future internet

MDPI

Article

Understanding How GitHub Supports Curation Repositories

Yu Wu [1,*], Jessica Kropczynski [2], Raquel Prates [3] and John M. Carroll [1]

[1] College of Information Sciences and Technology, The Pennsylvania State University, University Park, PA 16802, USA; jcarroll@ist.psu.edu
[2] School of Information Technology, University of Cincinnati, Cincinnati, OH 45221-0002, USA; jess.kropczynski@uc.edu
[3] Department of Computer Science, Federal University of Minas Gerais, Belo Horizonte, MG 31270-901, Brazil; rprates@dcc.ufmg.br
* Correspondence: yuw132@psu.edu; Tel.: +1-814-321-8276

Received: 4 February 2018; Accepted: 8 March 2018; Published: 10 March 2018

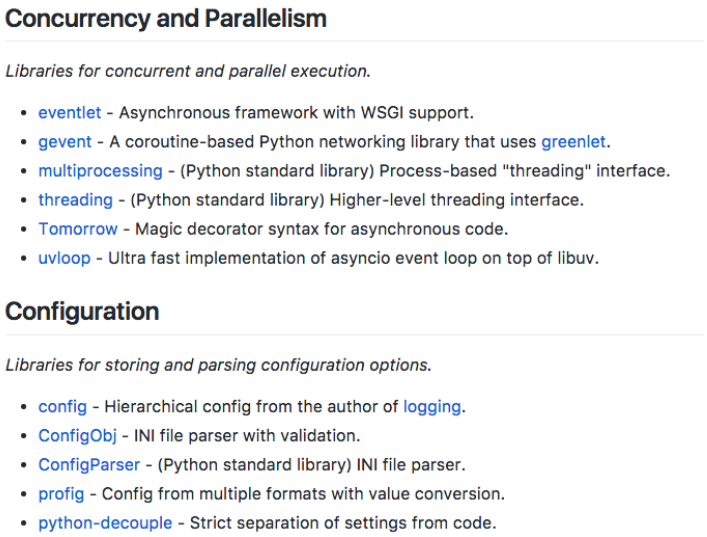
Abstract: In recent years, software developers have started to appropriate GitHub repositories to curate resources, in order to systematically select, evaluate, and organize existing artifacts for preservation and future use. Curation behaviors in social media sites, such as users' experiences to curate tweets from Twitter and pins on Pinterest, are well documented. However, GitHub, as a social coding platform, presents a new context for this activity, raising questions about the nature of curation on this task-driven online work site. To explore and understand curation on GitHub, we compared and contrasted curation repositories with software repositories using activity logs and analyzed the content of popular curation repositories. Our results show that: (1) curation repositories have become a favorite category of repositories in GitHub; (2) curation repositories leverage collaborative features and practices native to GitHub in new ways; (3) curation repositories collect and preserve high-quality resources for the software developers' community. Our results suggest that curation is becoming increasingly important to the software developers' community, and current practices can be better supported with tools designed specifically for curation.

Keywords: curation; GitHub; software developers' community

1. Introduction

GitHub is an online service for source code hosting and software project collaboration. It provides features for coordinating work, such as the issue tracker for a repository to discuss software features and bugs and the pull-request mechanism for software developers to make contributions to other repositories [1,2]. It also includes social features, such as following other GitHub users to make connections and receiving updates of others from activity traces [1,2].

Recently, software developers began to create GitHub repositories that systematically organize and index the Internet resources (Figure 1), and many gained notable popularity. Studies of this phenomenon have investigated the motivations of curators and user experiences with curation repositories [3,4]. The analysis of GitHub features for supporting curation, as well as the role that curation repositories have been playing in the software developers' community, are still missing. Therefore, in this paper, we are interested in adding to the literature by understanding the way that GitHub features are applied in curation repositories as well as the functions of curation repositories in the software developers' community.



Concurrency and Parallelism

Libraries for concurrent and parallel execution.

- eventlet - Asynchronous framework with WSGI support.
- gevent - A coroutine-based Python networking library that uses greenlet.
- multiprocessing - (Python standard library) Process-based "threading" interface.
- threading - (Python standard library) Higher-level threading interface.
- Tomorrow - Magic decorator syntax for asynchronous code.
- uvloop - Ultra fast implementation of asyncio event loop on top of libuv.

Configuration

Libraries for storing and parsing configuration options.

- config - Hierarchical config from the author of logging.
- ConfigObj - INI file parser with validation.
- ConfigParser - (Python standard library) INI file parser.
- profig - Config from multiple formats with value conversion.
- python-decouple - Strict separation of settings from code.

Figure 1. The "awesome Python" curation repository (https://github.com/vinta/awesome-python).

In recent years, curation behavior has been investigated in the context of social media sites, such as Twitter and Pinterest, and with respect to media contents, such as videos, images, text (tweets), and hyperlinks to other online resources [5,6]. Curation in GitHub is distinct from curation in those websites of in the following ways. First, hyperlinks posted in GitHub curation repositories are directed at the software developers' community, whose members share and cultivate a professional interest in software development. Second, the curation on GitHub is embedded in the context of GitHub, which is an ensemble of social coding features intended for software development and collaboration rather than for curation. The GitHub context raises interesting questions concerning how its features support such an appropriation for curation and what role it serves for this specific community. Therefore, this paper attempts to answer the following overarching research question:

RQ: *How does GitHub support curation practices?*

To answer this research question, we first compare how curation repositories are different from typical software repositories. As a relatively new way to utilize GitHub, users are likely to participate in such kind of repository differently. This relates not only to the number of GitHub users who star a repository, which usually shows a user's interests towards a repository [7,8], but also to the different types of activities that take place inside a curation repository, such as pull requests [8]. Thus, we explore our initial research question through the specific sub-research questions outlined below.

RQ1*: How are curation repositories different from the typical software repositories of GitHub?*

Prior literature has documented software practices on GitHub well [1,2,8], and some literature has provided insights into the curator's motivations and the users' experiences with curation repositories [3,4]. However, a gap still exists in how GitHub features are utilized in curation repositories and how they are adopted differently, as compared to the intended software practice. This research question intends to close this gap and to provide an account of the categories and user participation that make curation repositories different.

In addition to the comparison with software repositories, currently, we have little understanding of the details of curation repositories in terms of what kind of needs they address and what role they play in the software developers' community. Thus, the following research question will be investigated next.

RQ2*: What is the emerging role of curation repositories in the GitHub community?*

Specifically, this research question examines the function of curation by examining the contents, format, the owner's characteristics, and collaboration pattern of curation repositories on GitHub. The answer to this research question can elucidate why curation repositories are useful, why they have suddenly drawn great attention, and what kind of impact they bring to the software developers' community.

Through a statistical analysis of the activity logs to compare curation repositories with software repositories and a content analysis of the most popular curation repositories on GitHub, we find that curation repositories are more popular than software repositories, and GitHub users participate in curation repositories in a qualitatively different way. Most curation repositories are maintained by individual software developers and they intend to collect and preserve high-quality resources, originated from either inside or outside of GitHub, about the technology industry. Our findings suggest that curation repositories become an essential way for the software developers' community to centralize fragmented information and share knowledge. This study contributes to the understanding of curation in GitHub and sheds light on the potential ways to better support the practice.

2. Related Work

2.1. The GitHub Environment

GitHub is known as a social coding service for software developers to host source code artifacts and to collaborate with others using its social features [1,2]. In recent years, GitHub has become a popular site for researchers to study online software practices. Dabbish et al. (2012) found that GitHub features are signals for software developers to make social inferences [1]. For example, the recent activity and frequency of maintenance of a repository are signs of the project's liveliness, the number of followers a developer has is interpreted as the community status, and the number of stars and forks indicate the quality of a software project [1]. Marlow et al. (2013) found that software developers systematically use available cues in GitHub to form an impression about peers in collaboration, e.g., to find out more information, such as interaction style and expertise, about the contributor when responding to code commits [2]. Specifically, Tsay et al. (2014) focused on analyzing the GitHub features adopted in evaluating pull requests and found that some features, such as the status of the developers in the community, are likely to increase the acceptance rate [8].

These studies distinguish curation on GitHub from that on other social media sites in the following aspects. First, the user base of GitHub is converged, consisting of a dominating population of software developers focusing on software practices. Conversely, social media sites, such as Twitter and Pinterest, have a very general user base comprising users that curate for "Entertainment & Hobbies" [5], or "Food & Drink", "DIY & Craft", "Home Decor", and "Women's Fashion" [9]. Second, GitHub concentrates on software practice, where features for collaboration are emphasized [1,2,8], while curation in other social media sites focuses on individuals and does not emphasize collaboration among its users [6,9]. The comparison leaves an interesting question concerning the characteristics of GitHub as an entirely different environment for curation.

Activities on GitHub are organized around repositories. Specifically, a GitHub repository is the basic unit to organize a software project and related activities (https://guides.github.com/activities/hello-world/). GitHub users can send pull requests to a repository for making code contributions or comment in the issue tracker for bug-reporting or feature-requesting [1]. In addition, in 2012, GitHub announced a new feature that allows users to mark a repository as interesting by giving it a "star" (https://github.com/blog/1204-notifications-stars). Today, stars are considered an important indicator of the popularity of a repository [7]. Lima et al. (2014) found that the number of stars a repository receives follows a power-law distribution, which means that a large amount of community attention is focusing on a small number of repositories [10]. A curation repository is a GitHub repository that is created for the purpose of curation, which is an appropriation of GitHub features designed for software collaboration. We are interested in understanding (1) the way that GitHub features are adopted, and (2) how activities are organized around a curation repository.

2.2. Curation in GitHub

Curation is defined as the process that involves identifying, selecting, validating, organizing, describing, maintaining, and preserving existing artifacts [11]. Duh et al. (2012) described a curator as

the one who collects and organizes existing content into a larger unit, emphasizing the manual effort of curation as compared to automatic methods such as algorithmic search and aggregation [5].

The purposes of curation have been a popular topic in curation-related literature. Duh et al. found that entertainment and hobbies together with serious topics, such as society, politics, and economics, are among the most favored topics when curating tweets. Change et al. (2014) discovered that the most popular categories of pins are "food and drink", "DIY crafts", and "home decoration" [9]. We are interested in identifying the objectives of curation repositories in GitHub, which offers a different context for this behavior than Twitter or Pinterest.

Data provenance and leadership in curation emerge as interesting topics in the recent curation literature. Matthews et al. (2014) captured the source of each curated item in an organizational environment and found that both internal and external resources are curated with respect to different tools (wiki, blog, etc.) [12]. The curation efforts in intranet communities are often performed by community leaders, while members of the communities are encouraged to curate external resources [12]. We would also like to follow a similar procedure to analyze the sources of the curated items and the status of the owners of curation repositories on GitHub.

A recent study by Munaiah et al. (2017) developed a tool, called reaper, that evaluates a GitHub repository from eight dimensions to determine whether a repository is an engineered software project and to identify software projects that conform to the dimensions within a sample of 1,994,977 GitHub repositories [13]. The research focus of Munaiah et al. (2017) was different from the one of this paper, since they developed a set of metrics and curated software repositories from GitHub according to these metrics, which is a method of algorithmic curating [13]. In contrast, in this study, we investigate the curation behavior happening on GitHub that appropriates GitHub repositories for curation purposes, which is a different category of curation in the software practice.

Last but not least, since GitHub is a highly collaborative environment, we are also wondering whether and how collaborative features, such as pull requests, are used in curation repositories.

3. Method

To characterize popular curation repositories hosted on GitHub, we collected a dataset of activity logs on GitHub, identified the top curation and software repositories, compared them with top software repositories, and coded the contents of the curation repositories. The dataset was collected from GitHub Archive (https://www.githubarchive.org/), which captures a comprehensive GitHub timeline data. GitHub Archive data has been actively used for analysis in academic publications [10,14–16]. However, the dataset was influenced by a bug report about a crawler issue on 22 September 2013 (https://github.com/igrigorik/githubarchive.org/pull/37), which resulted in a loss of events. For consistency and data quality, we collected 109,782,635 events on 7,079,847 repositories that occurred between 1 October 2013 and 31 August 2014.

In order to find the commonalities of the curation repositories that are of interest to others, we selected the ones based on indicators of popularity. Trending repositories are displayed on GitHub by day, week, or month. The trending repositories typically average about 500 stars per repository (https://github.com/trending?l=all&since=weekly). Given that a repository that trends can be considered a relatively popular repository on GitHub, we then selected all repositories that had more than 500 stars within a date range. At the same time, many software repositories with more than 500 stars have been established for years. In order to have a fair comparison of curation and software repositories, only repositories that were created after January 1, 2013 were retained, resulting in 1929 repositories.

To identify curation repositories within this sample, we first identified 1384 software projects, whose programming languages were automatically detected by GitHub. For the remaining 545 repositories, we manually labeled each. The criterion used to determine if a repository was a curation repository was whether the primary content of the repository was a collection of Internet resources. As a result, we identified 49 curation repositories from the 545 repositories. As we could not verify the nature of the other 496 repositories, and since our immediate interest for this paper focused

on popular curation repositories and their comparison with software repositories, we discarded these 496 repositories from the sample.

After identifying the most popular curation and software repositories, to answer RQ1, i.e., *how are curation repositories different from the typical software repositories of GitHub*, we aggregated the activity log data into 49 curation repositories and 1384 software repositories, respectively, and applied a quantitative method to compare them. Specifically, for each type of activity, we analyzed whether the number of the activities for curation projects was different from the number of activities for software repositories. To answer RQ2, i.e., *what is the emerging role of curation repositories in the GitHub community*, we performed content analysis on the 49 repositories. For each curation repository, we coded the repository name, description, curated items, pull requests, and owners' profiles, which were retrieved from GitHub on September 1, 2014. Open coding strategies as suggested by Strauss (1987) were applied to developing a coding scheme [17]. Themes and concepts were identified, discussed, and refined iteratively among researchers [18]. The results are presented in the following section.

The data files and source code used in this paper are available in the following GitHub repository: https://github.com/chaconnewu/future-internet/.

4. Results

4.1. Curation Repository versus Software Repository

This subsection presents the results of the quantitative analysis of the comparison between curation repositories and software repositories with regards to popularity and different types of activities. It yields insights into how the GitHub infrastructure has been used for curation.

4.1.1. Popularity

We first compared the popularity of curation repositories with that of software repositories using the number of stars, which is an indicator of the status and popularity of a repository on GitHub [8].

Of the top 1433 repositories that had more than 500 stars in our sample, 49 were curation repositories and 1384 were software repositories. The top three most starred repositories were all curation repositories, and six of the top 20 most starred repositories were curation repositories.

To compare the two groups, an independent samples *t*-test was performed to compare the log-transformed mean of the number of stars between curation and software repositories. The reason for log-transformation was because the number of stars that a repository has does not follow a normal distribution. Rather, it is a type of count data. In software engineering, log transformation is an established practice for the analysis of count data [19]. The results showed that curation repositories received a significantly higher number of stars (M = 7.51, SD = 0.99) than software repositories (M = 6.98, SD = 0.65), $t(49.48) = -3.70$, $p < 0.001$. Figure 2 shows the average number of stars (not log-transformed) received by curation repositories and software repositories.

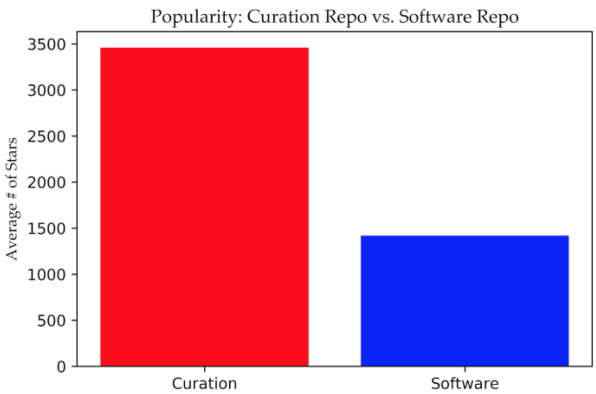


Figure 2. Average number of stars received by curation repositories and software repositories.

4.1.2. Comparison of User Participation for Curation and Software Repositories

In addition to the comparison of popularity, in order to characterize how GitHub features are leveraged in curation repositories, we compared the curation repositories and software repositories in terms of different types of user activities.

The number of activities performed in each repository in our sample was aggregated and log-transformed [19]. Then, independent samples *t*-tests were performed on each type of activity introduced above to compare curation repositories with software repositories. Since eight *t*-tests were performed among the two groups (including comparing the numbers of stars for popularity), a Bonferroni–Holm (1979) correction for multiple testing was applied to avoid the issue of multiple comparisons [20]. The results are shown in Table 1. All Bonferroni–Holm corrected *p* values were significant. Therefore, curation repositories resulted significantly different from software repositories for all seven types of activity. Specifically, curation repositories resulted to have higher numbers of *Fork*, *Push*, and *Pull Request* events and lower number of *Create*, *Delete*, *Issue Comment*, and *Issues* events.

Table 1. Independent sample *t*-tests to compare the log-transformed mean of the number of seven types of activities between curation projects and software projects.

Type of Event	Curation (N = 49)			Software (N = 1384)			*df*	*t*
	Mean	*Mean (log)*	*SD (log)*	*Mean*	*Mean (log)*	*SD (log)*		
Create	1.84	0.80	0.62	16.97	1.85	1.34	65.22	11.01 **
Delete	0.65	0.21	0.57	9.40	1.04	1.22	64.91	9.41 **
Fork	465.69	5.45	1.08	161.81	4.51	1.12	51.73	−6.04 **
Issue Comment	105.10	4.02	1.28	406.91	4.56	1.86	55.48	2.85 *
Issues	27.16	2.62	1.27	140.64	3.77	1.65	53.86	6.12 **
Pull Request	151.69	4.29	1.28	91.52	3.22	1.59	53.38	−5.76 **
Push	405.82	4.57	1.07	165.36	3.85	1.68	56.80	−4.52 **

* indicates the result is significant at $p < 0.01$, ** indicates the result is significant at $p < 0.001$, (*log*) indicates the statistics were calculated on log-transformed value.

GitHub records several types of activity logs (https://developer.github.com/v3/activity/events/types). The important and representative activities include the following: *Create* and *Delete* events indicate a change in the repository structure, such as creating or deleting a Git branch; *Issue Comment* and *Issues* events indicate user participation in issue tracker; *Fork* event happens when a GitHub user clones a repository to his/her own space, which is a prerequisite for a *Pull Request*; a *Pull Request* event indicates that a GitHub user makes some code changes of a repository and wants the owners of the repository to merge the changes, and, thus, it is an indicator of contributions from others; a *Push* event refers to the event of committing a code change in a repository.

The lower frequencies on *Create* and *Delete* events for curation repositories implies that curation repositories are less likely to change the repository structure. The lower frequencies of *Issues* and *Issue Comment* events indicates that curation repositories are less active on issue trackers, which possibly means that they have fewer bug reports and feature requests. The higher frequency of *Push* events shows the active development of curation repositories. The higher frequencies of *Pull Request* and *Fork* events indicate that curation repositories receive more units of contributions from the community.

4.2. The Emerging Role of Curation Repositories

In spite of the enormous amount of popularity curation received, the kind of information needs that it addresses and the role that it plays in the software developers' community are currently unknown. In this section, we applied content analysis to examine the curated contents, the owners, the contributors, and the interactions among owners and contributors in order to understand the roles of curation repositories on GitHub.

4.2.1. The Purposes of Curation Repositories

First, we investigated the contents of curation repositories. As shown in Figure 1, the content of a curation repository is usually a list of curated items of various types, grouped into a set of categories. We coded the topic of each curation repository and the types of curated items and examined the provenance of the curated items.

Overall, 47 of the 49 curation repositories included resources about a topic in the technology industry, one was about curating taco recipes, and one was about curating images.

The Quality Indicators of Curated Contents

We observed that the curation repositories usually include quality indicators in the repository name, description, or README.md file. Out of 49 projects, 27 contained quality indicators, including words or phrases like "awesome" (15 repositories), "must-watch" or "must-read" (3 repositories), "good" (3 repositories), "useful" (2 repositories), and "rock your world", "most influential", "favorite", "cool" (1 repository each). Some curation repositories also included a very specific definition of the quality of their contents, such as:

> *"An awesome package is one that is mature (not recently released), is well maintained, has a good amount of users, has good documentation, follows the best practices, and whose latest release is less than one year old. Awesome Django packages and projects are the ones that inspire and serve as examples."—awesome-django* (https://github.com/rosarior/awesome-django)

Types of Curated Items

Then, we followed the hyperlinks of each curation repository and recorded the type of resource it directed to. The most common types of curated resources were articles and software projects. The complete types of resources and the associated number of curation repositories that included that type of resources is presented in Table 2.

Table 2. Types of curated resources.

Type of Resources	#	Type of Resources	#
Articles	31	Courses	6
Software projects	30	Conferences	4
Websites	24	Audio	4
Books	20	Microblogs	4
Video	12	Q&As	4
Software	11	People	3
Blogs	9	Research Papers	3

Sources of Curated Items

Following Matthews et al.'s (2014) framework for identifying the sources of curated content [12], we examined the sources of the curated items in each curation repository. We coded the repositories using the following scheme: the repositories with more than 75% contents from GitHub were coded as internal curation, those with less than 25% contents from GitHub as external curation, and the rest was coded as hybrid curation.

Around 30% of the curation repositories engaged in internal curation for which the contents mostly came from within GitHub. In 61.2% of the curation repositories, the majority of the curated items came from external resources, which was originated outside of the GitHub realm (Table 3).

Table 3. Sources of the curation repositories.

Curation Type	# Curation Repositories	Percent
Internal Curation	15	30.6%
Hybrid Curation	4	8.2%
External Curation	30	61.2%

As seen in Figure 1, a section contains a themed list of items, which notes the resource name, link, and description. We further coded item description as follows: (1) if the curated list contained nothing except the name of the resources and hyperlink, it was coded as "No description"; (2) if most items (more than 70 percent) of the list had a one- or two-sentence description, it was coded as "Simple description"; (3) if most items (more than 70 percent) of the list had a description which the was either structured or longer than two sentences, it was coded as "Rich description" (Table 4).

Table 4. Levels of description in the curation repositories.

Level of Description	# Curation Repositories	Percent
No description	19	38.8%
Some description	20	40.8%
Rich description	10	20.4%

Around 60% of the curation repositories had some descriptions for each curated item, and 40% had no description at all. This showed the varying of formatting for the curated lists and a lack of standardization.

The results of this subsection outline the general characteristics of curation repositories: they are GitHub repositories that collect and organize alleged high-quality resources about technology industry, where articles and software projects are the most popular types of curated items; the curated items come from both inside and outside of GitHub, and the richness of the description of the curated items varies among different repositories.

4.2.2. The Owners of Curation Repositories

Repositories on GitHub are the results of collaborative efforts between owner and contributors. Owners create a repository, and contributors contribute to curation repositories through sending pull requests. For software repositories on GitHub, many are owned by an organization as a public space for their open source projects. GitHub organizations are group-owned accounts [21]. A repository that is owned by an organization is usually managed by a corporation or a group of developers [21]. For popular individual software repositories, owners are usually "coding rock stars", i.e., GitHub users who have lots of followers and attract community attention [1]. We wondered if the same holds true for the owners of curation repositories.

For the 49 curation repositories, two (4.1%) were owned by an organization, and the other 47 were owned by individuals. For the 1384 software repositories in our sample, 522 were owned by an organization (37.7%). This showed that the majority of the popular curation repositories were owned by individuals, which implies that most curation repositories could be successfully managed without group efforts.

The number of followers of an individual software developer is an indicator of its community status in GitHub [1,2,8]. An example of a "coding rock star" is GitHub user "dhh" [1], who has thousands of followers. Prior work has found that the distributions of the number of followers shows a power-law-like shape [10]. However, the follower distribution of the owners of curation repositories have not been reported. In addition, Matthews et al. (2014) reported that in a large enterprise environment, curators are usually the community leaders [12]. We wondered if the same

holds true also in the social coding site. In the following analysis, we used the number of followers to indicate if the owner of a curation repository was a "coding rock star" or a community leader.

By analyzing the activity log collected from GitHub Archive, we were able to find the number of followers the owners had prior to the creation of the curation repositories. This analysis focused on the 47 repositories that were owned by individuals. The results are presented in Table 5.

Table 5. The number of GitHub users following curation repository owners prior to the creation of curation projects.

# Followers	# Owners	Percent
<10	22	46.8%
Between 10 and 100	15	31.9%
≥100	10	21.3%

Nearly eighty percent of owners had less than 100 followers prior to their creation of curation repositories. It is apparent that the majority of the owners of the popular curation repositories were not community leaders before creating the curation repositories, which is different from the results in Matthews et al. (2014) for enterprise environments, where usually community leaders take the responsibility to curate resources [12].

4.2.3. Participation in Curation Repositories

This subsection concerns about how GitHub collaborative features are used in curation repositories. For 40 out of 49 of the curation repositories, only the owners pushed changes to the main branch of the repository. In 48 of the curation repositories, multiple contributors made contributions through pull requests. The number of contributors a curation repository had is presented in Table 6. The majority of curation repositories had more than 10 contributors.

Table 6. The number of contributors to curation repositories.

# of Contributors	# Curation Repos	Percent
<10	13	26.5%
Between 10 and 100	33	67.3%
>100	3	6.2%

Contributors of curation repositories appropriate the existing functions of GitHub to contribute to the repositories. In 29 curation repositories, there were contributors who made suggestions of resources in the issue tracker. Usually, they forked the curation repository to their own space, made a modification, and sent a pull request to the owner, which is the same process as the collaboration on software repositories on GitHub [2].

For each curation repository, we examined all pull requests that were created before August 31, 2014 to extract the proportion of pull requests with more than two participants involved. At the time of this step of the analysis, one curation repository was no longer available because of copyright issues. One repository did not have any other contributors and thus did not have any issues or pull requests. In addition, we eliminated six curation repositories that had less than three pull requests from this analysis. Therefore, this analysis was performed on 42 repositories. The results in Table 7 show that the contributions made to the majority of curation projects happened between only two people (contributor and owner). Rarely were there cases in which a third person was involved in a pull request to evaluate the resource or to make a comment (Table 7). This may be effective for owners to manage the curation repositories, since after contributors suggest an item, they can evaluate and then make a decision without introducing further steps. However, such mechanism might affect the quality of the alleged high-quality resources, since only two persons examine a new item.

Table 7. Percentage of Pull Requests that have more than two participants involved.

Percentage of Pull Requests That Have >2 Participants	# Repositories	Percent
≤10%	29	69%
Between 10 and 15%	7	16.7%
>15%	6	14.3%

To further understand how curation repositories are changed by pull requests, we retrieved all pull requests that where created before August 31, 2014 for the 42 curation repositories and the line changes for each using the GitHub API. We defined the following three types of pull requests, using "addition" to denote the number of lines added in a pull request and "deletion" to denote the number of lines deleted in a pull request:

- *Type Add*: in this pull request,

  addition > 0 and deletion = 0, or
  addition > 0 and deletion > 0 and (addition - deletion) > deletion

- *Type Delete*: in this pull request:

  addition = 0 and deletion > 0, or
  addition > 0 and deletion > 0 and (deletion - addition) > addition

- *Type Update*: the remaining cases, where addition and deletion were similar.

Following this definition of types of pull requests, for each curation repository, we calculated the percentage of each type of pull requests and plotted the results in Figure 3. The results showed that in most repositories of this sample, *Type Delete* represented a very small portion of the pull requests (less than 10%), while *Type Add* dominated most curation repositories, and *Type Update* represented a limited proportion. This indicated that curation repositories mostly received contributions to add more resources and sometimes to change the existing ones, but only rarely were there contributions to delete resources.

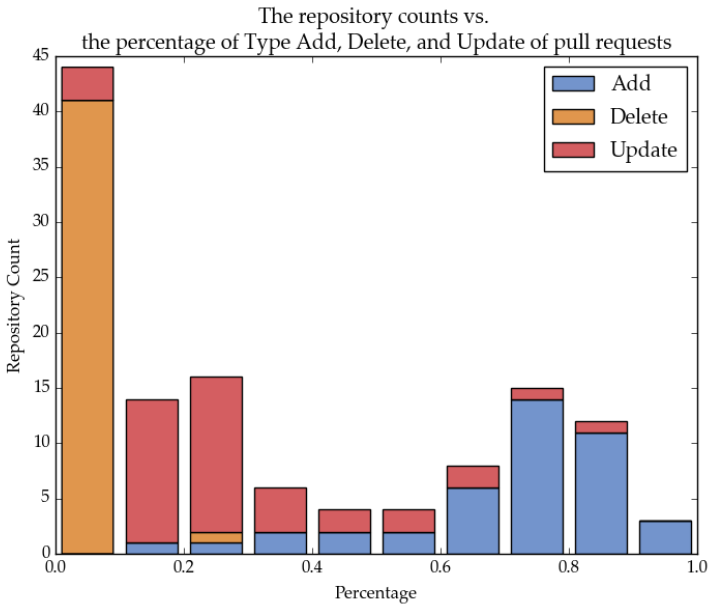


Figure 3. The repository counts versus the percentage of Type Add, Delete, and Update of pull requests.

5. Discussion

Our results illustrate the characteristics of curation repositories regarding: (1) their differences from software repositories on GitHub, (2) the topics and data provenance of the curated items, (3) the leadership, and (4) the collaboration patterns. The emergence of curation repositories and their high popularity have significant implications, which are discussed in this section.

5.1. Implications for the Software Developers' Community

Our results and analysis show that most curation repositories on GitHub select, organize, and preserve different types of high-quality resources, grouping them into different categories that are useful for software developers. The wide popularity of curation repositories indicates that they are well-received in the software developers' community and attract enormous attention. It is likely that curation repositories will become an important way for software developers to share knowledge within the community.

Software developers are increasingly active in participating in a set of different social media sites [22–24]. Although engaging in different sites creates vast opportunities for software developers to find information that is relevant and useful, it also introduces many burdens and challenges, such as (1) the fragmented resources spread over a set of social media sites, (2) the overhead to learn and master different social media channels, and (3) the difficulties in evaluating the quality of information in a large information space [24,25].

Curation on GitHub is likely to be a starting point to address these challenges. Curation repositories centralize fragmented resources all over the Internet. They are located in GitHub, a site many developers are already familiar with, where no other media literacy is required to master the tool, and they involve a collaborative human effort to evaluate the quality of the curated contents. In this way, curation repositories become an important way for the software developers' community to communicate quality resources so that millions of developers do not have to follow different social media sites and filter resources themselves.

5.2. Internal Curation on GitHub

A relatively large proportion of curation repositories are internal-based, which suggests that one important function of curation on GitHub is indexing GitHub-orientated resources.

GitHub has been reported to be a successful tool for self-hosting software repositories and increasing the effectiveness of collaboration [1,2]. As a result, many more developers and organizations began to host software projects on GitHub to allow contributions from others, while also contributing to other software projects. This action led the fast growing of the number of repositories on GitHub. The number of repositories on GitHub reached 10 million in 2013 (https://github.com/blog/1724-10-million-repositories). However, not all repositories hosted on GitHub are of high quality and can be appealing to software developers to use in their own projects or to contribute to. Curation repositories provide valuable navigational support for software repository retrieval, with the help of GitHub features as well as human effort. The high popularity of curation repositories and the large quantity of internally curated resources suggest that such attempt is highly welcomed in the software developers' community. They are likely to save the time and efforts software developers spend in locating the desired software repositories.

Meanwhile, as each curation repository usually supports a single (or several related) software development topic, it also raises the question about the scalability of curation practice. Particularly, with the fast progressing of software engineering, new programming languages, frameworks, and libraries are emerging daily, and thus the number of curation repository will grow as well. As a result, curation repositories as a whole will be fragmented. Some meta curation repository has already emerged (https://github.com/sindresorhus/awesome), which indexes and organizes curation repositories. However, the usage and effectiveness of such meta-lists are unknown. The user evaluation of such

curation repository and design efforts for organizing curation repositories can be an interesting future research direction.

5.3. Implications for the Owners of Curation Repositories

In the review of the characteristics of the owners of curation repositories, we found that most of them are individuals rather than organizations. This implies that the creation and maintenance of a curation repository do not require group efforts. It also suggests that curation in the social coding environment is different from that in the enterprise context, which tends to have a small leadership team that creates and maintains curation repositories [12].

We were also curious to understand if the owners of curation repositories were leaders within the community and found that many were little known prior to their creation of the repositories, for they did not have many followers, which is an indication of the leadership status in GitHub [1,8]. This is an interesting distinction, considering that the GitHub community often favors the work of reputable, well-known developers [1]. The reputation of these curation repositories shows that curators do not have to be community leaders within the social coding site for their curation repositories to be well received.

This result has important implications. Given the popularity and attention the curation repositories receive, it is an opportunity for not well-known software developers to create good curation repositories and make an impact in the community. In addition, the role of curator may become important in the software developers' community, because (1) currently there is no easy way to deal with the information fragmentation nor to address the difficulty in evaluating information [23,24], and (2) the software industry is changing fast, and new technologies are developed while old ones are deprecated every day [16]. It is likely that more curation efforts will be required in the software developers' community.

However, as shown by our results, as most owners of curation repositories are individuals, interesting questions arise on how well a curation repository can scale. The more popular a curation repository becomes, the more contributions it will receive, and the larger it will become. It will become increasingly hard for the owners to add new curated items, track existing ones, and at the same time, evaluate the ones suggested by contributors as the repository expands. It will be interesting to see if organizational efforts will be invested in a curation repository as it expands, or if a community, like open source projects, where there are core and peripheral members, will emerge around a certain curation repository.

5.4. Collaborative Curation on GitHub

The appropriation of GitHub for collaborative curation is of particular interest to this study because GitHub provides a number of collaborative features, such as issue-tracker and pull-request mechanisms, which become standard features in software practices.

Typical curation efforts include selecting, organizing, evaluating resources from multiple resources [5]. In addition to these activities, the owner of a curation repository will also interact with other contributors to curate resources that match the description of the repository. In general, curation repositories adopt the existing practices on GitHub intended for collaborative software development, in which contributors send pull requests (or issues for some curation repositories) to the owner to submit a change to an existing file (specifically, to add a new resource hyperlink). The owner will then evaluate the resources recommended by the contributors and decide whether to merge the change or not. In this way, curation repositories are collaboratively developed by a number of GitHub users. This kind of collaborative curation follows very similar contribution patterns as software repositories [1,2,8]. Therefore, curation repositories not only adopt GitHub features, but also appropriate a part of the software practices on GitHub.

However, this type of appropriation of GitHub for curation differs from that in the enterprise context described by Matthew et al. (2014), which combines a number of tools to organize and curate

resources to cope with information overload, and the community leaders usually curate the bulk part of the resources [12].

Further, our results show that most collaborative curation happens only between two persons, the owner of a curation repository and the contributor. This raises some doubts on whether the opinions of two persons can be well representative for an artifact intended for a large community. It suggests that GitHub features are underutilized in terms of evaluating resources for reaching community consensus. In addition, most pull requests to curation repositories add new resources, rather than deleting the existing ones. This suggests that the contributions to curation repositories rarely consider whether the existing resources are still up to date or appropriate to be included in the list. In the long run, if a curated list keeps growing, it can increase the navigational difficulties and affect the overall quality of the curated resources.

5.5. Design Implications

Our results demonstrate that curation repositories have become an important type of artifact developed in GitHub. The characteristics of such repositories have important implications.

From a design perspective, there are opportunities to design a better interface and provide a better user experience for curation repositories. Open source software projects are a major type of resources for curation, and software projects are created, flourished, and perished all the time. Under the current curation paradigm, there is no effective way to monitor if a curated item in a curation repository is under active development or not without manually checking. As suggested by recent literature, software developers leverage a set of features to make social inferences: for example, recent activity signals the activeness of a repository [1], and the number of stars indicates the community's interest in a project [8]. As most curation repositories are a single page with lengthy content, and, most of the time, the information contained in a curated item is brief, including only the name of the resources and simple description, these types of signals, such as the number of stars and activeness, can be appended to each curated item to help software developers evaluate curated items inside a curation repository.

6. Limitations

Our study is limited in the following aspects. Given the large volume of GitHub repositories, we were only able to examine a sample of them and elected to use the most popular ones as a logical boundary for our sample. Less popular curation repositories might have different properties. In addition, we applied content analysis and a quantitative method to generate the characteristics of curation repositories. However, GitHub users' motivations to create and collaborate on this type of practice and their perception of such repositories are also relevant and interesting, which should be the focus of future research efforts investigating this interesting phenomenon.

7. Conclusions

Curation on GitHub is an innovative appropriation of an existing tool in the software developers' community. In this paper, we studied the characteristics of curation in the software developers' community by investigating curation repositories in the following aspects: (1) the GitHub features used in curation repositories, (2) the characteristics of contents, formats, and owners of curation repositories, and (3) the collaboration patterns in curation repositories. Our results show that curation repositories make use of existing GitHub features to collect, organize, and retain resources about the technology industry. They centralize resources that are spread both inside and outside of GitHub. The comparison of activities between curation repositories and software projects illustrates that curation repositories have a more stable structure, receive more contributions from the community, and do not have multiple owners to lead them.

The emergence of curation on GitHub and its wide popularity has important implications. It suggests that curation may become an important way for software developers to communicate knowledge, as the challenges of participating in multiple social media channels to face a large volume

of resources are increasing [15,24]. Also, the curator role may become more important in the software developers' community, and software developers can curate resources to make an impact. Last, there is potential for appending different pieces of information signals to each curation item inside a curation repository to reduce the navigational cost inside a curation repository.

Acknowledgments: This work is partially supported by CNPq (Grant #248441/2013-2).

Author Contributions: Yu Wu came up with the research idea. Yu Wu, Jessica Kropcznyski, and John M. Carroll conceived and designed the experiments; Yu Wu performed the experiments; Yu Wu, Jessica Kropcznyski, Raquel Prates, and John M. Carroll analyzed the data; Yu Wu, Jessica Kropcznyski, Raquel Prates, and John M. Carroll wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dabbish, L.; Stuart, C.; Tsay, J.; Herbsleb, J. Social coding in GitHub: Transparency and collaboration in an open software repository. In Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, Seattle, WA, USA, 11–15 February 2012; ACM: New York, NY, USA, 2012; pp. 1277–1286.
2. Marlow, J.; Dabbish, L.; Herbsleb, J. Impression formation in online peer production: Activity traces and personal profiles in GitHub. In Proceedings of the 2013 Conference on Computer Supported Cooperative Work, San Antonio, TX, USA, 23–27 February 2013; ACM: New York, NY, USA, 2013; pp. 117–128.
3. Wu, Y.; Wang, N.; Carroll, J.M. *RepoHunter: Supporting Curation Repositories on GitHub*; iConference 2017 Proceedings; iSchools: Sheffield, UK, 2017.
4. Wu, Y.; Wang, N.; Kropczynski, J.; Carroll, J.M. The appropriation of GitHub for curation. *PeerJ Comput. Sci.* **2017**, *3*, e134. [CrossRef]
5. Duh, K.; Hirao, T.; Kimura, A.; Ishiguro, K.; Iwata, T.; Yeung, C.-M.A. Creating Stories: Social Curation of Twitter Messages. In Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media, Dublin, Spain, 4–7 June 2012.
6. Zhong, C.; Shah, S.; Sundaravadivelan, K.; Sastry, N. Sharing the Loves: Understanding the How and Why of Online Content Curation. In Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media, Cambridge, MA, USA, 8–11 July 2013; pp. 659–667.
7. McDonald, N.; Goggins, S. Performance and participation in open source software on GitHub. In Proceedings of the CHI'13 Extended Abstracts on Human Factors in Computing Systems, Paris, France, 27 April–2 May 2013; pp. 139–144.
8. Tsay, J.; Dabbish, L.; Herbsleb, J. Influence of social and technical factors for evaluating contribution in GitHub. In Proceedings of the 36th International Conference on Software Engineering, Hyderabad, India, 31 May– 7 June 2014; pp. 356–366.
9. Chang, S.; Kumar, V.; Gilbert, E.; Terveen, L.G. Specialization, homophily, and gender in a social curation site: Findings from Pinterest. In Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, Baltimore, MD, USA, 15–19 February 2014; pp. 674–686.
10. Lima, A.; Rossi, L.; Musolesi, M. Coding Together at Scale: GitHub as a Collaborative Social Network. In Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media, Ann Arbor, MI, USA, 1–4 June 2014.
11. Rotman, D.; Procita, K.; Hansen, D.; Sims Parr, C.; Preece, J. Supporting content curation communities: The case of the Encyclopedia of Life. *ASIST* **2012**, *63*, 1092–1107. [CrossRef]
12. Matthews, T.; Whittaker, S.; Badenes, H.; Smith, B. Beyond end user content to collaborative knowledge mapping: Interrelations among community social tools. In Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, Baltimore, MD, USA, 15–19 February 2014; pp. 900–910.
13. Munaiah, N.; Kroh, S.; Cabrey, C.; Nagappan, M. Curating GitHub for engineered software projects. *Empir. Softw. Eng.* **2017**, *22*, 3219–3253. [CrossRef]
14. Kalliamvakou, E.; Gousios, G.; Blincoe, K.; Singer, L.; German, D.M.; Damian, D. The promises and perils of mining github. In Proceedings of the 11th Working Conference on Mining Software Repositories, Hyderabad, India, 31 May–1 June 2014; pp. 92–101.

15. Wu, Y.; Kropczynski, J.; Shih, P.C.; Carroll, J.M. Exploring the ecosystem of software developers on GitHub and other platforms. In Proceedings of the Companion Publication of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, Baltimore, MD, USA, 15–19 February 2014; pp. 265–268.
16. Wu, Y.; Kropczynski, J.; Carroll, J.M. Making Big Data Transparent to the Software Developers' Community. In *Big Data: Algorithms, Analytics, and Applications*; Chapman and Hall/CRC: London, UK, 2015; pp. 176–190.
17. Strauss, A.L. *Qualitative Analysis for Social Scientists*; Cambridge University Press: Hong Kong, China, 1987.
18. Lacey, A.; Luff, D. *Qualitative Data Analysis*; Trent Focus: Sheffield, UK, 2001.
19. Mockus, A.; Nagappan, M.; Hassan, A.E. Statistics in software engineering: Pitfalls and good practices. In Proceedings of the 2013 International Conference on Software Engineering, San Francisco, CA, USA, 18–26 May 2013.
20. Holm, S. A simple sequentially rejective multiple test procedure. *Scand. J. Stat.* **1979**, *6*, 65–70.
21. Peterson, K. The Github Open Source Development Process. 2013. Available online: http://kevinp.me/github-process-research/github-process-research.pdf (accessed on 5 November 2017).
22. Singer, L.; Figueira Filho, F.; Cleary, B.; Treude, C.; Storey, M.A.; Schneider, K. 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, San Antonio, TX, USA, 23–27 February 2013; pp. 103–116.
23. Storey, M.A.; Singer, L.; Cleary, B.; Figueira Filho, F.; Zagalsky, A. The (r) evolution of social media in software engineering. In Proceedings of the on Future of Software Engineering, Hyderabad, India, 31 May–7 June 2014; pp. 100–116.
24. Storey, M.A.; Zagalsky, A.; Figueira Filho, F.; Singer, L.; German, D.M. How social and communication channels shape and challenge a participatory culture in software development. *IEEE Trans. Softw. Eng.* **2017**, *43*, 185–204. [CrossRef]
25. Cha, M.; Kwak, H.; Rodriguez, P.; Ahn, Y.-Y.; Moon, S. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, San Diego, CA, USA, 24–26 October 2007; pp. 1–14.