An Exploratory Analysis of Mobile Development Issues using Stack Overflow

Mario Linares-Vásquez, Bogdan Dit, Denys Poshyvanyk
Computer Science Department
The College of William and Mary
Williamsburg, VA, USA
{mlinarev, bdit, denys}@cs.wm.edu

***Abstract*—Question & answer (Q&A) websites, such as Stack Overflow (SO), are widely used by developers to find and provide answers to technical issues and concerns in software development. Mobile development is not an exception to the rule. In the latest SO dump, more than 400K questions were labeled with tags related to mobile technologies. Although, previous works have analyzed the main topics and trends in SO threads, there are no studies devoted specifically to mobile development. In this paper we used topic modeling techniques to extract hot-topics from mobile-development related questions. Our findings suggest that most of the questions include topics related to general questions and compatibility issues, and the most specific topics, such as crash reports and database connection, are present in a reduced set of questions.**

***Index Terms*—Stack Overflow, mobile platforms, mining software repositories, topic modeling.**

I. Introduction

Nowadays, mobile development is related to (i) a reduced set of technologies that allow developers to create mobile applications (apps) for specific software and hardware platforms, as well as (ii) a set of cross-platform tools that allow developers to use models or high-level programming languages to create multi-platform apps. Moreover, the different options of handset vendors that are available in the market have promoted the usage of several programming languages for mobile development (e.g., Android, iOS).

In the case of traditional software development, there are several paradigms, and consequently, different issues and concerns that are specific to the paradigm, the programming language, and the type of application to build. However, there is less evidence and knowledge related to the possible issues associated with mobile development than with traditional software development. In particular, we do not have solid evidence of the most common issues that developers face when developing apps for specific software platforms, such as Android, iPhone, or JavaME.

A recent study by Han *et al.* [1] analyzed fragmentation within Android by extracting topics from bug reports and using topic modeling techniques. Although the authors provide evidence of issues that affect the bugs reported in the Android bug repository, those issues are related to hardware-fragmentation and not include the software-fragmentation perspective. Barua *et al.* [2] used topic modeling to automatically extract the main discussion topics that software developers keep in Stack Overflow[1] (SO). Stack Overflow is a question & answer (Q&A) website widely used by developers to find and provide answers to technical issues and concerns in software development. In the study by Barua *et al.* [2], a total of 973,267 questions and 2,501,720 answers, which represent 27 months of SO activity (from July 2008 to September 2010), were analyzed. The main topics identified by Barua *et al.* [2] in SO are related to: web-related discussions, data management, platform-specific discussions, security, quality assurance and collaboration, knowledge/experience, and general discussions. In addition, two of the conclusions drawn from the analysis were: (i) mobile application development is a trend topic, with an upward-tendency that increments faster than web development; (ii) Android and iPhone development is much more prevalent than Blackberry development. However, Barua *et al.* [2] did not explore the details related to mobile development found in the discussions of SO, and to the best of our knowledge, these details were not addressed in any other research paper.

In this paper, we plan to further explore the issues that developers face when developing apps, by extracting topics representative of issues in mobile development. More specifically, in this paper we analyzed the mobile-development-related discussions from SO, by extracting the main topics that represent those discussions using Latent Dirichlet Allocation (LDA) [3]. Our work is similar to Barua *et al.*'s [2] because it uses LDA, but is different from the type of information analyzed. Because SO is widely used for finding answers to technical issues in programming, our assumption is that the terms found in questions and answers describe the latent structure of the technical issues associated with software development for mobile devices.

Several tags have been used by SO users to label questions related to mobile development. In the latest SO dump (August 2012) there were more than 400K questions labeled with different tags related to mobile technologies (e.g., languages and cross-platform tools). We used those questions and their accepted answers to extract the main discussion topics at two granularity levels: for the entire dataset, and tag specific

[1] http://stackoverflow.com/

corpora. We distinguished between questions that have accepted answers and questions without accepted answers. In addition to the topics analysis, we analyzed if SO contributors are more concerned to provide answers in specific technologies, or for multiple platforms.

All the data used in our study are publicly available at http://www.cs.wm.edu/semeru/data/msr13-so-mobile.

II. Data and Approach

Our study aims at answering the following three research questions (RQ):

- **RQ_1:** *Are there developers that provide accepted answers for several mobile platforms? Do developers provide answers only for a specific platform?* Our interest with this RQ is to validate if SO contributors use two or more platforms concurrently.
- **RQ_2:** *Which are the hot-topics that describe the answered questions related to mobile development in Stack Overflow?* This RQ aims at investigating the conceptual and technical concerns in the questions with accepted answers that software developers face in mobile application development.
- **RQ_3***: Which are the hot-topics that describe the unanswered questions related to mobile development in Stack Overflow?* RQ_3 considers unanswered questions, in contrast to RQ_2, because we are interested in the most important conceptual/technical concerns that characterize questions without accepted answers.

In our analysis, we distinguished between questions with an *accepted* answer and questions with *non-accepted* answers, similarly to the work by Treude *et al.* [4], which distinguished between *successful* and *unsuccessful q*uestions. The questions posted on SO can have *accepted* answers (i.e., answers that are verified and accepted by the question owner) or *non-accepted* answers (i.e., low-voted answers or unrelated ones, which were not validated by the question owner). Our assumption is that questions with only *non-accepted* answers are of interest to developers, yet they are hard to answer due to several factors, such as emerging technologies, scarce online support, etc., and these questions are good indications for revealing the trend of new technologies and approaches that the community should support.

Consequently, for RQ_1 and RQ_2 we used the accepted answers as representatives of *successful* answers in SO, instead of all the answers given to a specific question. For RQ_1, we used the answer owners as representatives of the mobile-developers community. Therefore, we analyzed if the contributors in SO provided *successful* answers to questions labeled in one or more mobile-development-related tag. For RQ_2 and RQ_3 we used Latent Dirichlet Allocation [3], similarly to the work by Barua *et al.* [2], to automatically extract the topics that are present in SO questions and answers, and the topic entropy to select the main topics. For each question with accepted answer we built a document with the terms in the question title, question body, and answer body. However, for RQ_3 we only used the title and body of questions with non-accepted answers.

A. Data Extraction Process

We used the posts from the latest official SO dump (August 2012) provided as a PostgreSQL dump[2] for the MSR 2013 Mining Challenge [5]. Each thread (also called discussion) in SO is composed of a question and a set of answers, and is labeled using predefined or user-defined tags. The contributor who posted the question is called the *question owner* (QO). One of the answers in the thread that is validated by the QO as the correct answer is called the *accepted answer* (AA). Thus, there could be questions without an AA, because no answer is selected by the QO as valid. The contributor, who posts the AA, is called the *accepted answer owner* (AAO).

For our analysis, we considered two types of corpora. The first one, called *aa,* consists of the question title and body, and the accepted answer body. The second corpus, called *naa,* contains only the title and body of the questions for which the QO did not mark an accepted answer. Note that we decided to exclude from the *naa* corpus the answers, because we assumed that the answers not accepted by the QOs do not reflect the same concepts in the corresponding question.

We selected the questions with the following *tags T*, related to mobile technologies: *android, bada, blackberry, iphone* and *ios, java-me, phonegap, symbian, tizen, webos,* and *windows-phone*. For the *ios* and *java-me* tags, we applied regular expressions to exclude the SO tags that contained the tokens *ios* or *java-me* but they were unrelated to mobile development, such as ***ios****tream*, *nag****ios***, *k****ios****k-mode*, ***java-me****lody*, ***java-me****tro-framework.* Using these tags, we built the *aa* and *naa* corpora at two granularity levels: (i) a corpus with the union of all the questions labeled with the tags in *T*, and (ii) a corpus for each tag in *T*.

All the documents were preprocessed using the following steps: (i) extract text from HTML content using the Java Swing HTML Parser[3]; (ii) remove all non-white characters except letters and underscore; (iii) split identifiers using the camel case notation; (iv) remove common words[4] (i.e., stop-words); and (v) stem words using the Porter algorithm for English. For the splitting step we used the `lang` package in the Apache Commons project[5] version 3.3.1; and for the stop-words removal and stemming we used Apache Lucene Core[6] 3.6.0. A description of the corpus is provided in our online appendix.

B. Analysis Method for RQ_1

To answer RQ_1 we computed the number of SO users that posted accepted answers in *n* mobile-technologies, with *n* ranging from one to ten. In addition we computed the number of SO users that posted accepted answers in all the possible subsets of size one and two in the power set of the considered tags (e.g., the number of accepted-answer-contributors in Android questions, Android or iPhone questions, etc.).

---

[2] http://2013.msrconf.org/challenge_data/201208_stack_overflow_postgres_dump.tar.bz

[3] http://docs.oracle.com/javase/1.4.2/docs/api/javax/swing/text/html/parser/package-summary.html

[4] The list of stop-words is included in our online appendix

[5] http://commons.apache.org/lang/

[6] http://lucene.apache.org/core/

TABLE 1. NUMBER OF AAOS GROUPED BY THE NUMBER OF TECHNOLOGIES WHERE THEY POSTED ACCEPTED ANSWERS

Technologies	1	2	3	4	5	6	7	>7
AAOs	34,911	4,377	855	195	53	5	2	0

C. Analysis Method for RQ_2 and RQ_3

We analyzed our corpus using the fast collapsed Gibbs sampling implementation of LDA [6], because it produces equivalent results as standard LDA implementation, yet it is much faster. For more details about LDA we refer the interested reader to [6] [3] [7].

We used the following parameters for LDA: 20 topics (for extracting only the high level topics), 1000 iterations (for convergence), and for the hyper parameters we choose standard values used in the information retrieval community on natural language corpora: $\alpha = 0.01$ and $\beta = 0.01$. Note that we chose a low α value because we were interested in high variability among the topic distribution, to easily identify the dominant topics (see below). However, during our evaluation we tried other combinations of LDA parameters and we observed similar results as for the previously enumerated configuration.

For each type of corpora (e.g., *aa* and *naa*) consisting of the documents from a specific tag (and all documents), we computed two metrics using the *document to topics distribution* matrix θ generated by LDA. The matrix θ has K number of topics and D number of documents and each entry $\theta_{k,d}$ denotes the probability of topic k pertaining to document d.

The first metric *TE(k)*, called **t**opic **e**ntropy (or topic scattering) [8] [9] of topic k is generated using the formula:

$$TE(k) = -\sum_{d=1}^{|D|} \theta_{k,d} \cdot \log(\theta_{k,d}) \quad (1)$$

The topic entropy (*TE*) measures how scattered is a topic across the corpus. For instance, topics with low entropy are concentrated in a small set of documents, whereas topics with high entropy are scattered throughout the corpus. Therefore, we considered topics with high *TE* values as important (i.e., hot-topics). Note that we did not normalize the *TE* as in [8] because we did not need to compare the *TE* across various sized corpora.

The second metric *NDDT(k),* called **n**umber of **d**ocuments with **d**ominant **t**opic k is defined as:

$$NDDT(k) = \sum_{d=1}^{|D|} I(d,k), \qquad I(d,k) = \begin{cases} 1 & \theta_{k,d} = \max_{j=1\ldots K}\{\theta_{j,d}\} \\ 0 & i.o.c \end{cases} \quad (2)$$

A topic k is dominant in document d if and only if $\theta_{k,d}$ is equal to the maximum topic probability among all the topics in d. Therefore, *NDDT(k)* quantifies the number of documents for which topic k is the most important.

III. ANALYSIS OF THE RESULTS

In this section we report the results aimed at answering the three research questions formulated previously.

A. Results RQ_1

Table 1 lists the number of AAOs, grouped by the number of mobile technologies. Posting answers in only one technology is the predominant choice, with 34,911 contributors that represent the 86% of the sample. Providing answers for two technologies is the choice of 11% of the contributors. Finally, 3% of the contributors provided accepted answers in more than two technologies.

TABLE 2. NUMBER OF AAOS THAT CONTRIBUTED IN TWO DIFFERENT TECHNOLOGIES. THE DIAGONAL LISTS THE AAOS IN A SPECIFIC MOBILE TECHNOLOGY. CONVENTIONS USED IN THE TABLE ARE: TZ (TIZEN), BB (BLACKBERRY), ADR (ANDROID), JME (JAVA MOBILE EDITION), PG (PHONEGAP), WP (WINDOWS PHONE), BD (BADA), IP (IPHONE/IOS), SB (SYMBIAN), WOS (WEBOS)

Tech.	**TZ**	**BB**	**ADR**	**JME**	**PG**	**WP**	**iP**	**BD**	**SB**	**WOS**
TZ	**1**	0	0	0	0	0	0	0	0	0
BB	-	**833**	373	213	51	36	201	0	25	9
ADR	-	-	**13,174**	316	323	215	1,599	13	69	26
JME	-	-	-	**702**	18	18	141	1	36	1
PG	-	-	-	-	**586**	27	249	0	10	4
WP	-	-	-	-	-	**1,228**	170	0	47	21
iP	-	-	-	-	-	-	**11,872**	5	3	0
BD	-	-	-	-	-	-	-	**36**	60	24
SB.	-	-	-	-	-	-	-	-	**187**	2
WOS	-	-	-	-	-	-	-	-	-	**73**

Table 2 lists the number of SO contributors that posted accepted answers in two different technologies. Android, iPhone, and Windows-phone are the top three technologies preferred by the AAOs. According to the *developer's mindshare index* reported by VisionMobile [10], 72% of the developers use Android, 56% use iOS, 21% use Windows Phone, 16% use BlackBerry, 7% use Symbian, and 2% use Bada. Our findings show that the order of preferred mobile platforms in SO, is the same than in the developer's *mindshare index* [10]: 44.88% of the AAOs prefer Android, 42.07% prefer iOS, 4.50% prefer Windows Phone, 2.55% prefer BlackBerry, 1.89% prefer JME, 0.54% prefer Symbian, and 0.11% prefer Bada. In the case of cross-platform tools, Phonegap is preferred by 3.26% of the AAOs; meanwhile 0.18% of the AAOs provided answers about Webos. Moreover, in the case of multi-platform contributors, the most preferred "duos" are Android-iPhone with 1,599 contributors and Android-BlackBerry with 316 contributors.

Therefore, for RQ_1 *we conclude that the majority of developers contributing with accepted answers in SO prefer to work in only one mobile platform. However, there are multi-platform contributors that provided accepted answers related to more than one mobile technology.*

B. Results RQ_2 and RQ_3

Columns 2 to 6 of Table 3 address RQ_2, and columns 7 to 11 address RQ_3. The rows list the *TE* and *NDDT* for a subset of topics in both corpora, and five descriptive words (i.e., topic labels) that we selected from the top 15 representative words for the topics. We assigned the topic labels by analyzing the top 15 words related to the topic and by reading the contents of some SO posts with the dominant topic. The complete table is found in our online appendix. We listed the topics ordered descendingly by their *TE* measure. We can observe that for *aa* and *naa* the topics range from a general (i.e., large topic entropy) to a specific scope (i.e., low entropy). For example, topic six, which has the highest entropy (and 29,194 *NDDTs*) encapsulates the broad need of users to find answers to their

TABLE 3. A SUBSET OF TOPICS AND THEIR REPRESENTATIVE WORDS FOR THE *AA* AND *NAA* CORPORA

	Questions with accepted answers (*aa* corpus)					Questions without accepted answers (*naa* corpus)				
Pos	**k**	**NDDT**	**TE**	**Label**	**Words**	**k**	**NDDT**	**TE**	**Label**	**Words**
1	6	29,194	**0.90**	General	need, want, object , data, class	19	38,203	**0.89**	General	want, need, thank, devic, help
2	17	28,571	**0.86**	Compatibility	devic, version, build, need, os	6	28,192	**0.87**	GUI	view, button, screen, event, click
3	10	19,784	**0.83**	GUI	view, control, ui, button, bar	17	16,564	**0.83**	Web browsing	web, url, http, request, page
4	1	17,420	**0.83**	Data types	string, arrai, kei, data, dictionary	8	20,396	**0.82**	IDE	project, test, run, error, install
5	9	18,742	**0.83**	IDE	eclips, devic, java, run, test	12	11,074	**0.79**	GUI	view, control, ui, button, bar
10	7	11,608	**0.81**	WS integration	request, url, respons, post, messag	3	8,142	**0.77**	Media	file,video,player,audio,media
11	13	14,369	**0.81**	Display	view, imag, ui, frame, anim	5	7,047	**0.75**	WS integration	connect, socket, server, respons, except
16	20	10,749	**0.76**	Layout	layout, view, height, width, parent	7	3,233	**0.72**	Maps	locat, map, point, latitud, longitud
17	4	6,032	**0.75**	Database	db, cursor, tabl, sqlite, queri	9	5,347	**0.72**	Graph. edition	draw, bitmap, canva, textur, paint
18	3	5,255	**0.75**	Media	player, audio, video, media, sound	18	3,619	**0.72**	Database	db, cursor, null, column, sql
19	12	4,581	**0.75**	Maps	locat, map, point, latitud, longitud	4	3,107	**0.70**	Crash	java, runtim, thread, error, except
20	19	2,656	**0.69**	Crash	core, system, buffer, armv, dylib	15	1,807	**0.63**	Crash	armv, system, core, dylib, info

questions (e.g., terms *need*, *want*, *question*). As the entropy decreases, the questions become more specific, and they relate to GUI controls, IDE development, web browsing, user input, camera/image preprocessing, web services, etc. The topics with the lowest entropy are related to databases, media players, and maps. Surprisingly, the topics with the lowest entropy (i.e., the one that focuses on very specific problems) are related to errors and crash reports. These posts contain a lot of automatically generated code (e.g., traces), as compared to the cases with high entropy. In general, the topics listed in Table 3 are the same for both corpora, except for topics with labels *Compatibility*, *Data types*, *Display* and *Layout* in *aa* corpus, and topics *Web browsing*, and *Graphics editing* in *naa* corpus.

When analyzing the data specific to the two largest mobile platforms (see our online appendix), Android contains a little over 2,800 answered questions related to crashes, and close to 2,500 question without accepted answers related to crashes, whereas for iPhone/iOS, there are about 1,000 answered crash-related questions, and close to 3,000 crash-related questions with non-accepted answers. Although we cannot directly compare these values, we could estimate that the proportions between unanswered and answered questions related to crashes is much higher for iPhone/iOS than for Android. This could be due to several factors, such as the closed-source model of the iPhone/iOS platform, which could narrow the ability of iOS developers to understand the problem and provide the appropriate solution. Other factor could be the format used to report crashes in both platforms: backtraces and crash reports in iOS could be harder to interpret than logcat-style crash reports in Android. However, these and other possible factors impacting our findings require future investigating.

For RQ_2 and RQ_3 *we conclude that although the same topics are discussed broadly (e.g., General, GUI) in answered questions and questions without accepted answers, there is a specific subset of topics more prone to get accepted answers than others. For example, questions related to data types, compatibility, and layout, are more prone to get an accepted answer, meanwhile questions related to accessing/browsing web content and graphics editing could represent hard-to-answer issues, features not supported yet by the technologies or non-interesting questions.*

IV. FUTURE WORK

We are currently working on making the study reproducible in TraceLab [11] a framework designed for constructing and sharing experiments in Software Engineering. In addition, we plan to replace the process of choosing default parameters for LDA, by leveraging LDA-GA [12], an approach that automatically finds the best suited LDA parameters that produce a near-optimal LDA model, based on the cohesiveness of the topics. More details about integrating this experiment with LDA-GA and TraceLab are found in our online appendix.

V. ACKNOWLEDGEMENTS

This work is supported in part by the United States NSF CCF-1016868 and NSF CCF-1218129 grants. Any opinions, findings and conclusions expressed herein are the authors' and do not necessarily reflect those of the sponsors.

REFERENCES

[1] D. Han, C. Zhang, X. Fan, A. Hindle, K. Wong, and E. Stroulia, "Understanding Android Fragmentation with Topic Analysis of Vendor-Specific Bugs," in *WCRE'12*, 2012, pp. 83-92.

[2] A. Barua, S. W. Thomas, and A. Hassan, "What are developers talking about? An analysis of topics and trends in Stack Overflow," *EMSE,* 2012.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research,* vol. 3, pp. 993-1022, 2003.

[4] C. Treude, O. Barzilay, and M.-A. Storey, "How do programmers ask and answer questions on the web? (NIER track)," in *ICSE'11*, pp. 804-807.

[5] A. Bacchelli, "Mining Challenge 2013: Stack Overflow," in *MSR'13*.

[6] I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling, "Fast Collapsed Gibbs Sampling For Latent Dirichlet Allocation," in *SIGKDD* 2008, pp. 569-577.

[7] M. Steyvers and T. Griffiths, "Probabilistic Topic Models," in *Handbook of Latent Semantic Analysis*, T. Landauer, D. McNamara, S. Dennis, and W. Kintsch, Eds., ed: Lawrence Erlbaum, 2007.

[8] S. W. Thomas, N. Bettenburg, D. Blostein, and A. Hassan, "Studying software evolution using topic models," *SCP,* pp. 1-23, 2012.

[9] P. Baldi, E. Linstead, C. Lopes, and S. Bajracharya, "A Theory of Aspects as Latent Topics," in *OOPSLA'08*, 2008, pp. 543-562.

[10] VisionMobile, "Developer Tools: The Foundations of the App Economy," 2013.

[11] E. Keenan, A. Czauderna, G. Leach, J. Cleland-Huang, Y. Shin, E. Moritz, M. Gethers, D. Poshyvanyk, J. Maletic, J. H. Hayes, A. Dekhtyar, D. Manukian, S. Hussein, D. Hearn, "TraceLab: An Experimental Workbench for Equipping Researchers to Innovate, Synthesize, and Comparatively Evaluate Traceability Solutions," in *ICSE'12*, pp. 1375-1378.

[12] A. Panichella, B. Dit, R. Oliveto, M. Di Penta, D. Poshyvanyk, and A. De Lucia, "How to Effectively Use Topic Models for Software Engineering Tasks? An Approach based on Genetic Algorithms," in *ICSE'13*.