9-2014

An Exploratory Study on Software Microblogger Behaviors

Yuan Tian
Singapore Management University, yuan.tian.2012@smu.edu.sg

David LO
Singapore Management University, davidlo@smu.edu.sg

2014 4th IEEE Workshop on Mining Unstructured Data

An Exploratory Study on Software Microblogger Behaviors

Yuan Tian and David Lo
Singapore Management University
Email: {yuan.tian.2012, davidlo}@smu.edu.sg

***Abstract*—Microblogging services are growing rapidly in the recent years. Twitter, one of the most popular microblogging sites, has gained more than 500 millions users. Thousands of developers are also using Twitter to communicate with one another and microblog about software-related topics such as programming languages, code libraries, etc. Understanding the behaviors of software microbloggers is one of the needed first steps toward building automated tools to encourage software microblogging activities and harness software microblogging to improve various software engineering activities. In this paper, we investigate the behaviors of software microbloggers in terms of their microblogging frequency, generated contents, and interactions among themselves. Our study is based on a dataset that contains more than 13 million microblogs generated by more than 42 thousands software microbloggers.**

I. Introduction

Microblogging is a new way to disseminate information. Every day millions of microbloggers generate a massive amount of publicly accessible contents. Twitter, one of the most popular microblogging sites, has more than 500 million users. Software developers also use microblogs to share information related to software engineering [1], [2]. Understanding the behaviors of these software microbloggers is an essential step towards building good tools to support software microbloggers and harness contents generated by these software microbloggers for various software engineering activities.

Previously, Bougie et al. and Tian et al. have manually analyzed several hundreds software-related microblogs and categorized them into several groups [1], [2]. However, questions about the behaviors of microbloggers who generate these software-related microblogs have not been fully addressed. In this study, we want to complement and extend these past studies by characterizing software microbloggers in terms of their microblogging frequency, their generated contents, and their interactions with one another. We also want to increase the scale of these past studies by investigating a much higher number of microbloggers and their microblogs. In this study, we analyze 42,096 software microbloggers and more than 13 million microblogs.

We investigate the following research questions:

- RQ1 How often software microbloggers microblog about software related contents?
- RQ2 What are the properties of the generated software related contents?
- RQ3 How software microbloggers interact with one another?

The main contributions of our work are as follows:

1) We study 42,096 software microbloggers and their 13,779,180 microblogs to characterize their behaviors in terms of: microblogging frequency, generated contents, and interactions with one another. This is the largest study performed so far on software microbloggers.
2) We compare the properties of software-related microblogs and other microblogs and perform statistical tests to confirm our findings.
3) We build an interaction network of software microbloggers based on their "follow" relationships and study to which degree these software microbloggers are connected.

The structure of this paper is as follows. In Section II, we briefly introduce Twitter. In Section III, we describe our methodology to collect and process microblogs. We answer the research questions in Section IV. We present related work in Section V. We conclude and mention future work in Section VI.

II. Background

Twitter is a popular microblogging site with millions of users all over the world. It provides microblogging service for users to share and exchange information through short messages called tweets. Twitter sets a limitation for the maximum number of characters inside one tweet to be 140.

Figure 1 shows an example of a tweet (i.e., a microblog in Twitter). The example tweet has a hashtag, a url link, and a mention. A hashtag starts with a "# " symbol and it is used to indicate the topic of a tweet. For instance, the example tweet is about "Java EE 7". Due to the 140 character limitation, microbloggers often include url links to external webpages in their tweets to supplement the contents of their tweets. Microbloggers could forward tweets to other microbloggers using mention, which is in the format of "@username".

Microbloggers can post tweets to reply to other microbloggers tweets, and these replies generally start with "@username". They can share others' tweets by retweeting their tweets. A retweet typically starts with "RT @username". They can also "follow" other microbloggers. If a microblogger u follows another microblogger v, u subscribes to the tweets generated by v and all v's tweets would be forwarded to u. In this case, u is a follower of v, and v is a followee of u.

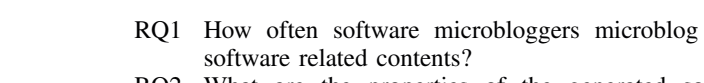

Fig. 1: A microblog from Twitter: (a) retweet, (b) URL link, (c) hashtag

III. Research Methodology

We follow a three step approach: first, we get interesting microbloggers; second, we download their software related

978-1-4799-6793-3/14 $31.00 © 2014 IEEE
DOI 10.1109/MUD.2014.14

microblogs; third, we process their microblogs and extract behavioral features. We describe the details of each step in the following subsections.

A. Getting Interesting Microbloggers

Our goal is to study the behaviors of software microbloggers, which are those generating software related microblogs. However, since there are millions of microblogs generated daily, it is impossible for us to collect all the microblogs and search for the microbloggers who generate software-related microblogs. Therefore, we start by getting a set of interesting microbloggers who are more likely to generate software-related microblogs. Following our previous tool paper [3], we first select a small set of seed microbloggers who are identified to be the top 100 software microbloggers[1]. We then build the set of interesting microbloggers by taking these seed microbloggers and adding others who follow or are followed by at least 5 seed microbloggers. We consider the follow relationships collected on March 1, 2013 (the start time of our experiment). At the end of this process, we have 90,883 microbloggers in our set of interesting microbloggers $\mu Bloggers^{\Uparrow}$.

B. Downloading and Characterizing Microblogs

In the second step, microblogs generated by people in $\mu Bloggers^{\Uparrow}$ are downloaded using the Twitter REST API[2]. We registered for a Twitter whitelisted account which allows us to make up to 20,000 API calls per hour. We downloaded microblogs generated by each microbloggers in $\mu Bloggers^{\Uparrow}$ for the period March 1 to May 31, 2013. Next, we categorize whether a microblog is software related or not. To do so, in this preliminary study, we use the following heuristics. In our tool paper [3], we collected 100 software-related words and phrases from related Wikipedia pages and StackOverflow.com's tags. These words and phrases include JavaScript, C#, etc. and cover three kinds of concepts: 1) Programming Languages, 2) Frameworks, Libraries, and Systems, and 3) Programming Concepts and Methodologies. We use these 100 words and phrases to judge whether a microblog is software-related or not: if a microblog contains at least one of these words or phrases then we deem it as software-related, otherwise, we deem it as not software-related. We perform this heuristics to automate and scale our software-related microblog identification process to large amount of data.

Next, we reduce the set $\mu Bloggers^{\Uparrow}$ to only include those who generate at least one software-related microblog. We refer to the resultant set of people as software microbloggers denoted as $\mu Bloggers^{SE}$. For each software microblogger, we download their microblogs and the follow relationships among themselves. We summarize some basic statistics of our dataset in Table I.

C. Extracting Behavioral Features

In this step, we extract multiple behavioral features for each person in $\mu Bloggers^{SE}$ by processing their microblogs and their follow relationships. These features are divided into 3 groups to answer the research questions raised in Section I.

[1]http://www.noop.nl/2009/02/twitter-top-100-for-software-developers.html

[2]https://dev.twitter.com/docs/api

TABLE I: Dataset description.

Collection Date	March 1 to May 31, 2013
# Software Microbloggers	42,096
# Microblogs	13,779,180
# Software-related Microblogs	520,469
# Follow Links	341,043

For RQ1, we calculate two features related to the frequency or the number of times these software microbloggers generate software-related microblogs. For each software microblogger we compute: 1. average number of software-related microblogs generated per month, 2. proportion of his/her microblogs that are software-related. These features give insights about the absolute and relative frequency of software microblogging activities among software microbloggers, respectively.

For RQ2, we calculate some statistics from the contents that are generated by software microbloggers. For each software microblogger, by analyzing the software related microblogs he/she generates, we compute: 1. average length of the microblogs (in words), 2. average number of URLs per microblog, 3. average number of mentions per microblog, 4. average number of hastags per microblog, 5. proportion of retweets, and 6. proportion of replies. We also compute the above statistics for other general microblogs that a software microblogger generates.

For RQ3, we extract several social features to characterize the interactions among software microbloggers. We first construct a directed graph based on the follow relationships among software microbloggers. Each node in the graph corresponds to a software microblogger and each directed edge stands for a follow link between two software microbloggers. For instance, an edge from node u to node v means software microblogger u follows v. Given this interaction network, we compute:

1) The average degree. The degree of a node u in the network is the number of edges that start from u or end at u.
2) The reciprocity rate. An edge from u to v is deemed reciprocal if there is an edge from v to u. The reciprocity rate is the proportion of edges that are reciprocal.
3) The clustering coefficient. The local clustering for a node u measures how close its neighbors are to being a clique [4]. The clustering coefficient for a network is the average value of local clustering coefficients of all nodes in the network.

IV. Result Analysis

A. RQ1: Microblogging Frequency

To address this research question, we investigate the number of software-related microblogs generated by software microbloggers and the proportion of their microblogs that are software-related. To do this, for each software microblogger in our dataset, we extract two frequency-related features presented in Section III.

Table II shows the number of software microbloggers that generate a particular number of software-related microblogs per month (from March-May 2013). Among the 42,096 software microbloggers that we analyze for 3 months, we find

that 36,245 (86.1%), 5,478 (13.0%), 305 (0.7%), and 68 (0.2%) of them on average produce 1, >1-10, >10-100, and >100 software-related microblogs per month, respectively. Thus, the majority of software microbloggers rarely generate software-related microblogs. Still, more than 5,000 of them produce on average one software-related microblog every 3-15 days. Also, 305 microbloggers on average produce at least one microblog every 3 days. Interestingly, 68 microbloggers produce more than 3 software-related microblogs daily. Many of the microblogs generated by these 68 microbloggers contain software-related news.

TABLE II: Microblogging frequency: average number of software-related microblogs per month.

Avg. # Software-Related Microblogs Per Month	# Software Microbloggers
1	36,245
>1-10	5,478
>10-100	305
>100	68

Table III shows the number of software microbloggers where a particular proportion of their microblogs are software related. Among the 42,096 software microbloggers, we find that for 4,221 (10.0%), 27,809 (66.1%), 9,526 (22.6%), and 540 (1.3%) of them, 1%, >1-10%, >10-50%, and >50% of their microblogs are software related, respectively. For the majority of software microbloggers, only a small proportion of their microblogs (10%) are software related. Still, for more than 9,000 software microbloggers, a good proportion (>10-50%) of their microblogs are software related. Furthermore, 540 software microbloggers generate more software-related microblogs than other microblogs.

TABLE III: Microblogging frequency: proportion of software-related microblogs.

Proportion of Software-Related Microblogs	# Software Microbloggers
1%	4,221
>1%-10%	27,809
>10%-50%	9,526
>50%	540

Although most of the software microbloggers are not very active in generating software related contents, still, there are 305 and 68 software microbloggers that on average generate >10-100 and >100 software-related microblogs per month, respectively. For more than 9,000 software microbloggers, >10-50% of their microblogs are software-related. Interestingly, 540 software microbloggers generate more software-related microblogs than other microblogs.

B. RQ2: Generated Contents

In the second research question, we investigate properties of software-related contents in microblogs that are generated by the 42,096 software microbloggers. We also compare and contrast the software-related and other microblogs that are generated by these 42,096 microbloggers. We consider the six features described in Section III. These content-related features capture various properties of the generated microblogs.

Table IV shows the statistics of all six content-related properties. In this table, we compare the statistics of software-related microblogs with those of their other microblogs. For each microblogger, we compute the averages of the statistics of their software-related microblogs and those of other microblogs. We then report the means of the averages across the microbloggers. We highlight the higher mean for each property. We find that on average software-related microblogs are 16.3% longer (in terms of number of words), contain 13.0% more URL links, and have 46.7% more hashtags. Also, a larger percentage of software-related microblogs are retweets (7% difference). However, on average, software-related microblogs contain 10.7% less mention. Also, a smaller percentage of software- related microblogs are replies (12% difference).

TABLE IV: Properties of generated microblogs. "Std.Dev" is short for Standard Deviation. Column "Sig?" indicates for each property whether there is a statistically significant difference between the mean of software-related and that of other microblogs.

	Software-related		Others		
Properties	**Mean**	**Std.Dev**	**Mean**	**Std.Dev**	**Sig?**
Length	**18.25**	5.00	15.70	3.45	Yes
# URL	**0.52**	0.41	0.46	0.30	Yes
# Mention	0.75	0.57	**0.84**	0.40	Yes
# Hashtag	**0.44**	0.76	0.30	0.39	Yes
% Reply Tweets	16%	0.26	**28%**	0.23	Yes
% Retweet Tweets	**31%**	0.34	24%	0.22	Yes

To further test whether the differences in means are statistically significant, we perform the Wilcoxon signed-rank test [5]. We apply the Wilcoxon signed-rank test because our data is paired: for each microblogger, we have the averages of properties for his/her software-related tweets and those of his/her other tweets. The p-values returned by the Wilcoxon signed-rank test for all six features are less than 0.05, which confirms that the software-related microblogs are significantly different from other microblogs.

Software-related microblogs are statistically significantly longer, contain significantly more URLs and hashtags; however, they have significantly less mentions. Software-related microblogs get retweeted more often but are replied less often than other microblogs.

C. RQ3: Microblogger Interactions

In the third research question, we investigate interactions among software microbloggers. We also compare and contrast software microbloggers and general microbloggers in Twitter. Following the details described in Section III, we generate an interaction network where each node in the network is a software microblogger. Based on this interaction network, we then compute the 3 features described in Section III. These interaction-related features capture various properties of the interactions among software microbloggers.

Table V lists the properties of our software microbloggers network (SMN) and properties of a general twitter network that is reported in a past study [6] (GTN). We note that the average degree of nodes in SMN is only 42% of the average degree of nodes in GTN. This shows that, in general, the number of "friends" (i.e., people that follow or are followed by a microblogger) a software microblogger has that are also software microbloggers, is less than the number of "friends" a general microblogger has. This is intuitive.

TABLE V: Properties of interaction networks.

Properties	Software Microbloggers	General Twitterers
Average Degree	8.10	18.86
Microblogs	13.9%	58%
Clustering Coefficient	0.545	0.106

From the reciprocity rate, we can learn that among software microbloggers, there are less bidirectional relationships. The proportion of reciprocal links in SMN is less than a quarter of that in GTN . Many software microbloggers are interested in contents generated by another group of microbloggers but not the other way round. From the clustering coefficients, we learn that software microbloggers are about 5 times more clustered than general microbloggers. This indicates that the community of software microbloggers is more dense and tightly knitted than that of general microbloggers.

The number of software microblogger "friends" that a software microblogger has is less than the number of "friends" that a general microblogger has. The flow of information among software microbloggers is more unidirectional as compared to the flow of information among general microbloggers. Software microbloggers are more tightly knitted than general microbloggers.

D. Threats to Validity

Threats to internal validity relate to experimental bias. In this preliminary work, we identify software-related microblogs by using 100 software-related keywords identified in [3]. There might be software-related microblogs that are not identified since they do not contain these 100 software-related keywords. In the future, we plan to reduce this threat by using more software-related keywords. In addressing RQ3, we use statistics of a network of 87,897 microbloggers sampled from a Twitter network by Java et al. [6]. We and Java et al. use different sampling strategies and this might introduce bias. In the future, we plan to collect our own network of general microbloggers using a similar sampling strategy (as the one used in this paper) to reduce the possibility of potential bias.

Threats to external validity refer to the generalizability of our findings. We have analyzed 42,096 software microbloggers and more than 13 million microblogs. Still there are many more microbloggers and microblogs that we do not analyze. In the future, we plan to increase the number of software microbloggers and analyze more software-related microblogs.

V. Related Work

A number of studies have proposed tools to better harness social media. Guzzi et al. and Begel et al. propose methods to integrate social media into the software development process [7], [8]. Achananuparp et al. [3] build a visualization tool to help software developers understand trends in software development. Prasetyo et al. propose a technique that can categorize a microblog as relevant or irrelevant to engineering software systems [9]. Other works perform exploratory studies on how social media is currently used by software developers. Tian et al. categorize 300 software-related microblogs into ten groups including: news, commercials, tools & code, Q & A, etc. [2]. Bougie et al. analyze 11,679 tweets that are made by 68 software microbloggers and analyze the topics of the microblogs (i.e., software engineering related, gadgets, current events, daily chatter) and the proportion of tweets that contain mentions, URLs, hashtags, and are retweets [1]. Singer et al. survey around 300 GitHub developers and find that they often use Twitter to keep up with new techniques and new changes in the industry [10].

Our work is closest to the work by Tian at al. [2], Bougie et al. [1], and Singer et al. [10]. Different from the work by Tian et al., we study the behavior of software microbloggers not by categorizing the contents of their microblogs, rather by analyzing their microblogging frequency, six properties (length, number of URLs, etc.) of their generated microblogs, and the interactions among software microbloggers. The work by Bougie et al. has also studied some properties (i.e., mentions, URLs, hashtags, retweets) of microblogs that software microbloggers generate. However, these properties are computed over all microblogs including those about current events and daily chatter. They also consider a much smaller set of software microbloggers (68 versus 42,096). Furthermore they do not compare and contrast software-related and other microblogs, or analyze microblogging frequency, or study interactions among microbloggers. Different from the work of Singer et al., we learn different aspects of developers' behaviors by analyzing their microblogs and interaction network.

There are studies that analyze how developers use other Web 2.0 sites (aside from microblogging sites) [11], [12] or propose tools that leverage these sites to help developers perform their tasks better [13]–[20]. A more complete review of studies that analyze Web 2.0 resources is available in the survey by Tian and Lo [21].

VI. Conclusion and Future Work

In this work, we analyze the behaviors of software microbloggers in terms of their microblogging frequency, generated content, and interactions. We analyze a dataset of 42,096 microbloggers and more than 13 million microblogs collected over a period of 3 months. We find that some software microbloggers generate many (>100) software-related microblogs monthly. Also, for more than 10,000 software microbloggers, more than 10% of their microblogs are software related. For 540 of them, a majority of their microblogs are software related. We also find that software-related microblogs are significantly longer, contain more URLs and hashtags, and are retweeted more often than other microblogs. Furthermore, we find that the community of software microbloggers is more tightly knitted than that of general microbloggers.

In the future, we plan to expand this preliminary study by reducing the threats of validity described in Section IV. We also want to investigate how our results can be leveraged to create tools that extract and summarize useful information from software microblogs for developers.

Acknowledgement. This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office, Media Development Authority (MDA).

REFERENCES

[1] G. Bougie, J. Starke, M.-A. Storey, and D. M. German, "Towards understanding twitter use in software engineering: preliminary findings, ongoing challenges and future questions," in *Proceedings of the 2nd international workshop on Web 2.0 for software engineering*. ACM, 2011, pp. 31–36.

[2] Y. Tian, P. Achananuparp, I. N. Lubis, D. Lo, and E.-P. Lim, "What does software engineering community microblog about?" in *Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on*. IEEE, 2012, pp. 247–250.

[3] P. Achananuparp, I. N. Lubis, Y. Tian, D. Lo, and E.-P. Lim, "Observatory of trends in software related microblogs," in *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*. ACM, 2012, pp. 334–337.

[4] D. J. Watts and S. H. Strogatz, "Collective dynamics of small-worldnetworks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[5] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics bulletin*, pp. 80–83, 1945.

[6] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: understanding microblogging usage and communities," in *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*. ACM, 2007, pp. 56–65.

[7] A. Begel, R. DeLine, and T. Zimmermann, "Social media for software engineering," in *Proceedings of the FSE/SDP workshop on Future of software engineering research*. ACM, 2010, pp. 33–38.

[8] A. Guzzi, M. Pinzger, and A. van Deursen, "Combining micro-blogging and ide interactions to support developers in their quests," in *Software Maintenance (ICSM), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1–5.

[9] P. K. Prasetyo, D. Lo, P. Achananuparp, Y. Tian, and E.-P. Lim, "Automatic classification of software related microblogs," in *ICSM*, 2012, pp. 596–599.

[10] L. Singer, F. M. F. Filho, and M.-A. D. Storey, "Software engineering at the speed of light: how developers stay current using twitter," in *ICSE*, 2014, pp. 211–221.

[11] D. Pagano and W. Maalej, "How do developers blog?: an exploratory study," in *MSR*, 2011, pp. 123–132.

[12] A. Barua, S. W. Thomas, and A. E. Hassan, "What are developers talking about? an analysis of topics and trends in stack overflow," *Empirical Software Engineering*, vol. 19, no. 3, pp. 619–654, 2014.

[13] S. Gottipati, D. Lo, and J. Jiang, "Finding relevant answers in software forums," in *ASE*, 2011, pp. 323–332.

[14] D. Hou and L. Mo, "Content categorization of api discussions," in *ICSM*, 2013, pp. 60–69.

[15] B. Zhou, X. Xia, D. Lo, C. Tian, and X. Wang, "Towards more accurate content categorization of api discussions," in *ICPC*, 2014, pp. 95–105.

[16] Y. Tian, P. S. Kochhar, E.-P. Lim, F. Zhu, and D. Lo, "Predicting best answerers for new questions: An approach leveraging topic modeling and collaborative voting," in *SocInfo Workshops*, 2013, pp. 55–68.

[17] S. Henß, M. Monperrus, and M. Mezini, "Semi-automatically extracting faqs to improve accessibility of software development knowledge," in *ICSE*, 2012, pp. 793–803.

[18] Y. Zhang and D. Hou, "Extracting problematic api features from forum discussions," in *ICPC*, 2013, pp. 142–151.

[19] X. Xia, D. Lo, X. Wang, and B. Zhou, "Tag recommendation in software information sites," in *MSR*, 2013, pp. 287–296.

[20] D. Surian, N. Liu, D. Lo, H. Tong, E.-P. Lim, and C. Faloutsos, "Recommending people in developers' collaboration network," in *WCRE*, 2011, pp. 379–388.

[21] Y. Tian and D. Lo, "Leveraging web 2.0 for software evolution," in *Evolving Software Systems*, 2014, pp. 163–197.