedded in a Web page.

III. Research Questions and Methodology

Our study focused on answering the following two research questions

RQ_1. What do gists look like? We surveyed the contents and metadata of Gists to get a picture of how they are used and whether users collaborate around them.

RQ_2. How are users using gists? We searched Web pages (including blogs) and Twitter to discover how users describe their use of Gists. We combined this information with the results of the previous research question to get a full picture of how Gists are being used.

Our methodology can be summarized as a mixed methods approach [6] that combines both quantitative and qualitative analysis on a collection of data. The study was composed of three main components, with the first two being used to answer RQ_1, and the third to answer RQ_2.

1) A quantitative analysis of Gist metadata and contents, based on the Gists collected from a large portion of GitHub users.
2) A qualitative analysis of a sample of Gists based on a small, random sample of the Gists collected. We inferred the purpose of a Gist by inspecting the contents of each file that composed the Gist.
3) A qualitative analysis of users' comments about Gists, based on search results from Websites and Twitter.

A. Data Set

In July 2014, we downloaded the metadata and contents of a large sample of Gists using the following method. We started by downloading the list of GitHub users from the GHTorrent project [8], which contained 2.9M. GHTorrent contains big part of the activity starting from 2012. We filtered these users to identify true users (GHTorrent contains both organizations and users, see [9] for details); This left us with 2.4M users. From this list of 2.4 million users, we randomly sampled 750k (31%). For each of these 750k users, we downloaded the metadata of all their gists (if they had any). Only 103k users had at least one gist. In total, these 103k users had 762k Gists. We also proceeded to download the first 618k Gists. This data is available in the replication package at http://turingmachine.org/2015/gists.

Table II summarizes the metadata associated with a Gist. As Gists use Git for their underlying storage mechanism, users can fork Gists, and modifications to Gists are recorded as commits. GitHub allows any user to make comments on a Gist.

[10] https://github.com/about/press

[11] http://www.stackoverflow.com/

[12] http://pastebin.com/

[13] https://snipt.net/

[14] http://en.wikipedia.org/wiki/GitHub#Gist

Description	Size
Total population of users	2,407,094
Sampled users	750,000
Users with Gists	103,092
Gists of sampled users	762,034
Gists downloaded	618,393

TABLE I: Sample used in this study.

Description	Size
Gist unique identifier	Unique to all Gists in the system
Description	As described by its owner
Files count	Number of files in Gist
Forks count	Forks of Gist made by other users
Commits count	Number of commits pushed to Gist
Commits history	Number of additions and deletions pushed in every commit
Comments count	Number of comments on Gist by other users
Language	List of languages of each file in Gist
Gist size	
Creation date	
Last modification date	

TABLE II: Description of the Gist metadata.

Gists can be composed of one or more files. Each of the files in a Gist contains metadata, which is described in Table III. The *MIME type* attribute documents the type of file using the MIME notation [7]. We also computed metrics on Gists that consisted of text files: number of lines using the `wc` UNIX command; lines of code per file (for source code files) using SLOCcount[15].

Description	Size
Filename	
MIME type	Type of file using MIME notation
Language	For source code, its programming language (based on its extension)
Size	In bytes
Lines*	Number of lines in Gist
SLOCS*	Number of lines for source code files

TABLE III: Description of the file metadata in Gists. Those marked with * were computed from the downloaded Gists.

B. *Manual Analysis of Gists*

To understand the content of Gists, we randomly sampled 398 Gists among the ones already downloaded for our quantitative analysis. To abstract recurring patterns, we extracted themes from each Gist's content following Creswell's guidelines [4] for coding. We started reading through the Gists to obtain a general idea of their content and then grouped the contents into categories. Because a Gist is composed of one or more files, we considered two strategies to segment them: analyze the content type of a Gist as a whole, and determine the relationships between the files in a Gist. In the first case, a Gist could be labelled with one or more terms.

Three researchers performed the manual analysis to cross-validate the coding process and minimize potential bias.

C. *Analysis of Users' Discussions Regarding Gists*

We also considered the information in Web pages and Twitter postings that described how Gists are used. We looked for the most relevant Web pages explaining the use of Gists—either official or unofficial—as well as the most recent comments at the time we were conducting the study.

1) Web Pages: To find relevant Web pages, we queried the Yahoo, Bing, Ask, and Duck Duck Go search engines. We used a script to scrape the first page of links provided for the following queries: *"What is a GitHub Gist"*; *"How do I use GitHub Gists"*; and *"What are GitHub Gists"*. Once we collected the links, we manually read each Web page and made note of the suggested uses.

2) Twitter Postings: To find relevant Twitter postings (colloquially known as *"tweets"*), we performed the query *"GitHub Gists"* using the Twitter Web Search interface. To minimize profile bias on Twitter, we performed the query anonymously, in a private Web browser session with no cookies. We narrowed the search to 6 months, from January 1st to July 31st, 2014, which resulted in a collection 492 tweets. We manually read each tweet and—when appropriate—followed the links pointed there, and made note of the usages suggested.

IV. RESULTS

A. *RQ_1: What do gists look like?*

When considering the results in this section, it is important to keep in mind that this study only pertains to public Gists (we are not able to mine private Gists). GitHub does not impose any restrictions on the number of private Gists a user can have (compared to private repositories which are available only to paying users). Hence, if we repeated this study on private Gists, the results could be different.

1) Users and Their Gists: Gists are used by a small proportion of users. As Table IV shows, 14% of users have at least 1 public Gist. Because the population of users with no public Gists is so large, our remaining analysis will concentrate on users with at least 1 public Gist (103k).

Users	Count	%
Having at least one public Gist	103,092	13.8%
Having no public Gists	645,368	86.2%

TABLE IV: Comparison of users with at least one Gist and those who do not have any.

For users with public Gists, the distribution of the number of public Gists per user is shown in Figure 1. The median number of public Gists is 3 (quartiles 1 & 10). Only 4% of users have 30 or more public Gists.

We hypothesized that the number of Gists a user has depends on their use of GitHub—users with more repositories or commits are more likely to use Gists. For this reason, we calculated the correlation between the number of Gists and other activities; We used the set of all users in the sample, not only those with Gists. The correlation between the number of commits a user has performed and the number of Gists they

[15] http://www.dwheeler.com/sloccount/

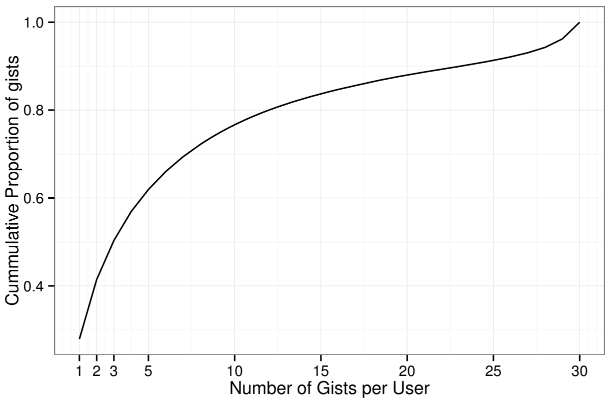

Fig. 1: Accumulated proportion of Gists per user.

have is 0.2, between the number of repos and the number of Gists is 0.31, and between the days since the user registered and the number of Gists they have is 0.37. In all these cases, the p-value was significantly smaller than 0.001. While the correlations are weak, they suggest that as people use GitHub more, they are more likely to use Gists.

2) What Are the Contents of Gists: We quantify the contents of Gists in several ways, starting with files per Gist. In theory, a Gist is a Git repository, and as such, a Gist can contain any number of files. Table V shows the accumulated distribution of files per Gist: 86.8% of Gists contain a single file, and 98.5% of them contain at most 4 files. Very few (0.02%) had more than 10 files.

Files	Gists	%
0	148	0.0%
1	661,565	86.8%
2	53,041	7.0%
3	24,108	3.2%
4	12,588	1.6%
>=5	10,584	1.4%

TABLE V: Breakdown of Gists by the number of files they contain.

GitHub documents the contents of Gist files in two ways: by MIME type, and by the associated programming language or data format (if applicable).

The breakdown by MIME type is depicted in Figure 2. As it can be seen, while the text/plain category dominates (many source code files fall into this category—see below), there is a wide variety of file types. Some of the most common types include images (PNGs are 9.3%, GIFs 0.8%, and JPEGs 0.3%), HTML files (4.7%), JSON data (1.5%), and CSS (2.1%). Some programming languages are identified as MIME types (php, sh, ruby, python, javascript), but many others are not (they are included in the text/plain category).

GitHub identifies the language of the file based on its extension. This includes source code files, data files (such as JSON and XML), and some text files (such as Markdown and HTML). In our sample, 71% of Gists contained at least 1 file classified in this manner, representing 260 different languages. Figure 3 shows the distribution of the top 30 languages. The top 5 programming languages are JavaScript (12.9%), Ruby (11.8%), PHP (10.4%), Python (7.1%), and Shell script (6.3%). The next 4 are markup languages: Markdown (5.9%), HTML (5.7%), XML (4.4%), JSON (2.6%). Note that the remaining 230 languages account for 7.5% (the "Other" category).

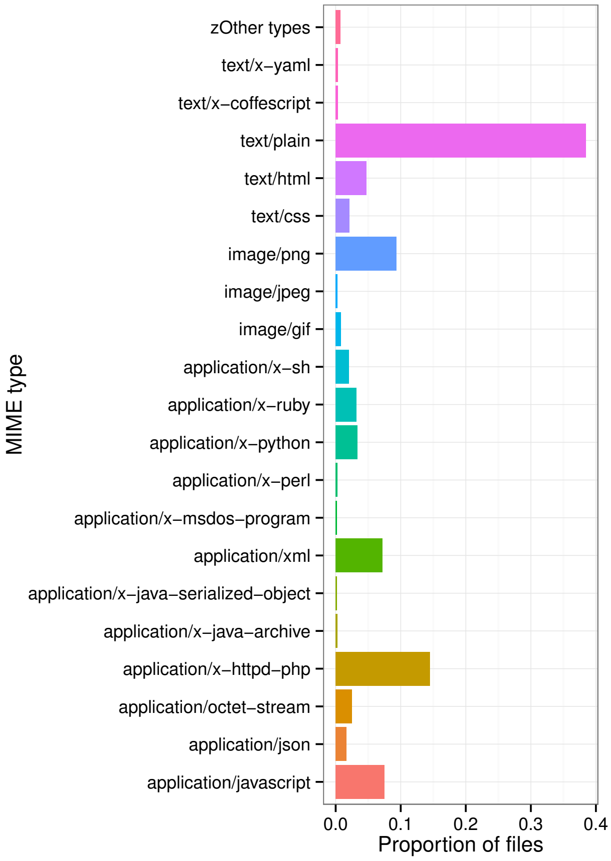

Fig. 2: Proportion of files by MIME type.

When analyzing Gists, we first quantify them by size. Figure 4 shows the distribution of the size of files in Gists using three metrics: bytes, lines, and SLOCs. In terms of bytes, the median size is 920 bytes (quartiles 374 & 2339 bytes). There exist some outliers: 0.06% of files are larger than 1 megabyte. For text files, we counted the number of lines per file: the median number is 22 lines (quartiles 9 & 54 lines). Using SLOCcount, we computed the number of SLOCs in each file. SLOCcount also identified 33 different programming languages (26.4% of files): the median was 18 SLOCs (quartiles 8 & 39).

3) Activity: Because Gists are stored using Git version control, we can trace their evolution (their commits) and the collaboration around them (the number of users who have forked them). In terms of commits, most Gists have very few:

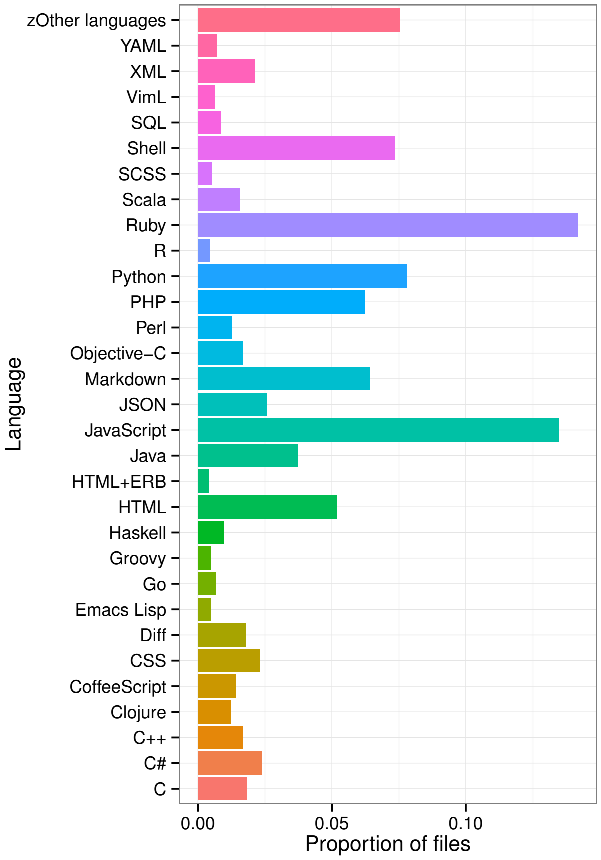

Fig. 3: Proportion of source code files by programing language.

62.9% had a single change, and 92.8% had 2 to 5 commits. The distribution of commits is shown in Figure 5. In terms of forks, only 5.1% of Gists have been forked once, and less than 0.8% have been forked 3 or more times (as shown in Table VI). There are, however, 23 Gists that have been forked more than 100 times. One feature that is different between regular repositories and Gists is that any user can add a comment to a Gist. As with commits and forks, very few Gists have comments (as seen in Table VII): only 6.9% received 1 or more comments.

Number of forks	Gists	%
0	723,833	94.9%
1	28,262	3.7%
2	4,914	0.6%
≥ 3	5,025	0.8%

TABLE VI: Forks per Gist.

4) Manual Analysis of the Contents of Gists: We manually inspected 398 randomly sampled Gists. We segmented the contents of Gists according to two strategies: we analyzed the content of the Gist as a whole, and then analyzed the relationships between the files within the Gist.

Comments per Gist	Gists	%
0	709,098	93.1%
1	33,603	4.4%
2	8,938	1.2%
≥ 3	3726	1.3%

TABLE VII: Comments per Gist.

a) Content Type: We observed that Gists are used for multiple purposes, not only for sharing code snippets. We categorized the Gists as source code (*Code*), any other form of text (*Note*), or both—for those cases when a Gist contains multiple files and at least one file each (*Code* or *Note*).

The breakdown of this first categorization is shown in Table VIII. There is a predominance of source code among Gists, although other types of text (*Note*) cannot be dismissed. The number of Gists that combine both types are low, which is expected given that most Gists contain a single file (see Table V). There are, however, 5 Gists that were classified differently by each researcher (*Not classified*).

Content type	Count	%
Code	290	72.9%
Note	92	23.1%
Both	11	2.8%
Not classified	5	1.3%

TABLE VIII: Breakdown of Gists by major categories of content type.

As seen in Table VIII, code is the most prominent use of Gists. However, we also found other uses, such as system configuration information, sharing public cryptographic keys, generic letters, or even a menu for—apparently—a restaurant. One Gist was written entirely in Japanese, for which we used an online translation tool to understand its content. Thus, GitHub Gists are not limited to western languages.

At the same time, we determined additional categories to better describe the purpose of a Gist given its content. The resulting categories are (in alphabetical order):

- Blog: Technical content in narrative format.
- Class: Definition of one class or module (source code).
- Command: A short command to be run in a shell (source code).
- Configuration: Configuration files used to build code, or for any other purpose.
- Data: Data stored in JSON, CSV, or other format.
- Diff: Differences between files or different versions of the same file.
- Documentation: Explanatory text about a piece of code or technology.
- Function: Definition of one or several functions (source code).
- Fragment: Partial piece of code (that is not a function nor a class), text, or command.
- Log: System log files, output of a program, and/or error messages.
- Non-technical: Notes without any technical content.

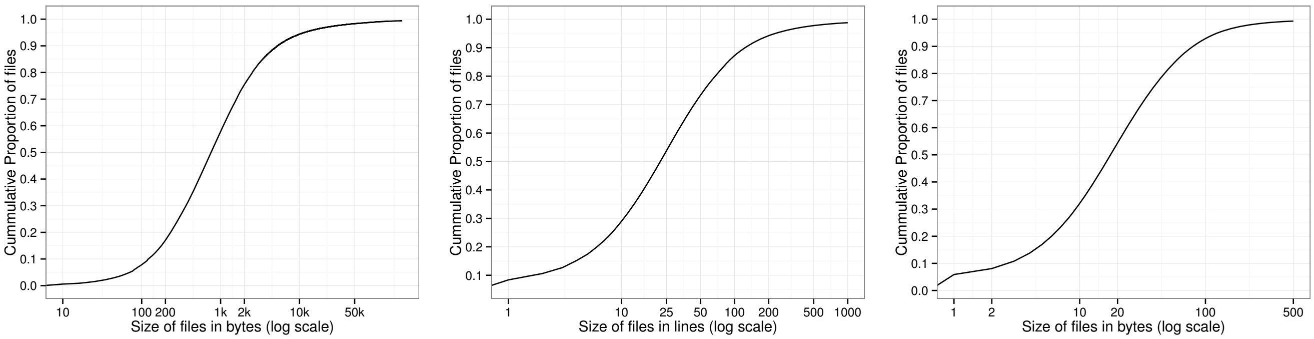

Fig. 4: Size of files in Gists in bytes, lines, and SLOCs

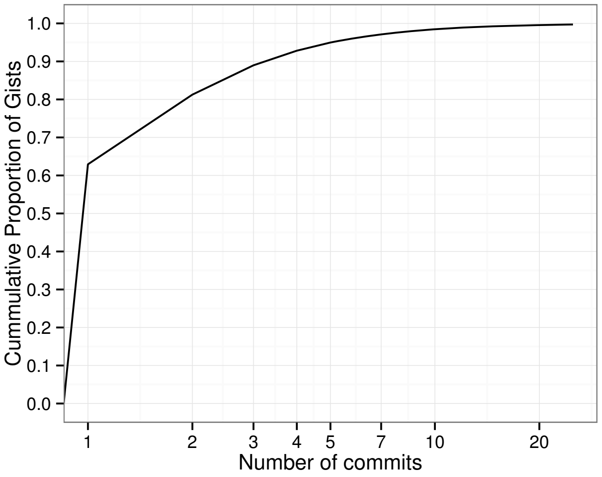

Fig. 5: Accumulated distribution of the number of commits per Gist.

- Template: Coding example or text with patterns.
- Test: Code or text used to test a program or system.

During the categorization process, we observed that the MIME type of a file might not be enough to understand a Gist. For example, the content of an HTML page could contain documentation, embedded JavaScript code, or be used as input for another program. We also observed that a Gist can be segmented in multiple categories, especially the Gists composed of multiple files.

Table IX contains the distribution of the more detailed categories based on the content of Gists. As a Gist can be classified in multiple categories, the sum of them do not represent 100% of Gists. Morever, we observed that 81 (20.35%) of the Gists were complete scripts, modules or programs. Hence, they are not represented in any of the categories.

b) When a Gist Has More than One File, They Are Related: A Gist is composed of one or more files. When a Gist has multiple files, we observed that they appear to be related. We used the following labels to categorize these relationships:

Content type	Count	%
Function	75	18.8%
Fragment	64	16.1%
Data	43	10.8%
Class	43	10.8%
Log	37	9.3%
Configuration	37	9.3%
Test	34	8.5%
Command	27	6.8%
Documentation	22	5.5%
Template	20	5.0%
Blog	14	3.5%
Non-technical	8	2.0%
Diff	7	1.8%

TABLE IX: Distribution of Gists by specific content type.

- Attachment: One file contains configuration or information of another file.
- Generation: One file is the input or output of another file.
- Reference: One file refers to another file, calls functions or methods defined in another file.
- Independent: Files in the Gist are independent or the same file is repeated.
- Single File: The Gist contains only one file.
- Test: One file is used to test the code of another file.

As shown in Table X, most sampled Gists contained only one file (this is consistent with the results of the quantitative analysis). For multi-file Gists, they are spread uniformly among the different categories. Aside from “Single file”, only “Independent” does not reflect an actual dependency between files.

Relationship between files of each Gist	Count	%
Single file	341	85.7%
Independent	19	4.8%
Reference	16	4.0%
Generation	11	2.8%
Test	5	1.3%
Not classified	4	1.0%
Attachment	2	0.5%

TABLE X: Distribution of Gists by relationship between filess

B. RQ_2 How are users using gists?

As described in Section III, we answered this question using two different sources of information: Websites and Twitter messages. Where appropriate, we provide quotations from the sampled Websites and tweets to illustrate our findings.

1) Suggested Uses on the Web: To learn more about how people are using Gists, we searched for “How do I use GitHub Gists”, “What are GitHub Gists for”, and “What is a GitHub Gist” using the Yahoo, Ask, Bing, and Duck Duck Go search engines. Then we made note of the uses discussed on the first page of results returned for each search.

We found that there were two prominent usage categories for Gists: official and unofficial. In the official usage category, the community uses Gists for code sharing, syntax highlighting and embedding in forums, and for simply storing snippets of code. GeoJSON map rendering is the only official usage that was unmentioned in the community.

Overall, we observed that Gists are suggested as a solution in cases where a full Git repository would be unnecessary.

a) Storing Code: Gists are intended as places to store snippets of code or other small pieces of information.

> *“Instead of creating a complete repository for only 1 or 2 files all the time, just add them to a Gist.”*[16]

b) Sharing Code: GitHub used to allow private gists, which were SSL encrypted to protect the contents; however it appears that this is no longer an available feature. “Private” Gists have been replaced with “secret” Gists. Secret Gists have an obfuscated URL but are otherwise visible to the public—simply providing the URL of the Gist to team members gives them access to a wiki-like code sample or snippet. By having all gists visible by the public, it encourages sharing of code. For the purpose of this paper, “private” and “secret” gists are interchangeable.

> *“Gists are a great way to share your work. You can share single files, parts of files, or full applications . . . Every Gist is a Git repository, which means that it can be forked, cloned, and manipulated in every way.”*[17]

c) Embedding Code: Gists allow users to embed the contents into blogs, forums, or any text field that supports JavaScript. This allows people to focus on their message rather than on the process of formatting the code in HTML.

> *“You can embed any Gist in your Web pages with a line of JavaScript code.”*[18]

d) To-Do Lists: Gists can serve as to-do lists, using the markdown rendering. The version control system records the times when tasks are completed, adding functionality at no cost.

> *“[To-do lists] help me stay organized, prioritize my day, and add structure to an otherwise chaotic schedule. I recently discovered what appears to be the best yet simplest way to keep a to-do list: a GitHub Gist.”*[19]

e) Web Hosting: The community has developed third-party applications that can render any HTML code stored in a Gist, making Gists an effective single-page Website.

> *“This [Website] is a simple viewer for code examples hosted on GitHub Gist. Code up an example using Gist, and then point people here to view the example and the source code, live!”*[20]
>
> *“You can write your HTML, CSS and JavaScript code in plain text, save the Gist as index.html and then use bl.ocks.org to serve the rendered version of that HTML Web page as it should appear in the browser.”*[21]

f) Editing Text: The markdown rendering in Gists makes them a simple Web text editor.

2) Uses of Gists Reported on Twitter: We analyzed 6 months of Twitter messages that contained information on GitHub and Gists. We filtered the postings that described or linked usages of Gists, then we categorize them. The resulting categories are described below.

a) Sharing Content: GitHub Gists is used to share snippets of code or any form of text. Although it is one of the purposes featured by GitHub regarding to Gists, we observed people advertising Gists frequently in Twitter.

> *“Gists - https://gist.github.com/ Gist is a simple way to share snippets and pastes with others. All gists are Git repositories.”*[22]

We also observed that a collection of related Gists can become a repository over time. For example, we observed 8 tweets[23] linking to a repository containing an introduction to Monad Transformers[24]. The repository was originally a collection of more than 30 Gists with snippets that evolved into a repository[25].

b) Saving Learning Outcomes: Gists can be used to aggregate disperse information technical or record technical tips and learning outcomes tips. For example, a tweet[26] linked to a Gist[27] containing a compilation of business models used by different Internet companies. The Gist is a compilation list of information gathered from a discussion in Hacker News—a social news web site. Although the tweet looks like a call for moderation on how people used Gists, the linked tweet had attracted a considerable attention. At the time of this study, it had received 24 comments, 35 revisions, 201 forks,

[16] https://www.adayinthelifeof.nl/2010/12/26/github-gists-revisioned-code-snippets-for-free/

[17] https://help.github.com/articles/about-gists/

[18] http://www.labnol.org/internet/github-gist-tutorial/28499/

[19] http://lifehacker.com/why-a-github-gist-is-my-favorite-to-do-list-1493063613

[20] http://bl.ocks.org/

[21] http://www.labnol.org/internet/github-gist-tutorial/28499/

[22] https://twitter.com/sstranger/status/493436023729586176’’

[23] E.g. https://twitter.com/philadev/status/490531557816692738

[24] https://github.com/kqr/gists/blob/master/articles/gentle-introduction-monad-transformers.md

[25] https://github.com/kqr/gists

[26] https://twitter.com/pessimism/status/492255854163689472

[27] https://gist.github.com/ndarville/4295324

and 1,633 stars. In comparison to the other results—seen in sections IV-A2, IV-A3, and IV-A4)—this Gists is an outlier because it is not related to code (23.1%), it has more than 3 comments (1.3%) and more than 3 forks (0.8%).

c) Collaboration: Gists also act as a tool to help people with collaboration. People can put their work in a private Gist and every member can commit to it, as is inferred from the following tweet:

> *"Every once in a while I think I wish I could "draft" a Pull-request, issue, or comment on GitHub, then I remember that private Gists exist."*[28]

d) Embedding Content in Blogging Platforms: One of the largest blogging Websites, WordPress, supports the embedding of Gists[29]. Many other blogging platforms, such as Medium, also support Gist embedding.[30]

> *"Thinking of migrating all of the code for my blog posts into @GitHub Gists like [url] - Would that be valuable to you?"*[31]

e) Version-controlled Lists: Some users have come up with non-trival ways to make full use of Gists. One good example is a popular blog being widely tweeted on Twitter. Authored by Carl Sednaoui[32], it teaches people how to maintain a to-do list using a private GitHub Gist.

> *"GitHub Gists are a great way to keep version-controlled lists (in this case, US states I've visited). . . "*[33]

V. Discussion of Results

With Gists, GitHub provides a simple way to create and version small files. Among the users we sampled, only 1 in 8 had public Gists, and most of these users had very few in total (the median number is 3). Our results show that someone's use of Gists is correlated with the length of time they have used GitHub. As such, we expect that the use of Gists increases significantly over time. It is also likely that there is a critical mass effect: As more people use and talk about Gists, others will follow suit.

As GitHub expected, most Gists are very small: 86.8% have only 1 file and the median size is 920 bytes (22 lines for text Gists). However, we found that their contents vary widely. While a large proportion contain source code, people are also using Gists for binary files (such as images) and data files (such as XML and JSON). Based on our qualitative analysis and manual sampling of Gists, the following themes have surfaced.

A. Gists Are Mostly Used to Store Source Code but Other Formats Are Frequently Used Too

As expected, Gists are mostly used to store snippets of source code. Our manual analysis showed that they cover a wide range of uses: shell scripts, class templates, complete functions, fragments of functions, etc. Because most do not evolve, we hypothesize that these snippets are being archived for future reference. In addition to source code, Gists also contain other information formats. The Markdown markup language is the fourth most common language, suggesting that storing snippets of text is also an important use of Gists. Similarly, we found that almost 10% of Gists contain images. Gists are also used to store logs, diffs, JSON data, and test data.

B. Gists Are Used to Create Reusable Web Components

GitHub provides a mechanism to dynamically embed Gists into Web pages. When a Gist is embedded, it is nicely typeset (according to the syntax of its language) into HTML. This includes Markdown and Org, both markup languages for text designed to be converted into HTML. When a Gist is rendered, it is bound with a box and text that identifies it as a Gist hosted in GitHub. However, this box can be removed. For example, gist-embed[34] enhances GitHub's rendering of Gists by removing any signs that the content comes from GitHub. Gists can act as dynamic "includes" in Web pages (whether formatted text or source code). Other formats that do not require typesetting (such as images and CSS) are easier to reuse since they do not need to be embedded. For example, if one wants to host an image in GitHub, all that is needed is to create a Gist and then refer to this Gist using the GitHub URL that retrieves the original content (the "raw" Gist). This is likely one of the reasons that we found that almost 10% of Gists are images, and might explain why 8% are JavaScript snippets. Dynamically using Gists in Web pages (either embedded or in raw format) has three advantages:

1) For languages that must be converted into HTML (such as Markdown or source code), it allows authors to ignore the complexities of authoring HTML, and concentrate on creating content.
2) It allows the reuse of components for the Web. The same Gist can be reused multiple times.
3) It isolates the evolution of the component from the use of the component. Because Gists are used dynamically (in raw format, or embedded and rendered) by Web pages, they can be updated without having to change the Web page that uses them.

The evidence we have collected suggests that this is an important use of Gists, especially for Web pages that include source code. Converting source code to a nicely typeset HTML would require the use of extra tools. By hosting the code snipped as a Gist in GitHub, the job of rendering the code into HTML is no longer the responsibility of the user. As a result, it is not only easier to embed source code into a Web

[28] https://twitter.com/nuclearsandwich/status/249213040610910209

[29] http://crunchify.com/how-to-embed-and-share-github-gists-on-your-wordpress-blog

[30] https://medium.com/the-story/yes-we-get-the-gist-1c2a27cdfc22

[31] https://twitter.com/jessealtman/status/456390107952467968

[32] http://carlsednaoui.com/post/70299468325/the-best-to-do-list-a-private-gist

[33] https://twitter.com/dliggat/status/458090816930848768

[34] https://github.com/blairvanderhoof/gist-embed/blob/master/gist-embed.js

page, but the source code looks good. It is hard to evaluate the impact of this feature, but it is likely this rendering has a direct impact on the readability of Web pages that present source code—compared to simply using a pre-formatted HTML tag (<pre>), for example.

C. Gists Are Used as an Enhanced Online Document Editing System That Adds Version Control Features

With Gists, GitHub provides a lightweight method to edit documents with the full benefits of version control. GitHub isolates the user from the creation and editing of Gists from Git. A user only needs a Web browser to gain all the benefits that Git provides for tracking changes (when a change is made, who made the change, and what the change was). We have found evidence that Gists are used for this purpose, even though it is not common (less than 8% of Gists have more than 6 changes).

D. Users Do Not Collaborate Around Gists

Even though we studied public Gists, we found that they are mostly personal artifacts. Gists are rarely forked, and the majority of Gists never change. Unfortunately, GitHub does not provide statistics regarding how many users visit a Gist, nor when. Therefore, we cannot create a picture of how useful Gists are to those who do not fork them or modify them. It appears as if some Gists serve as an external memory to their owner, a memory that the owner is happy to share with everybody (when the Gist is public).

E. Gists Are a Public Scrapbook

As described above, it is very likely that many public Gists are reusable components, intended to be used by their owner in the creation of Websites. However, it is also likely that many Gists are artifacts that the user would like to share with anybody who finds them useful. In a way Gists are a public online scrapbook where developers can collect small artifacts that they find useful, and that can also be potentially useful to others.

VI. Future Work

This study is exploratory. We are just scratching the surface on what Gists are and how they are used. One aspect that we have not researched is what users think about Gists, and future work should survey and interview users. For example: What motivates a person to create a Gist? What do they use Gists for? What factors determine if a Gist should be public or private? Do they expect their Gists to be used by others? Our study was limited to public Gists. Are private Gists different from public Gists?

Another area that requires future research is exploring how users find and reuse Gists (which they have not authored). GitHub provides a search engine for Gists, hence, it expects users to benefit from the Gists of other users. It could be also interesting to study further why the scripting languages used in Gists outnumber other languages such as Java or C; is sharing snippets easier for scripting languages? is this an indicative of activity on GitHub projects developed in such languages?

We hypothesize that some Gists are created for future use. In this case, the user has considered that the snippet is important enough to be remembered as a Gist. It could be interesting to see if there are some commonalities among the Gists of different users. Are different users storing similar Gists?

When a Gist is meant to be reused in the future, such Gist is a potential reusable component. Gists might provide an interesting view on reuse at a higher level of granularity than libraries. It could be interesting to perform clone detection between Gists and source code in the owners' repositories (and other repositories) to find out how reused a Gist is. This could mean that there are certain functionalities that the user frequently requires.

GitHub is not the only Website that stores snippets of code. We need research that explores other repositories and compares them to the results described herein.

VII. Threats to Validity

We triangulated the results of the quantitative and qualitative analysis to overcome potential threats to validity. The quantitative analysis was performed by two researchers, while the qualitative analysis was performed independently by three researchers on the same data set, which will reduce the likelihood of erroneous results. In this section, we explain how we addressed each threat to validity. We are providing an online package that includes the data we analysed and the results of the manual analysis at http://turingmachine.org/2015/gists.

A. Construct Validity

The manual categorization of Gists may introduce errors into the results. The categorization activity involved having a researcher categorize the Gist by interpreting the contents of the files contained in the Gist. If a researcher misinterpreted the contents, it would introduce errors. To minimize the errors introduced by misinterpretation, three researchers categorized the 398 Gists in the sample. Each researcher followed Creswell's guidelines [4] for coding to minimize the introduction of subjective bias by the researcher.

An unsuitable sample of the Twitter posts (tweets) and Web search engine results can also affect the validity of the results negatively. Twitter returns posts with priority given to more recently posted results—the results of the query depend on the time it was executed. This dependence makes the results transient and likely to change. While search engines are still susceptible to changes over time, they use a score-based search algorithm to return the most relevant pages, making the results less transient. The internal details of each search engine are unknown to us; it is unclear wether the results returned by the search engines are representatives samples of the queries performed. We hope that the qualitative analysis of both data sets, in conjunction with the qualitative and quantitative analysis of Gists, reduces distortion in the results.

B. Internal Validity

GitHub provides secret Gists, which are hidden from search engines and the public forum, but are available to anyone with the Gist identifier. We performed the analysis on data collected from public Gists so our results are limited. The contents of secret Gists may differ from the contents of public Gists.

C. External Validity

This study is exploratory and only applies to Gists in GitHub. While there are other snippet storage sites on the Internet (such as *pastebin* and *snip*), we do not make any claims regarding the generalizability of our results to those other sites.

VIII. Conclusions

In this paper, we conducted an exploratory study of GitHub Gists, quantitatively measuring 762k Gists that belong to 750k users, manually coding the content of hundreds of Gists, and exploring the common Gist usages described in Web pages and on Twitter. Our qualitative analysis allowed us to identify recurring patterns in the data that might be difficult to detect quantitatively.

Our goal was to understand the purpose of Gists and how they are used. We summarize our results below.

RQ_1. What do Gists look like? Usually a Gist is a small snippet of source code. Although we found Gists that did not contain source code, those were less frequent (23.1% in contrast to 72.9% of code). In most cases, Gists are composed of one file whose size is relatively small. We also found that Gists are written in many different programming languages, with JavaScript and Ruby being among the most popular.

RQ_2. How are users using Gists? The usage of Gists goes beyond the official purpose promoted by GitHub. GitHub describes Gists as a way to share source code and to embed source code into external services such blogs or forums. However, we found that Gists are also used to maintain online notes with the full benefits of version control. We also found an incipient set of tools that help users manage their Gists; We expect the number of such tools to grow as Gists become more popular.

References

[1] T.F. Bissyande, F. Thung, D. Lo, Lingxiao Jiang, and L. Reveillere. Popularity, interoperability, and impact of programming languages in 100,000 open source projects. In *Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual*, pages 303–312, July 2013.

[2] A. Capiluppi, A. Serebrenik, and L. Singer. Assessing technical candidates on the social web. *Software, IEEE*, 30(1):45–51, Jan 2013.

[3] Joohee Choi, Junghong Choi, Jae Yun Moon, Jungpil Hahn, and Jinwoo Kim. Herding in open source software development: An exploratory study. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work Companion*, CSCW '13, pages 129–134, New York, NY, USA, 2013. ACM.

[4] John W Creswell. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, volume 2. Sage Publications, 2009.

[5] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb. Leveraging transparency. *Software, IEEE*, 30(1):37–43, Jan 2013.

[6] Steve Easterbrook, Janice Singer, Margaret-Anne Storey, and Daniela Damian. Selecting Empirical Methods for Software Engineering Research. In *Guide to Advanced Empirical Software Engineering*, pages 285–311. Springer London, 2008.

[7] N. Freed, J. Klensin, and T. Hansen. Request for Comments: 6838 Media Type Specifications and Registration Procedures. Internet Engineering Task Force (IETF) http://tools.ietf.org/html/rfc6838, 2015.

[8] Georgios Gousios. The GHTorrent dataset and tool suite. In *MSR '13: Proceedings of the 10th Working Conference on Mining Software Repositories*, may 2013. Best data showcase paper award.

[9] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. The promises and perils of mining GitHub. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, pages 92–101, New York, NY, USA, 2014. ACM.

[10] Anirban Majumder, Samik Datta, and K.V.M. Naidu. Capacitated team formation problem on social networks. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1005–1013, New York, NY, USA, 2012. ACM.

[11] Jennifer Marlow and Laura Dabbish. Activity traces and signals in software developer recruitment and hiring. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 145–156, New York, NY, USA, 2013. ACM.

[12] Jennifer Marlow, Laura Dabbish, and Jim Herbsleb. Impression formation in online peer production: Activity traces and personal profiles in GitHub. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 117–128, New York, NY, USA, 2013. ACM.

[13] Leif Singer, Fernando Figueira Filho, Brendan Cleary, Christoph Treude, Margaret-Anne Storey, and Kurt Schneider. Mutual assessment in the social programmer ecosystem: An empirical investigation of developer profile aggregators. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 103–116, New York, NY, USA, 2013. ACM.

[14] D. Spinellis. Git. *Software, IEEE*, 29(3):100–101, May 2012.

[15] Bogdan Vasilescu, Vladimir Filkov, and Alexander Serebrenik. Stackoverflow and GitHub: Associations between software development and crowdsourced knowledge. In *Social Computing (SocialCom), 2013 International Conference on*, pages 188–195, 2013.

[16] Rahul Venkataramani, Atul Gupta, Allahbaksh Asadullah, Basavaraju Muddu, and Vasudev Bhat. Discovery of technical expertise from open source code repositories. In *Proceedings of the 22Nd International Conference on World Wide Web Companion*, WWW '13 Companion, pages 97–98, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.

Bibliography

[1] Scott Chacon and Junio C Hamano. *Pro git*, volume 288. Springer, 2009.

[2] B. de Alwis and J. Sillito. Why are software projects moving from centralized to decentralized version control systems? In *Cooperative and Human Aspects on Software Engineering, 2009. CHASE '09. ICSE Workshop on*, pages 36–39, May 2009.

[3] P.C. Rigby, E.T. Barr, C. Bird, P. Devanbu, and D.M. German. What effect does distributed version control have on oss project organization? In *Release Engineering (RELENG), 2013 1st International Workshop on*, pages 29–32, May 2013.

[4] D. Spinellis. Git. *Software, IEEE*, 29(3):100–101, May 2012.

[5] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb. Leveraging transparency. *Software, IEEE*, 30(1):37–43, Jan 2013.

[6] A. Capiluppi, A. Serebrenik, and L. Singer. **Assessing Technical Candidates on the Social Web**. *Software, IEEE*, 30(1):45–51, Jan 2013.

[7] Jennifer Marlow and Laura Dabbish. Activity traces and signals in software developer recruitment and hiring. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 145–156, New York, NY, USA, 2013. ACM.

[8] Rahul Venkataramani, Atul Gupta, Allahbaksh Asadullah, Basavaraju Muddu, and Vasudev Bhat. Discovery of technical expertise from open source code repositories. In *Proceedings of the 22Nd International Conference on World Wide Web Companion*, WWW '13 Companion, pages 97–98, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.

[9] C. Teyton, J.-R. Falleri, F. Morandat, and X. Blanc. Find your library experts. In *Reverse Engineering (WCRE), 2013 20th Working Conference on*, pages 202–211, Oct 2013.

[10] T.F. Bissyande, F. Thung, D. Lo, Lingxiao Jiang, and L. Reveillere. Popularity, interoperability, and impact of programming languages in 100,000 open source projects. In *Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual*, pages 303–312, July 2013.

[11] Anirban Majumder, Samik Datta, and K.V.M. Naidu. Capacitated team formation problem on social networks. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1005–1013, New York, NY, USA, 2012. ACM.

[12] Jennifer Marlow, Laura Dabbish, and Jim Herbsleb. Impression formation in online peer production: Activity traces and personal profiles in github. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 117–128, New York, NY, USA, 2013. ACM.

[13] Leif Singer, Fernando Figueira Filho, Brendan Cleary, Christoph Treude, Margaret-Anne Storey, and Kurt Schneider. Mutual assessment in the social programmer ecosystem: An empirical investigation of developer profile aggregators. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, pages 103–116, New York, NY, USA, 2013. ACM.

[14] Joohee Choi, Junghong Choi, Jae Yun Moon, Jungpil Hahn, and Jinwoo Kim. Herding in open source software development: An exploratory study. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work Companion*, CSCW '13, pages 129–134, New York, NY, USA, 2013. ACM.

[15] Bogdan Vasilescu, Vladimir Filkov, and Alexander Serebrenik. Stackoverflow and github: Associations between software development and crowdsourced knowledge. In *Social Computing (SocialCom), 2013 International Conference on*, pages 188–195, 2013.

[16] G. Gousios and D. Spinellis. Ghtorrent: Github's data from a firehose. In *Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on*, pages 12–21, 2012.

[17] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. The promises and perils of mining github. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, pages 92–101, New York, NY, USA, 2014. ACM.

[18] Weiliang Wang, G. Poo-Caamano, E. Wilde, and D.M. German. What is the gist? understanding the use of public gists on github. In *Mining Software Repositories (MSR), 2015 IEEE/ACM 12th Working Conference on*, pages 314–323, May 2015.