Evaluating the Accuracy of Collaboration Networks for Free Software Development Projects

Michael Catanzaro
Igalia S.L.
Email: mcatanzaro@igalia.com

***Abstract*—Several recent papers apply social network analysis techniques to study software development practices in free software projects. We demonstrate serious, previously-unreported methodological issues with a recently-published work on software development collaboration networks, showing that its network construction process suffered from serious sources of error. Research on collaboration networks must be more carefully considered in order to avoid arriving at invalid results due to the effects of errors and confirmation bias.**

I. Introduction

Recently, many software companies have moved towards allowing employees to work remotely, in order to attract more talented employees from across the world, or to save on costs of office space. Of course, working remotely increases the difficulty of collaboration with other employees. Recent software engineering research, such as [1], seeks to determine whether distributed development teams can function effectively by examining existing collaboration in free software projects, which have historically been developed by distributed groups of programmers. This is done by constructing networks in which developers are represented by nodes and collaboration between developers by edges, and visually inspecting the results.

This paper evaluates the methodology and conclusions of [1], which focuses on collaboration networks for the WebKit project. Unfortunately, [1] is found to suffer from several methodological issues. After providing in Section II brief background information on the WebKit project and on the social network and ethnographic analyses performed in the original paper, Section III examines these methodological issues and discusses possible remedies. We conclude in Section IV with a warning that researchers must be skeptical of conclusions drawn from visual inspection of collaboration networks. A more rigorous approach to such analysis is required.

II. Background

We first examine basic background on the WebKit project needed to understand the context of this work, then discuss some points regarding the social network analysis and online research performed for [1].

A. WebKit

WebKit is a popular free software web rendering engine, most notable for its use in Apple's Safari web browser. It is also used by a variety of other OS X, iOS, Windows, and GNU/Linux applications. Originally forked by Apple from KDE's KHTML rendering engine, it rose to prominence in 2008 with the release of Google Chrome, a then-new browser based on WebKit. WebKit quickly became the web engine of choice for almost any purpose, due to its focus on embedding APIs at a time when its only serious competitor, Mozilla, moved to eliminate its own support for embedding to simplify the development of Firefox [2], driving embedders to WebKit. Soon a large number of industry players—most any company that needed to display web content in a software application or consumer product—were committed to WebKit development.

After the release of Chrome, Google quickly became the largest contributor to WebKit and held this position for several years. However, there existed several technical differences between Google and Apple's use of WebKit. Notably, Google implemented its multi-process architecture inside Chromium, above the WebCore component of WebKit, whereas Apple implemented its multi-process architecture in the WebKit2 component of WebKit, which Google did not use. This and other technical differences caused significant friction in the project, and ultimately led to Google's decision in 2013 to fork the WebCore component into Blink, which now resides in the Chromium project [3]. Since then, many companies besides Google have switched to Chromium and ceased contributing to WebKit.

[1] presents the claim that WebKit2 is a fork of WebKit, and that Blink is a fork of WebKit2. It includes a flow chart to represent this. This is not correct. WebKit2 is one component within the WebKit project, not a fork. Blink is a fork, but of the WebCore component of WebKit, not of WebKit2.

B. Social Network Analysis

An example of a collaboration network from [1] is shown in Figure 1. The paper reports that this network represents collaboration between September 2008 (when Google began contributing to WebKit) and February 2011 (the departure of Nokia from the project); however, it actually represents the period between February 2011 and July 2012 [4]. The depicted size of each node is proportional to its degree centrality. Note that this network is inaccurate as it is affected by the issues discussed in Section III.

The paper does not precisely describe the methods used to study the network, referring to the entire process as "social network analysis." Although the precise details of this analysis are not essential to the work, it would be desirable for the paper to have described its analysis process in more

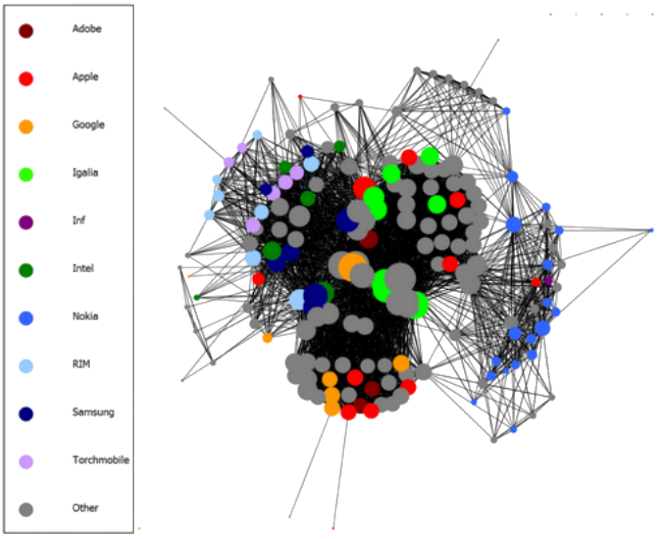

Fig. 1. A collaboration network from [1], reproduced here in its original form in accordance with the terms of its CC BY-NC-SA 4.0 license [4].

detail, rather than in vague terms. We know that community detection was performed using fast modularity maximization [5] to generate an alternate visualization of the networks [4]; however, the results were not presented in [1], possibly due to space constraints.

C. Virtual Ethnography

Virtual ethnography is a new term for the process of studying cultures by reading Internet resources [6], such as online news articles, corporate press releases, and developer blog posts. The authors of [1] performed extensive ethnographic research on the WebKit project to obtain an understanding of the history of the project. They used this research to select key events in the project history, such as the launch of the Chrome web browser, the departure of Nokia from the project, and the beginning of patent lawsuits between Apple and Samsung, as discussed in [1]. The authors then created collaboration networks that reflected only commits made between those events.

The authors used their knowledge of the project's contributors and history to arrive at conclusions based on the collaboration networks. For instance, they observed that companies that appear on the peripheries of the collaboration networks were less "willing to contribute back to the existing projects on an ongoing basis, to assure that the technology continued to meet their [...] needs," etc. Although their conclusions are reasonable and most likely correct, we show in Section III that the collaboration networks these conclusions were drawn from contain so many errors as to be essentially meaningless. Because accurate conclusions were drawn from meaningless networks, it is clear that the prior ethnographic study led to confirmation bias in drawing specific interpretations from the collaboration networks.

III. Methodological Issues

We examine various methodological issues discovered by analyzing [1]. Section III-A discusses the effects on the collaboration network of choosing a poor definition of collaboration. Section III-B discusses a major source of error in detecting the company affiliation of many contributors. Section III-C describes a serious mistake in the data collection process. Each of these issues is quite severe, and any one taken alone calls into question the validity of the entire study. It must be noted that such issues are not necessarily unique to [1], and must be kept in mind for all future studies that utilize collaboration networks.

A. Defining Collaboration

Obviously, the precise definition used to model collaboration has tremendous impact on the usefulness of the resultant collaboration network. Many collaboration networks are built using definitions of collaboration that are self-evidently useful. For example, [7] considers collaboration networks of movie actors, where nodes represent actors and an edge exists between two nodes if the corresponding actors performed together in the same movie. [8] examines coauthorship networks, where nodes represent researchers and an edge exists between two nodes if the corresponding researchers coauthored an academic paper together. There is little doubt that, in such cases, edges in the network represent real-world collaboration.

[1] adopts a similar approach to building collaboration networks, where developers are represented by nodes, and an edge exists between two nodes if the corresponding developers modified the same file in the time period under consideration for the construction of the network. However, it is not clear that this definition of collaboration is actually useful. Consider that it is a regular occurrence for developers who do not know each other and may have never communicated to modify the same files. Consider also that modifying a common file does not necessarily reflect any shared interest in a particular portion of the software project. For instance, a file might be modified when making an interface change in another file, or when fixing a build error occurring on a particular platform. Such occurrences are, in fact, extremely common in the WebKit project. Additionally, consider that there exist particular source code files that are unusually central to the project, and must be modified more frequently than other files. It is highly likely that almost all developers will at one point or another make some change in such a file, and therefore be connected via a collaboration edge to all other developers who have ever modified that file. Figure 2 shows a snapshot of the Subversion (SVN) revision history for one such central file. Notice that each developers who recently modified this file was working on entirely unrelated projects. This is typical.

It is true, as assumed by [1], that particular developers work on different portions of the WebKit source code, and collaborate more with particular other developers. For instance, developers who work for the same company typically, though not always, collaborate most with other developers from that same company. However, [1]'s naive definition of collaboration

Age	Author	Log Message
6 days	mmaxfield	Addressing post-review comments after r200116 …
12 days	mmaxfield	[WK2] [OS X] Create API for switching RTL scrollbar policy
2 weeks	mitz	<rdar://problem/25893246> WebKit2 has failed to build: err
2 weeks	ryanhaddad	Take 2 for fixing builds that do not support AVKit Unreview
2 weeks	ryanhaddad	Fix builds that do not support AVKit Unreviewed build fix. *
3 weeks	carlosgc	[GTK] WebKitWebView should propagate wheel events nc
3 weeks	timothy_horton	Swipe view gesture should be reversed in right-to-left cor
3 weeks	carlosgc	Pending API Request URL is wrong after reloading …
3 weeks	carlosgc	Pending API request URL no set when loading Data, Altern
3 weeks	commit-queue	Unreviewed, rolling out r199660. …
3 weeks	carlosgc	Pending API request URL no set when loading Data, Altern
3 weeks	jer.noble	Allow WebVideoFullscreenManager and Proxy to be used
3 weeks	andersca	[Mac] Add API for open panel handling to WKWebView …

Fig. 2. Recent WebKit SVN revision history for the file WebPageProxy.cpp, truncated for legibility. The commit messages reveal that the contributors who recently modified this file were working on unrelated projects.

should ensure that most developers will be considered to have collaborated equally with most other developers, regardless of the actual degree of collaboration. For instance, consider developers A and B who regularly collaborate on a particular source file. Now, developer C, who works on a platform that does not use this file and would not ordinarily need to modify it, makes a change to some cross-platform interface in another file that requires updating this file. Developer C is now considered to have collaborated with developers A and B on this file! Clearly, this is not a desirable result, as developers A and B have collaborated far more on the development of the file. Moreover, consider that an edge exists between two developers in the collaboration network if they have ever both modified *any* file anywhere in WebKit; then we can expect to form a network that is almost complete. It is evident that some method of weighting collaboration between different contributors would be desirable, as the unweighted collaboration network does not seem useful.

One might argue that the networks presented in [1], such as the network in Figure 1, clearly show developers exist in subcommunities on the peripheries of the network, that the network is clearly not complete, and that therefore this definition of collaboration sufficed, at least to some extent. However, this is only due to another methodological error in the study. Section III-C explains how the study managed to produce collaboration networks with noticeable subcommunities despite these issues.

We note that the authors of [1] chose this same definition of collaboration in their more recent work on OpenStack [9], so there exist multiple studies using this same flawed definition of collaboration. We speculate that this definition of collaboration is unlikely to be more suitable for OpenStack or for other software projects than it is for WebKit. The software engineering research community must explore alternative

Email Domain	Affiliation	Commits
apple.com	Apple	8760
webkit.org	Unknown	1523
igalia.com	Igalia	1014
gmail.com	Unknown	445
samsung.com	Samsung	275
crf.canon.fr	Canon	206
navercorp.com	Naver Corporation	129
outlook.com	Unaffiliated	93
cs.washington.edu	Apple	43
collabora.co.uk	Collabora	29

TABLE I
EMAIL DOMAINS USED BY TOP WEBKIT CONTRIBUTORS IN 2015

models of collaboration when undertaking future students of software development collaboration networks—in particular, models with edge weights—in order to more accurately reflect collaboration.

One particular model of collaboration that might work well for WebKit specifically is to measure collaboration between committer and reviewer. In the WebKit project, committers must have all substantial patches approved by a WebKit reviewer. The patch review process reflects actual collaboration between committer and reviewer, and would likely serve as a more suitable starting point for developing collaboration networks, especially if edge weights are employed. However, this approach would not work in general, as WebKit's review procedures are not typical of other free software projects.

B. Detecting Contributor Affiliation

One difficulty when building collaboration networks is the need to correctly match each contributor with the correct company affiliation. Although many free software projects are dominated by unaffiliated contributors, others, like WebKit, are primarily developed by paid contributors. Table I displays the companies that contributed the most to WebKit in 2015, based on the number of times a particular email domain appears in WebKit changelog entries made during that year.[1]

Note the presence of two emails domains categorized as Unknown: webkit.org and gmail.com.[2] Many developers commit to WebKit using personal email accounts such as GMail accounts; additionally, many developers use webkit.org email aliases, which were previously available to active WebKit contributors. These developers may or may not be affiliated with companies that contribute to the project. Use of personal email addresses is a source of inaccuracy when constructing collaboration networks, as it results in an undercount of corporate contributions. For instance, Table II lists a small sample

[1]The University of Washington domain listed in Table I is shown as associated with Apple because the only contributor to use this domain is an Apple employee.

[2]outlook.com is classed as Unaffiliated rather than Unknown because only one developer used this email domain, and he is known to be unaffiliated with companies contributing to WebKit development.

Contributor	Email Domain	Actual Affiliation
Adam Barth	webkit.org	Google
Alex Christensen	webkit.org	Apple
Csaba Osztrogonác	webkit.org	University of Szeged
Daniel Bates	webkit.org	Apple
Eric Seidel	webkit.org	Google
Ryosuke Niwa	webkit.org	Apple
Sam Weinig	webkit.org	Apple
Sukolsak Sakshuwong	gmail.com	Apple

TABLE II
PERSONAL EMAIL ACCOUNT DOMAINS USED IN CHANGELOG ENTRIES BY SELECTED WEBKIT CONTRIBUTORS, AND THEIR ACTUAL AFFILIATIONS

```
"Mario Sanchez Prada" : {
   "emails" : [
      "mario@webkit.org",
      "mario@endlessm.com",
      "mario.prada@samsung.com",
      "msanchez@igalia.com"
   ],
   "expertise" : "WebKitGTK+, Accessibility, WebKit2,
   "nicks" : [
      "msanchez"
   ]
},
```

Fig. 3. Selected entry from contributors.json showing a developer with multiple registered email accounts.

Email Domain	Affiliation	Commits
chromium.org	Unknown (primarily Google)	11124
apple.com	Apple	4943
webkit.org	Unknown	3904
gmail.com	Unknown	1879
intel.com	Intel	1607
google.com	Google	1576
rim.com	BlackBerry	1139
igalia.com	Igalia	1108
nokia.com	Nokia	872
samsung.com	Samsung	806

TABLE III
EMAIL DOMAINS USED BY TOP WEBKIT CONTRIBUTORS IN 2012

of contributors who have contributed to WebKit using personal email accounts, alongside their actual company affiliations. We can expect this issue has led to serious inaccuracies in the reported collaboration networks.

This substantial source of error is neither mentioned nor accounted for in [1]; all contributors in Table II were therefore miscategorized as unaffiliated. However, the authors clearly recognized this issue, as it has been accounted for in their more recent work covering OpenStack [9] by cross-referencing email addresses from git revision history with a database containing corporate affiliations maintained by the OpenStack Foundation. Unfortunately, no such effort was made for the WebKit data set.

The WebKit project maintains comparable data in the form of a JSON file[3] which could be used to associate personal email addresses with company email addresses, in order to reduce the number of corporate contributions incorrectly categorized as unaffiliated. However, it is not clear how to handle contributors whose affiliation changes over time. For example, Figure 3 shows a sample of data from this file for a developer who has contributed to WebKit while moving between three different companies. This same situation holds for many other developers (although usually a given contributor has only switched between two companies rather than three). Affiliation changes prevent naively using the JSON data to match personal emails to company emails, ensuring that personal emails will remain a source of error unless handled manually.

It must be noted that simply looking at the total number of commits coming from particular email domains reveals dramatic changes without any network construction or visualization required. Comparing the data in Table I to that in Table III, showing the same data from 2012—the last year before Google's Blink fork—clearly reveals the exodus of corporate contributors from the WebKit project in recent years.[4]

Table III also reveals that the project was previously dominated by contributors with chromium.org email domains. This domain is equivalent to webkit.org in that it can be used by contributors to the Chromium project regardless of corporate affiliation; however, most contributors with Chromium emails are actually Google employees. The high use of Chromium emails by Google employees appears to have led in [1] to a dramatic—by roughly an entire order of magnitude—undercount of Google's contributions to the WebKit project, as only contributors with google.com emails were considered to be Google employees. The vast majority of Google employees used chromium.org emails, and so were counted as unaffiliated developers. This demonstrates the importance of community-specific knowledge when engaging in studies of free software projects. Most WebKit developers were surely aware that the project was, at the time, dominated by Google, but the severe underrepresentation of Google nodes in [1]'s collaboration networks went unnoticed until now. We can safely assume that most of the gray nodes in Figure 1 should actually be orange.

As a final example of error caused by changing emails, consider that in 2013, Research In Motion Limited (RIM) changed its name to BlackBerry Limited. Around this time, its developers ceased to use rim.com emails in WebKit changelogs, and began using blackberry.com emails. Since this switch, its developers were classed as unaffiliated by

[3]Located in the WebKit source repository at Tools/Scripts/webkitpy/common/config/contributors.json

[4]It also reveals that Igalia has remained a core WebKit contributor after BlackBerry and Nokia left the project, despite [1]'s observation that Igalia previously represented the interests of these companies in the WebKit project.

[1], because the study considered only rim.com emails.[5] Fortunately, the BlackBerry case had very little impact on the networks because it occurred towards the end of the study period, and most blackberry.com commits were lost due to the issue described in Section III-C. The drastic undercounting of Google commits is clearly a more serious issue.

C. Missing Data

[1] incorrectly claims to have gathered its data from both WebKit's SVN revision history and from its changelog files. We must draw a distinction between changelog entries and SVN revision history. Changelog entries are inserted into changelog files that are committed into the SVN repository; they are completely separate from the SVN history. Each subproject within the WebKit project has its own set of changelog files used to record changes under the corresponding directory.

In fact, [1] processed only the changelog files. This was actually a good choice, as the changelog files are much more accurate than the SVN history, for two reasons. Firstly, it is easy for a contributor to change the email address entered into a changelog file, e.g. after a change in company affiliation. However, it is difficult to change the email address used to commit to SVN, as this requires requesting a change with the SVN administrator; accordingly, contributors are more likely to use older email addresses, lacking accurate company affiliation, in SVN revisions than in changelog files. Secondly, many SVN revisions are not directly committed by contributors, but rather are actually committed by the Commit Queue bot, which runs various tests before committing the revision, or, more rarely, by a completely different contributor. In this case, the proper contributor's name will appear in only the changelog file, and not the SVN data. Some developers are dramatically more likely to use the Commit Queue than others. Various other reviews of WebKit contribution history (e.g. [10]) that examine data from SVN (or `git-svn`) history rather than from changelog files are flawed for this reason. Fortunately, by relying on changelog files rather than SVN metadata, the authors of [1] avoid this problem.

Unfortunately, a serious error was made in processing the changelog data. WebKit has many different sets of changelog files, stored in various project subdirectories (JavaScriptCore, WebCore, WebKit, WebKit2, etc.), as well as toplevel changelogs stored in the root directory of the project. Regrettably, the authors of [1] were unaware of the changelogs in subdirectories and based their analysis only on the toplevel changelogs, which contain only changes that occurred in subdirectories that lack their own changelog files. In practice, this restricted the scope of the analysis to a very small minority of changes, primarily to build system files, manual tests, and the WebKit website. That is, *the reported collaboration networks do not reflect collaboration on any actual source code files.* All source code files are contained in subdirectories with their own changelog files, and therefore no source code files were actually considered in [1]'s analysis of collaboration on source code changes.[6]

We speculate that the analysis's focus on build system files likely exaggerates the effects of clustering in the network, as different companies used different build systems and thus were less likely to edit the build systems used by other companies, and that an analysis based on the correct data would display less of a clustering effect. Certainly, there would be dramatically more edges in the already-dense networks, because, as described in Section III-A, an edge exists between two developers if there exists any one file in WebKit that both developers have modified. Omitting all of the source code files from the analysis therefore dramatically reduces the likelihood of edges existing between nodes in the networks.

IV. Conclusion

We found that the original study was impacted by an unsuitable definition of collaboration used to build the collaboration networks, severe errors in counting contributor affiliation (including the classification of most Google employees as unaffiliated developers), and the omission of almost all the required data from the analysis, including all data on modifications to source code files. Nevertheless, the authors were able to derive many accurate conclusions about the WebKit project from their inaccurate collaboration networks. Such conclusions were clearly the work of confirmation bias, and illustrate the dangers of seeking to find particular meanings or explanations through visual inspection of collaboration networks. Researchers must work forwards from the collaboration networks to arrive at their conclusions, rather than backwards by attempting to match the networks to conclusions gained from prior ethnographic knowledge.

Acknowledgments

The author owes a debt of gratitude to Jose Teixeira and Tingting Lin of the University of Turku, Finland, the authors of [1], for publishing methodological details, source code, and changelog data for their study on the web [4]. Without the raw changelog data used for their analysis, the incomplete data collection issue would never have been discovered. Without their source code, this critique of their work may not have been attempted. This should be considered a testament to the importance of open publication of relevant code and data used in research. Many researchers today consider their source code and raw data to be either proprietary secrets to be guarded, or simply not important enough to merit publication. It is hoped that, in the future, such practices will come to be universally recognized as unethical, unscientific, and detrimental to the advance of knowledge. Teixeira and Lin are to be commended for their openness, which made this work possible.

[5]Note that when a contributor switches to a new email, she is henceforth represented by a different node in the collaboration network.

[6]This is not strictly true. In fact, a very small number of changes to source code files were considered in the analysis. However, this was only due to the occasional improperly-constructed changelog entries, where changes to files in subdirectories with their own changelogs were inappropriately recorded by developers in the toplevel project changelog. As such events were quite rare relative to the tremendous number of actual source code changes, this can be safely discounted.

The author additionally credits Carlos García Campos of Igalia for compiling the WebKit contributor affiliation data used in Table I and Table III.

REFERENCES

[1] J. Teixeira and T. Lin, "Collaboration in the open-source arena: The webkit case," in *Proceedings of the 52nd ACM conference on Computers and people research*. ACM, 2014, pp. 121–129.

[2] C. Lord, "The case for an embeddable gecko," http://web.archive.org/web/20160322223357/http://chrislord.net/index.php/2016/02/24/the-case-for-an-embeddable-gecko/, archived: 2016-02-24.

[3] A. Barth, "Blink: A rendering engine for the chromium project," http://web.archive.org/web/20160423072603/http://blog.chromium.org/2013/04/blink-rendering-engine-for-chromium.html, archived: 2013-04-23.

[4] J. Teixeira, T. Lin, and M. Krész, "Webkit social network analysis," http://web.archive.org/web/20140226221413/http://users.utu.fi/joante/WebKitSNA, archived: 2014-02-26.

[5] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Physical review E*, vol. 69, no. 6, p. 066133, 2004.

[6] "Cyber-ethnography," http://web.archive.org/web/20160229005748/https://en.wikipedia.org/wiki/Cyber-ethnography, archived: 2016-02-29.

[7] L. A. N. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley, "Classes of small-world networks," *Proceedings of the national academy of sciences*, vol. 97, no. 21, pp. 11 149–11 152, 2000.

[8] M. E. Newman, "Scientific collaboration networks. i. network construction and fundamental results," *Physical review E*, vol. 64, no. 1, p. 016131, 2001.

[9] J. Teixeira, G. Robles, and J. M. González-Barahona, "Lessons learned from applying social network analysis on an industrial free/libre/open source software ecosystem," *Journal of Internet Services and Applications*, vol. 6, no. 1, pp. 1–27, 2015.

[10] M. Olsson, "Browser engines 2015: Commit rates and active developer counts," http://web.archive.org/web/20160305090451/http://mo.github.io/2015/11/04/browser-engines-active-developers-and-commit-rates.html, archived: 2016-03-05.