Title	Analyzing Social Behavior of Software Developers Across Different Communication Channels
Author(s)	Iqbal, Aftab
Publication Date	2013
Publication Information	Aftab Iqbal and Marcel Karnstedt and Michael Hausenblas (2013) Analyzing Social Behavior of Software Developers Across Different Communication Channels International Conference on Software Engineering and Knowledge Engineering (SEKE)
Item record	http://www.deri.ie/sites/default/files/publications/iqbal_a_et_al _june_2013.pdf; http://hdl.handle.net/10379/4476

Analyzing Social Behavior of Software Developers Across Different Communication Channels

Aftab Iqbal, Marcel Karnstedt and Michael Hausenblas
Digital Enterprise Research Institute (DERI)
National University of Ireland, Galway (NUIG)
firstname.lastname@deri.org

Abstract

Software developers use different project repositories (i.e., mailing list, bug tracking repositories, discussion forums etc.) to interact with each other or to solve software related problems. The growing interest in the usage of social media channels (i.e., Twitter, Facebook, LinkedIn) have also attracted the open source software community and software developers to adopt an identity in order to disseminate project-related information to a wider audience. Much research has been carried out to analyze the social behavior of software developers in different project repositories but so far no one has tried to study the social communication patterns of developers in other social media channels. We in this paper presents a new dimension to the social aspects of software developers and study if the social communication patterns of software developers is different on project repositories and social media channels (i.e., Twitter).

1 Introduction & Motivation

In software engineering, many tools with underlying repositories have been introduced to support the collaboration and coordination in distributed software development. Research has shown that these project repositories contain rich amount of information about software projects. By mining the information contained in these project repositories, practitioners can depend less on their experience and more on the historical data [14]. Examples of project repositories are [15]: source control repositories, bug tracking repositories, mailing list archives etc. Software developers[1] use these repositories to interact with each other or to solve software-related problems. Much research has been carried out to analyze the social network structure and behavior of software developers by extracting rich information from these project repositories [22, 13, 24].

The growing interest in the usage of online social media channels (e.g., Facebook, Twitter, LinkedIn etc.) have also attracted the open source software community. Open source projects are often found to adopt an identity on these social media channels (e.g., Apache Solr/Lucene[2] on Twitter, MySQL[3] on Facebook) in order to disseminate project-related information (release announcements, major bug fixes etc.) or gather feedback/questions posted by the users. Software developers contributing to open source projects also exists on social media channels. Quite often, they discuss, debate or share experiences with others relevant to a software project using hashtags (e.g., `#apache`, `#maven`, `#hadoop` etc.). Hence, the discussions covering open source projects are not limited to dedicated forums or mailing lists, there also exists huge amount of information on the social media channels. However, on the social media channels, less technical details relevant to the project's architecture, code or bugs are discussed. Much of the information available is regarding the experiences[4] or announcements[5] particular to a software project but such valuable information can not be ignored.

It is worth mentioning that the information related to open source projects are distributed on the Web in heterogeneous data islands i.e., social media channels and project repositories. Hence, there is a need to bridge the connection between project repositories and social media channels as shown in Figure 1. By enabling this connection, we will have an integrated view on the software project which can be exploited to support certain use case scenarios:

- End-users response on a particular release of a software project.
- The popularity of a software project by applying sentiment analysis [17] on social messages (i.e., tweets,

---

[1]In this paper, we use the term "software developers" or "developers" to represent those who have commit rights on the source control repository of a project.

[2]https://twitter.com/SolrLucene
[3]https://www.facebook.com/mysql
[4]https://twitter.com/olamy/status/231031288734285824
[5]https://twitter.com/olamy/status/305334578103582720

posts etc.).

- Keeping track of software developer's social activity related to a software project.
- Analysis of the social behavior of software developers in different communication channels (i.e., social media channels, project repositories).
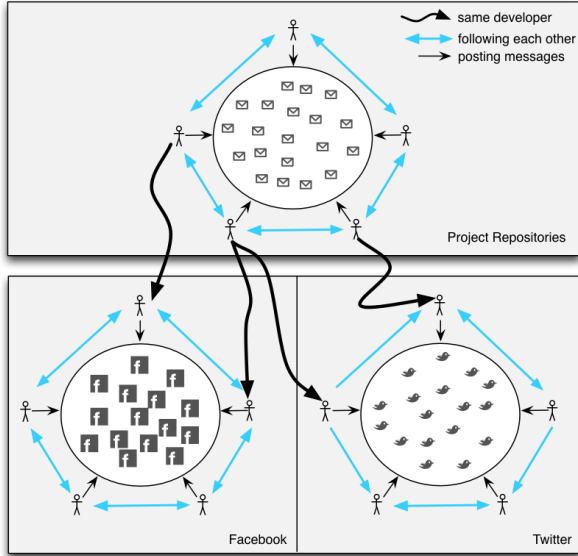

Figure 1: Linking Social Media Channels and Project Repositories.

The social behavior of users have been studied in depth in the past on different communication channels[6] separately. However, to the best of our knowledge, no research work has been done so far on the comparison and analysis of the social behavior of software developers in different communication channels. There is no research work available which analyzes the behavior of software developers communication with each other on the mailing list/bug repositories and their communication on social media channels (e.g., Twitter). This motivates us to study the social communication patterns of software developers in different communication channels.

Among different social media channels available to date, we chose Twitter as a social media channel for this study. Our initial investigation reveals that software developers contributing to open source projects also use and communicate with each other on the social media channels (Twitter in particular). For example, Figure 2 shows the developers social network structure (derived from the communication happened on the mailing list) of an *Apache* project. Among them, few developers are also found on Twitter. We derived the social network structure based on their tweets (e.g.,

mentioning other developers in tweets) which is shown in Figure 3. We removed the labels from nodes (cf. Figure 2 and Figure 3) in order to keep the privacy of developers.
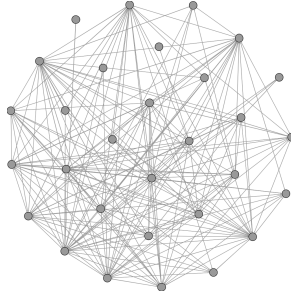

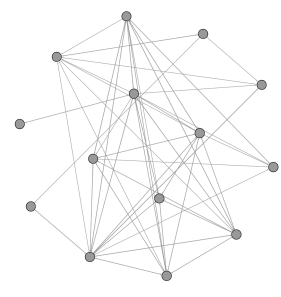Figure 2: Social Relation on Mailing List

Figure 3: Social Relation on Twitter

The social network structures of software developers (cf. Figure 2 and Figure 3) contributing to the same software project provides us the basis to investigate the social behavior of software developers in different communication channels. We will investigate if software developers use Twitter as another medium of communication in contrast to the traditional medium of communication (mailing list, bug tracking repositories, forums etc.). This will laid down the foundations to study the social behavior of software developers with each other in different communication channels. In the current scope of this paper, we will not take into account what software developers are discussing on Twitter but instead we will focus on the communication happened between developers on Twitter in a given period of time and compare it with their communication happened on project repositories for the same period of time.

The contribution of this work is manifolds: we have identified social media channels as a platform which is used by the open source community and software developers to disseminate project-related information to a wider audience. Further we highlighted the need to integrate project repositories and the social media channels (interlinking project-related tweets/posts/hashtags, developer ID(s), project ID(s) etc.) in order to get an integrated view on the software project. We have introduced a new dimension to analyze the social aspects of software developers by taking into account non-traditional communication channels (i.e., Twitter, stackoverflow, LinkedIn, Facebook etc.) which are also used by software developers. We have conducted an initial experiment to investigate the correlation between software developers communication with each other on Twitter and in project repositories by analyzing their communication data over time.

2 Methodology

In this section, we describe our methodology to extract information from different data sources and the usage of a common model and standard format to represent extracted

---

[6]We use the term "communication channels" to refer project related communication channels (e.g., mailing list, bug tracking repository, discussion forums etc.) and online social media channels (e.g., Facebook, Twitter etc.)

information in order to support better query and integration. Further, we describe our approach to compute communication network data which later is used to understand how software developers communicated with each other in different communication channels over the period of time.

2.1 Transforming Data Sources into RDF

With "Linked Data Driven Software Development" (LD2SD) [20], we have introduced a Linked Data-based methodology to relate and integrate data across software repositories explicitly and unambiguously. We propose to use Semantic Web technologies to represent data from different software repositories. As such, we propose to use RDF [21] (Resource Description Framework) as the core, target data model. Once modeled in RDF, the data can be easily integrated, indexed and queried using the SPARQL query[7] standard and associated tools. Finally, the integrated data can be published on the Web using Linked Data principles[8] allowing third parties to discover and subsequently crawl the knowledge, and also allowing to interlink with background or other related information available remotely on the Web. We refer the readers to [16] for details on how these standards would be used. Instead here we focus on transforming data from project repositories and Twitter to RDF. We used our custom written script to convert mailing lists and bug tracking repositories data to RDF. An excerpt of an exemplary RDF representation of an email is shown in Listing 1[9]. Due to space limitations, we do not show the RDF representation of a bug report but refer the readers to [19] for further details on the RDFication process.

```
1  @prefix foaf: <http://xmlns.com/foaf/0.1/> .
2  @prefix sioct: <http://rdfs.org/sioc/types#> .
3  @prefix email: <http://simile.mit.edu/2005/06/ontologies/email#> .
4  @prefix dc: <http://purl.org/dc/elements/1.1/> .
5  @prefix sioc: <http://rdfs.org/sioc/ns#> .
6  @prefix : <http://srvgal85.deri.ie/linkedfloss/mail2rdf/> .
7
8  :A9DB451E-4F0F-435F-8E13-9F4D86996BA4 a sioct:MailMessage ;
9  email:from <http://srvgal85.deri.ie/linkedfloss/aheritier> ;
10 dc:subject "Re: guice & memory usage was:" ;
11 email:body "There's been little to no feedback on beta-2 so ..." .
12 sioc:reply_of :4C5D409A.9060901 ;
13 dc:Date "2010-08-18T22:43:37+02:00";
14
15 <http://srvgal85.deri.ie/linkedfloss/aheritier> a foaf:Person ;
16 foaf:name "Arnaud Heritier" ;
17 foaf:mbox <mailto:aheritier@example.org> .
18 ...
```

Listing 1: An Exemplary Email RDFication.

In order to compute the social communication of a software developer with other fellow software developers on Twitter, we first manually checked if software developers exists on Twitter and using the Twitter account frequently. We found few software developers who does exist on the Twitter platform but tweeted very little (≈10-20 tweets only). We skipped such software developers in the data

crawling and transformation process due to less data available for them. Twitter offers an Application Programming Interface (API)[10] which makes it easy to crawl and collect data from Twitter. We crawled developers Twitter profiles and their tweets using Twitter API and later transformed it to RDF using our custom written scripts. An excerpt of an exemplary RDF representation of a developer's Twitter profile is shown in Listing 2.

```
1  @prefix foaf: <http://xmlns.com/foaf/0.1/> .
2  @prefix ls: <http://lab.linkeddata.deri.ie/linkedfloss/ns/#> .
3  @prefix dcterms: <http://purl.org/dc/terms/> .
4
5  <http://srvgal85.deri.ie/linkedfloss/twitter/brettporter> a foaf:Person ;
6  foaf:accountName "brettporter" ;
7  foaf:name "Brett Porter";
8  foaf:homepage <http://twitter.com/brettporter> ;
9  ls:followers "994";
10 ls:following "707";
11 ls:status_count "6050";
12 dcterms:created "2007-03-26T00:05:50";
13 dcterms:description "CTO of MaestroDev, Director of ASF, and long time
        Maven, Archiva, open source guy. Author and sadly infrequent coder.
        Christian. Husband. Father. Australian." ;
14 ls:location "Sydney, Australia" ;
15 dcterms:language "en";
16 foaf:knows <http://srvgal85.deri.ie/linkedfloss/twitter/aheritier>;
17 ...
```

Listing 2: An Exemplary Twitter Profile in RDF.

Developers often tweets about project-related information using hashtags (e.g., #maven, #lucene) or communicate with other fellow developers by explicitly mentioning his/her name in a tweet. An excerpt of an exemplary RDF representation of a tweet is shown in Listing 3. After transforming the data sources to RDF, we loaded the RDF data sets into our public SPARQL endpoint[11].

```
1  @prefix sioctypes: <http://rdfs.org/sioc/types#> .
2  @prefix dcterms: <http://purl.org/dc/terms/> .
3  @prefix sioc: <http://rdfs.org/sioc/ns#> .
4
5  <http://twitter.com/brettporter/statuses/20971803268>
6         a       sioctypes:MicroblogPost ;
7  dcterms:created "2010-08-12T13:40:49" ;
8  dcterms:creator <http://srvgal85.deri.ie/linkedfloss/twitter/brettporter> ;
9  sioc:content "@aheritier the template hasn't been updated for the
        Confluence 3 upgrade, since we don't typically use the static
        rendering" ;
10 sioc:id "209950243918327809" ;
11 sioc:mentions <http://srvgal85.deri.ie/linkedfloss/twitter/aheritier>
12      .
```

Listing 3: An Exemplary Tweet RDFication.

In this paper, we are not focusing on interlinking developer's information and project related tweets from Twitter to the various software artifacts (i.e., bug, email, commit ID, source-code, developer ID etc.) contained in project repositories. Hence, we do not present any approach on creating owl:sameAs links between relevant entities across different data sources. Different approaches [23, 11, 18, 19] could potentially be utilized in order to achieve the interlinking between Twitter data sets and software artifacts but it is not in the current scope of this paper.

2.2 Social Relation Computation Approach

The social network structure based on the mailing list archives was constructed by using the reply structure of the

---

email threads for direct communication among developers. This approach defines a link as the interaction between the poster of actual email and the replier to the poster email. For example, Listing 1 indicates that the email is a reply of another email (cf. line#12). Hence, we can easily query the poster of email `4C5D409A.9060901` (cf. line#12) in order to create a social link between both software developers. In the case of bug tracking repository, link was defined based on the comment posted by a software developer on a particular bug and the immediate previous commenter on the same bug. The social network structure based on the Twitter data was constructed by exploiting the common practice of using well defined markup in a tweet: @ followed by a user identifier address the user. This approach defines a link as the interaction between the software developer who posted the tweet and the software developer mentioned in the tweet. For example, Listing 3 indicates that the tweet mentions a software developer which is also reflected through `sioc:mentions` property (cf. line#11). Hence, we can create a social link between both software developers (`aheritier` and `brettporter`). Social relations between developers on the Twitter was extracted by querying the Twitter data sets using `sioc:mentions` predicate property. Social relation between any 2 software developers were computed only if both software developers communicated directly to each other on the project repositories and the Twitter platform. Periods without any communication are common as software developers may still contribute to the same project even if they haven't communicated for many days. To tackle this issue, communication between software developers were captured on monthly basis where each month value represent the number of times both software developers communicated directly to each other. Collecting communication data on monthly basis provides good amount of data for every pair of software developers in order to analyze their social communication patterns over the period of time. The initial time-stamp for calculating the social relation between any 2 software developers were computed by comparing the earliest dates where communication happened between both software developers on the project repositories and Twitter. The later date was then used as the starting time-stamp to compute the social relation between both software developers. For example, let say earliest communication happened between 2 developers on the project repositories was `2008-05-25` and on the Twitter was `2010-03-15`. Thus, we will consider `2010-03-15` as the starting time-stamp and compute monthly social interaction between both software developers on the project repository and Twitter over the period of time.

3 Evaluation

Before we discuss the results of our evaluation, we describe the projects selected for evaluation. We gathered data from project repositories of 10 *Apache* projects (c.f Table 1). The reason of choosing *Apache* projects is that the repositories are on the Web and available to download (i.e. mailing list archives, bugs, commit logs etc.). We selected data from the beginning of each *Apache* project to date. The primary source of communication among developers in *Apache* projects is through mailing lists. Most *Apache* projects have at-least 3 different mailing lists: `user`, `dev` and `commits` but some projects have more than 3 mailing lists (e.g., `announcements`, `notifications` etc.). For our study, we downloaded only the `dev` mailing list archives of each *Apache* project under consideration. The reason is, software developers communicate often with each other on the `dev` mailing list rather than on any other mailing list. From the source-control repository data sets of each *Apache* project, we extracted a list of software developers who made commits to the project. Later, we manually checked if these software developers also exist on Twitter and using the Twitter account frequently. Table 1 shows for each *Apache* project the number of software developers who have made commits to the source-control repository and the developers found on Twitter.

Apache Projects	Developers (SVN)	Developers (Twitter)
Apache Camel [1]	36	24
Apache Directory [2]	51	11
Apache Felix [3]	47	17
Apache Hadoop [4]	97	35
Apache Logging [5]	37	7
Apache Lucene [6]	51	18
Apache Maven [7]	40	10
Apache Mina [8]	28	9
Apache MyFaces [9]	82	16
Apache OfBiz [10]	25	11

Table 1: Developers Contributed to Apache Projects and found on Twitter.

The results in Table 1 shows good evidence of the existence of software developers on Twitter. Although, not all software developers contributing to the *Apache* projects found on Twitter. Based on the methodology described in previous section, we found 107 distinct pair of software developers who communicated with each other on the project repositories and Twitter. In the specific case of *Apache OfBiz* project, we didn't find even a single pair of software developers who communicated with each other on project repositories and Twitter. For each pair of software developer, we computed how many times both software developers communicated directly with each other on the project repositories and Twitter, on monthly basis. For an example, we show the communication happened between a pair of software developers over the period of time in Figure 4. The figure shows that communication pattern among both software developers is different throughout the time period under consideration. For example, in `2009-08-12` both developers communicated directly with each other 7 times on the project repositories in contrast to 6 times commu-

4

nication on Twitter for the same month. Further, we see few months where both developers didn't communicate at all (e.g., `2011-09-12`) and in certain months they appear to communicate on only one communication channel (e.g., `2011-03-12`).
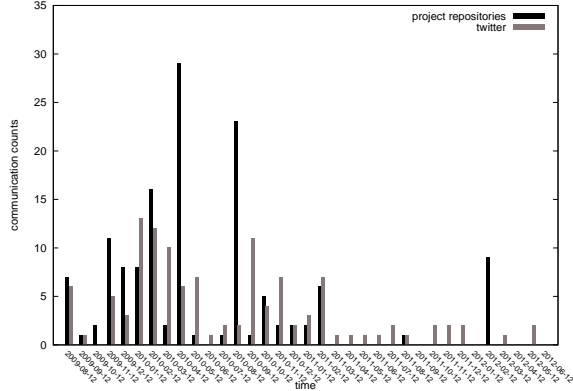

Figure 4: Social Communication Between 2 Software Developers on Different Communication Channels.

Given the social dynamics of software developers in different communication channels (cf. Figure 4), we evaluated if there is a correlation between both developers communication pattern on project repositories and Twitter by measuring Pearson's correlation. The Pearson correlation test based on the communication data between both developers yielded a correlation value, `r=0.447`. The `r` value indicates that the social communication between both developers on different communication channels is significant and show a positive correlation of developers communicating on different communication channels.

In order to find if the communication among software developers is directly proportional on different communication channels, we aggregated the monthly communications occurred on the project repositories and Twitter for every pair of software developers. We plotted the resulting graph in Figure 5. The graph shows that for majority of software developers, the communication on project repositories and Twitter is not directly proportional.

The highest correlation value we found is a developer pair with the value (193,105) where 193 indicates the communication counts on project repositories and 105 on Twitter. The Pearson correlation was calculated for that particular pair based on their monthly communication data which resulted in `r=0.522`. Similarly, the lowest correlation value found is a developer pair with value (107,1) which resulted in `r=-0.040`. We calculated the Pearson's correlation for all developer pairs and computed the mean and median value which is shown in Table 2.

Developer pairs	Mean	Median
107	0.191	0.12

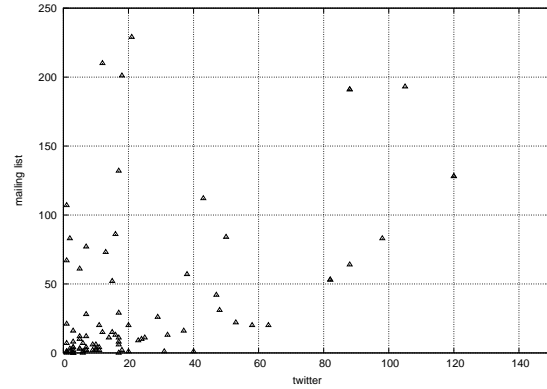Table 2: Mean and Median of Correlation Values

Figure 5: Scatterplot of Twitter vs. Project Repositories Communication Between Software Developers

Based on the results in Figure 5 and Table 2, we found that the correlation between developers communication on Twitter and project repositories is not strong. One potential reason could be the 140 characters limitation on Twitter which keeps software developers to communicate more through traditional communication channels (i.e., mailing list, IRC channels, forums etc.). Other reason might be the usage of Twitter to communicate non-work related activities. However, it is quite interesting to observe how developers communicate with other fellow developers in different communication channels. In our experiment, we found many developers who communicated less on Twitter but their communication on project repositories were strong.

As a next step for future work, we will also take into account the social communication graph of a developer with fellow developers as well as non-developers. Based on that, we will be able to understand if developers communication pattern on Twitter is low in general or if they have less communication only with fellow developers.

4   Conclusion & Future Work

In this paper, we have motivated and introduced a new dimension to the analysis of social dynamics of software developers by taking into account social media channels. The usage of social media channels is becoming popular among open source software community and software developers to disseminate project-related information to a wider audience. This motivated us to investigate if the communicational behavior of software developers is same across different communication channels or different from each other. Our initial results based on the data of 10 different *Apache* projects shows that there is very low correlation between software developer communications across different communication channels. Further, our results shows that the social communication between software developers on Twitter is comparatively low than the traditional communication

channels (i.e., mailing lists, bug tracking repositories etc.).

The work proposed in this paper laid down the idea of taking into account all possible social media channels which developers could possibly use to communicate with each other. Based on that, researchers will be able to measure and compare the hierarchy and centralization of software developers in different communication channels in contrast to previous studies where researchers had been using only mailing lists, bug tracking repositories or discussion forums [24, 12]. Furthermore, integrating social messages/posts to project-related artifacts will open up new research challenges allowing to analyze the impact of end-user's response on the success/failure of an open source project.

In the near future, we will take into account stackoverflow communication network data and further analyze the behavior of software developers communication patterns across a variety of communication channels.

Acknowledgment

References

[1] http://camel.apache.org/.

[2] http://directory.apache.org/.

[3] http://felix.apache.org/.

[4] http://hadoop.apache.org/.

[5] http://logging.apache.org/.

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

[7] http://maven.apache.org/.

[8] http://mina.apache.org/.

[9] http://myfaces.apache.org/.

[10] http://ofbiz.apache.org/.

[11] M. Conklin. Project entity matching across floss repositories. In *Proceedings of 3rd International Conference on Open Source Systems*, 2007.

[12] K. Crowston, , K. Crowston, and J. Howison. Hierarchy and centralization in free and open source software team communications. *Knowledge Technology Policy*, 18:65–85, 2005.

[13] K. Crowston and J. Howison. The social structure of open source software development teams. In *First Monday*, 2003.

[14] A. E. Hassan. The Road Ahead for Mining Software Repositories. In *Future of Software Maintenance (FoSM) at Int. Conf. on Software Maintenance(ICSM)*, 2008.

[15] A. E. Hassan, A. Mockus, R. C. Holt, and P. M. Johnson. Guest editor's introduction: Special issue on mining software repositories. *IEEE Trans. Softw. Eng.*, 31(6):426–428, 2005.

[16] T. Heath and C. Bizer. Linked data: Evolving the web into a global data space (1st edition). *Synthesis Lectures on the Semantic Web: Theory and Technology*, 1(1):1–136, 2011.

[17] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA, 2004. ACM.

[18] A. Iqbal and M. Hausenblas. Integrating developer-related information across open source repositories. In *IEEE 13th International Conference on Information Reuse and Integration (IRI), 2012*, 2012.

[19] A. Iqbal and M. Hausenblas. Interlinking developer identities within and across open source projects: The linked data approach. *ISRN Software Engineering*, 2013.

[20] A. Iqbal, O. Ureche, M. Hausenblas, and G. Tummarello. LD2SD: Linked Data Driven Software Development. In *International Conference on Software Engineering and Knowledge Engineering (SEKE 09)*, 2009.

[21] G. Klyne, J. J. Carroll, and B. McBride. Resource Description Framework (RDF): Concepts and Abstract Syntax). W3C Recommendation 10 February 2004, RDF Core Working Group, 2004.

[22] Y. Long and K. Siau. Social network structures in open source software development teams. *Journal of Database Management*, 18(2):25–40, 2007.

[23] G. Robles and J. M. Gonzalez-Barahona. Developer identification methods for integrated data from various sources. *SIGSOFT Softw. Eng. Notes*, 30(4):1–5, 2005.

[24] A. Wiggins, J. Howison, and K. Crowston. Social dynamics of floss team communication across channels. In *Fourth International Conference on Open Source Software (IFIP 2.13*, 2008.