On the use of Internet Relay Chat (IRC) meetings by developers of the GNOME GTK+ project

Emad Shihab, Zhen Ming Jiang and Ahmed E. Hassan
Software Analysis and Intelligence Lab (SAIL)
Queen's University
Kingston, ON, K7L 3N6, Canada
{emads, zmjiang, ahmed}@cs.queensu.ca

Abstract

Developers of open source projects are distributed across the world. They rely on email, mailing lists, instant messaging, IRC channels and more recently IRC meetings to communicate. Most of the studies thus far focus on the use of mailing lists by OSS developers, however, an increasing number of open source projects are using IRC meetings to hold developer meetings.

In this paper, we mine the `#gtk-devel` *IRC meeting channel and study the usage of the IRC meetings held by the GNOME GTK+ core developers and maintainers. We look at three different dimensions: the discussion volume of the meetings, the number of participants attending the meetings and the activity of these participants. Our findings show that IRC meetings are gaining popularity among open source developers and maintainers: the IRC meeting discussions are increasing in volume, have increasing attendance levels, and the participants actively contribute to the meetings. To the best of our knowledge, this is the first study on the use of developer IRC meetings by OSS developers.*

1 Introduction

Developers of Open Source Software (OSS) are distributed across the world. They communicate through mailing lists, emails, Internet Relay Chat (IRC) channels, IRC meetings or instant messaging (IM). Their discussions cover a wide range of topics such as design decisions, code quality, patch reviews and future project plans [9,12]. These discussions contain a wealth of information that can be mined to better understand the dynamics of OSS development.

Most of the work studying the communication of OSS developers has used mailing lists to conduct their studies (e.g. [3, 11]). IRC has been around since the late 1980s, however, its use by the OSS development community did not start until recently. For example, 8 years ago (year 2000), neither Apache nor Mozilla had official developer IRC channels, and today they both do [8].

There are two types of developer IRC channels: general developer IRC channels and developer IRC meeting channels. *General developer IRC channels* are common servers, open 24 hours a day, where developers can connect and discuss questions that pop up at the spur of the moment, informally and with no real agenda [7]. General developer IRC channels are very similar to instant messaging, except for the fact that messages, by default, are viewable by everyone logged into the channel. Several prior studies used data from general developer IRC channels to study the cultures and beliefs of OSS developers and the social networks in commercial software development [4–6].

On the other hand, *developer IRC meeting channels* are used by developers to hold focused group discussions in a short period of time (usually 1 hour). These meetings are generally used to discuss several maintenance and project related issues such as upcoming releases, major bugs or task assignments. Therefore, mining IRC meeting logs can provide us with valuable information that can be leveraged by researchers to better understand OSS development.

In this paper, we study the usage of IRC meeting channels by the GNOME GTK+ core developers and maintainers using three different metrics: the meeting discussion volume, the meeting attendance levels and the contribution level of meeting participants. We look to answer the following questions:

1. *Is the discussion volume of IRC meetings changing over time?*
2. *Do participants attend IRC meetings and does their attendance change over time?*
3. *How much do IRC meeting participants contribute?*

Our studies show that IRC meeting channels are gaining popularity among open source developers and maintainers.

Overview of Paper. Section 2 describes the IRC data. The framework used to mine and analyze the IRC logs is

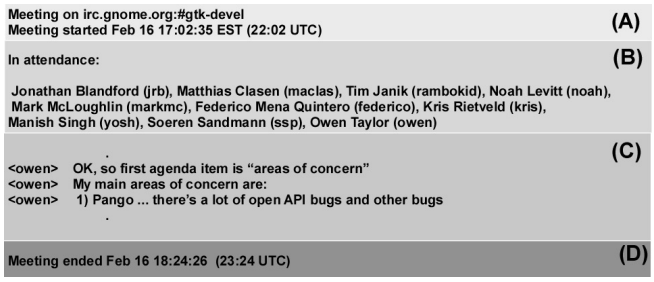

Figure 1: Sample IRC meeting log

detailed in Section 3. Our results are presented and interpreted in section 4. The paper is concluded in Section 5.

2 IRC Data

The meeting logs of the `#gtk-devel` IRC meeting channel are archived by the GTK+ project on their Meeting Space site [1]. A sample meeting log is shown in Figure 1. The first few lines of the meeting logs contain information about the meeting, such as the start time (denoted as A in Figure 1). Then, the list of attendants is noted (denoted as B). These lines are followed by the messages exchanged by the meeting participants (denoted as C). The meeting logs are concluded with a few lines that mention the end time of the meeting (denoted as D). It is important to note that not all of the meeting logs follow the standard format. In some cases, only the messages exchanged were made available in the archived logs.

Furthermore, there are different types of IRC message lines that one might encounter when mining meeting logs, depicted in Figure 2. In some cases, the month, date and time are included in the time stamp, while in others only the time is logged. In some cases, the time stamp is omitted altogether. In the next subsection, we outline the framework used to mine and store the IRC data.

3 The IRC Analyzer Framework

We built a framework that parses the meeting logs and stores the messages in a PostgreSQL database.

Step 1: Data Collection

IRC meetings are held by the GTK+ core team "as regularly as possible" to discuss various project related issues, i.e., bugs, release schedules, code quality [2]. However, the meetings are open to anyone interested in the GTK+ project.

Date	Name	Message
Jun 05 21:05:40	<name>	agenda for the meeting ...
	<name>	agenda for the meeting ...
[21:05:40]	<name>	agenda for the meeting ...
21:05	name	agenda for the meeting ...
21:05	<name>	agenda for the meeting ...

Figure 2: Different types of IRC messages

We obtained the meeting logs for the years 2004 till 2008 from the GTK+ Meeting Space site [1] and mined a total of 105 meeting logs. The logs contained 17,217 message lines from 148 different participants.

Step 2: Data Parsing

Initially, we conducted an inspection of the meeting logs and identified 5 different types of IRC message lines (shown in Figure 2). Then, we built the IRC message parser to handle the different message types. Our framework uses regular expressions to handle the different message types, and was able to successfully parse all of the 105 meeting logs.

Step 3: Multiple Alias Resolution

Participants of meetings assign themselves nicknames when joining the IRC meeting or change their nicknames during the meeting. Therefore, there can be multiple nicknames (aliases) for the same person. This so called multiple alias problem is similar to the multiple alias problem observed in mailing lists [3].

For example, the participant `jrb` uses the aliases:

```
jrb
<jrb>
<jrb_>
<jrb_meet>
<jrb_sick>
```

Using name resolution heuristics we were able to find and resolve the majority of the aliases. However, manual inspection was needed to resolve some of the rare cases. Furthermore, it is worth noting here that the majority of the meeting participants use abbreviated names, therefore, methods such as the one proposed by Robles and Gonzalez-Barahona [13] need to be used to accurately identify the participant's real names. Such identification becomes extremely important when multiple data sources (i.e. source code repository, mailing lists and IRC meeting logs) are used in combination and one needs to be able to identify the same person in all sources.

Step 4: Data Storage

After parsing the message lines, we reconstructed the IRC messages in preparation for storage in the database. Each IRC message contains three properties: `date`, `name` and `message`. Then, the information was stored in a PostgreSQL database for further use.

The use of a database eased the exploration of the large data at hand since we could rapidly explore different questions and generate specialized views to answer these questions.

4 Results and Interpretations

In this section, we report our results and answer the questions posted earlier.

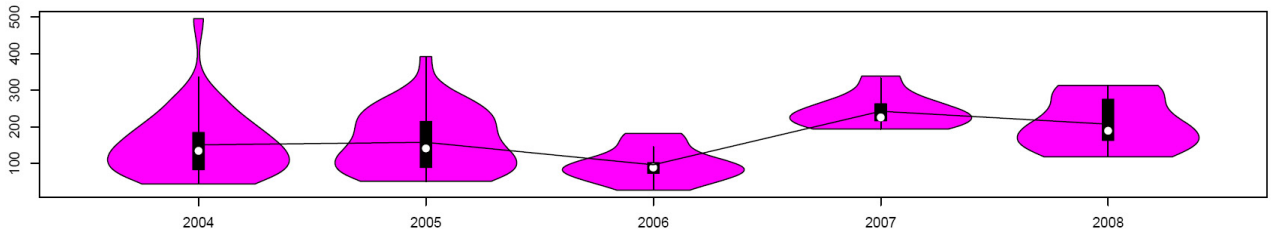

Figure 3: Number of message lines in IRC meetings

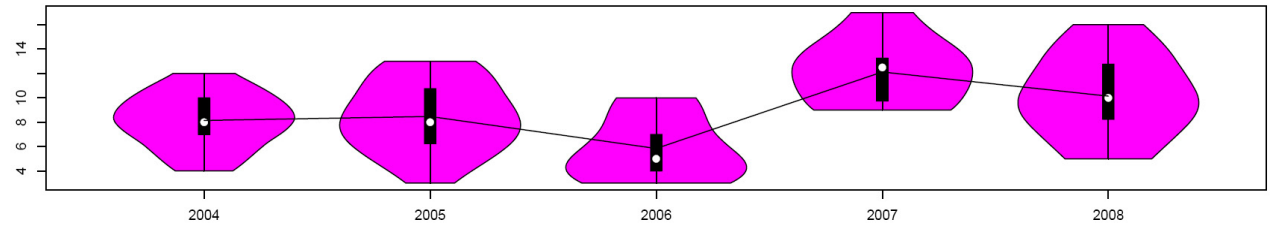

Figure 4: Number of participants attending IRC meetings

4.1 Discussion Volume

1) *Is the discussion volume of IRC meetings changing over time?*
As a first step, we wanted to study the change in discussion volume. The reason for this study is to see whether meetings are increasing in popularity over time.

We measured the number of message lines in each meeting and plot our findings using violin plots. Violin plots [10] are similar to box plots. The center of the plot shows the median. The top of the plot shows the maximum value and the bottom shows the minimum value. The first and third quartile are represented as the top and the bottom of the thick line in the center of the violin plot. The main advantage of violin plots, compared to a box plot is the fact that violin plots present the density. The wider the violin plot, the higher the density. In addition, we plot the moving average (denoted by the black line).

Considering the violin plot for 2004 in Figure 3 as an example, we can see that the median is approximately 150 (denoted by the white dot), the max is approximately 500 and the min is approximately 50 (represented by the top and bottom of the violin plot). The highest density occurs for value 100 (i.e. the violin is widest around 100) and the high values (between 400 and 500) have a small density (i.e. the pointy top of the violin plot around 500).

It can be observed from Figure 3 that a general trend showing an increase in discussion volume is observed. The increase in discussion volume may be due to two factors: 1) the fact that meeting attendants are discussing more or 2) the fact that meetings are becoming increasingly popular and more attendants are joining in the discussions. Also, the majority of the meetings are between 100 and 200 messages in length. However, there are a few meetings that are longer than usual (represented by the pointy tops of the violin plots, e.g. 2004). We hypothesize that these meetings were probably held just before major releases. Traditionally, meetings held before major releases are longer than usual because they cover many issues such as, open bugs, code freezes, feature inclusions and documentation updates. In the future, we plan to explore the reason for the increase in discussion volume and the relationship between discussion volume and release schedules.

4.2 Meeting Attendance

2) *Do participants attend IRC meetings and does their attendance change over time?*
Studying the attendance of participants provides insight about popularity and usefulness of the IRC meetings. Generally, low meeting attendance could indicate that participants are not seeing the usefulness of the meetings and therefore, they decide not to attend. At the same time, we do not expect full attendance by participants because meetings are held for 1 hour, once a week (Tuesdays at 20:00 UTC) and are mostly attended by members of the core team (made up of 10 members in the case of the GTK+ project). Some of the participants may not be able to attend due to time zone differences or scheduling conflicts.

Figure 4 shows the number of participants for each year. It is observed that on average, meetings are attended by 6 or more members. The number of attendants in recent years (2007 and 2008) is higher than the previous years, indicating that perhaps meetings are becoming more popular among developers of the project. We plan to study the popularity of meetings among OSS developers in more detail in our future work.

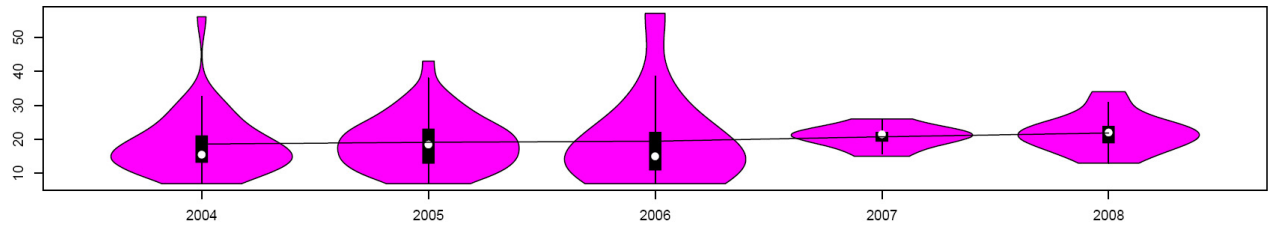

Figure 5: Number of IRC messages per participant per session

4.3 Participants' Contribution

3) *How much do IRC meeting participants contribute per session?*
As we have seen above, the meeting attendance is satisfactory and increasing over time. Now we would like to determine whether participants actively participate in the meetings or whether they attended just because they are obliged to. If we determine that participants actively participate in meetings, then we can say that participants see a benefit in these meetings and therefore, keep coming back and participating. If on the other hand we notice that participants are not actively participating, then perhaps they are not seeing the benefit of these meetings and studying them may not be so beneficial.

We measure the number of message lines per participant in a session and plot our findings in Figure 5. It is observed that the average number of messages per participant is between 15-20 message lines. Further, we observe that in the years prior to 2007, there exist a few participants who contribute more than the average (i.e. the pointy tops of the violin plots). Most likely, these above average participants are project leaders who are looking for updates from others. As for the recent years (2007 and 2008), we see a different type of violin plot. In those years, we observe that the majority of the participants have very similar contribution levels. This might suggest a change in leadership or a change in meeting style. We plan to explore this point further in our future work.

5 Conclusion and Future Work

In this paper, we mined and studied the IRC meeting logs of the `#gtk-devel` IRC meeting channel. We used the mined information to study the usage of the IRC meeting channel by project developers and maintainers, their attendance levels and their contributions. We found that: 1) the discussion volume is increasing over time, 2) IRC meetings have a positive level of attendance that is increasing over time and 3) participants are actively contributing in the IRC meetings.

In the future, we plan to further our study on the aforementioned points and study the IRC meeting contents to see whether we can leverage the information in IRC meeting logs to gain better insight on OSS development.

References

[1] Gtk mettings space. http://live.gnome.org/GTK+/Meetings.

[2] The gtk+ project. http://www.gtk.org/development.html.

[3] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks. In *MSR '06: Proceedings of the 2006 international workshop on Mining software repositories*, pages 137–143, New York, NY, USA, 2006. ACM.

[4] G. Breach. I'm not chatting, i'm innovating! locating lead users in open source software communities. http://www.business.uts.edu.au/management/workingpapers/files/Breach2008.pdf.

[5] M. Cataldo, P. A. Wagstrom, J. D. Herbsleb, and K. M. Carley. Identification of coordination requirements: implications for the design of collaboration and awareness tools. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, 2006.

[6] M. S. Elliot. The virtual organizational culture of a free software development community. In *Proceedings of the 3rd Workshop on Open Source Software Engineering*, 2003.

[7] D. M. German. The gnome project: a case study of open source, global software development. *Software Process: Improvement and Practice*, 8(4):201–215, September 2004.

[8] D. M. German, D. Cubranić, and M.-A. D. Storey. A framework for describing and understanding mining tools in software development. *SIGSOFT Softw. Eng. Notes*, 30(4):1–5, 2005.

[9] A. E. Hassan. The road ahead for mining software repositories. In *Proc. FoSM 2008. Frontiers of Software Maintenance*, pages 48–57, Sept. 28 2008–Oct. 4 2008.

[10] J. L. Hintze and R. D. Nelson. Violin plots: A box plot-density trace synergism. *The American Statistician*, 52(2):181–184, 1998.

[11] D. Pattison, C. Bird, and P. Devanbu. Talk and work: a preliminary report. In *MSR '08: Proceedings of the 2008 international workshop on Mining software repositories*, pages 113–116, 2008.

[12] P. C. Rigby, D. M. German, and M.-A. Storey. Open source software peer review practices: A case study of the apache server. In *ICSE '08: Proceedings of the 30th international conference on Software engineering*, pages 541–550, New York, NY, USA, 2008. ACM.

[13] 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.