On the Central Role of Mailing Lists in Open Source Projects: An Exploratory Study

Emad Shihab Nicolas Bettenburg Bram Adams Ahmed E. Hassan

Software Analysis and Intelligence Lab (SAIL)
Queen's University
Kingston, K7L 3N6, Canada
{*nicbet, emads, bram, ahmed*}*@cs.queensu.ca*

Abstract. Mailing lists provide a rich set of data that can be used to improve and enhance our understanding of software processes and practices. This information allows us to study development characteristics like team structure, activity, and social interaction. In this paper, we perform an exploratory study on the GNOME project and recover operational knowledge from mailing list discussions. Our findings indicate that mailing list activity is driven by a dominant group of participants, that it is greatly connected to development activity, yet influenced by external factors like market competition. Our results provide a broad picture of the central role played by mailing lists in open source projects.

1 Introduction

Most open source developers communicate through mailing lists. This style of communication makes mailing lists a rich source of information which researchers can use to understand software processes and improve development practices. Mailing lists have been used to infer social structure [4,5,11], identify architectural changes [1], and most recently to study the code review process [3, 14, 18].

However, understanding the generality of the results derived from mailing lists requires that we first understand how mailing lists are used in practice and the impact of their usage patterns on the information in the lists. For example, previous studies (e.g. [4, 5]) studied the social structure of developers using mailing lists, however, does this social structure change over time? How fast does the structure change?.

The central role played by mailing lists is depicted in Figure 1. Developers use mailing lists to discuss a variety of issues and project decisions [1, 10]. Many of these issues and decisions are related to and affect the source code. These issues are often driven by external factors such as the introduction of new features in competing products.

In this paper, we perform an exploratory study on the role played by mailing lists. Performing an exploratory study on mailing lists provides a holistic view of their role. This holistic view enhances the understanding of the findings of in-depth studies, unveils details which may not be apparent through in-depth studies and helps identify interesting directions for future research.

To perform our study, we use the mailing lists from 22 GNOME projects. The study centers around the following aspects, shown in Figure 1, in an open source project:

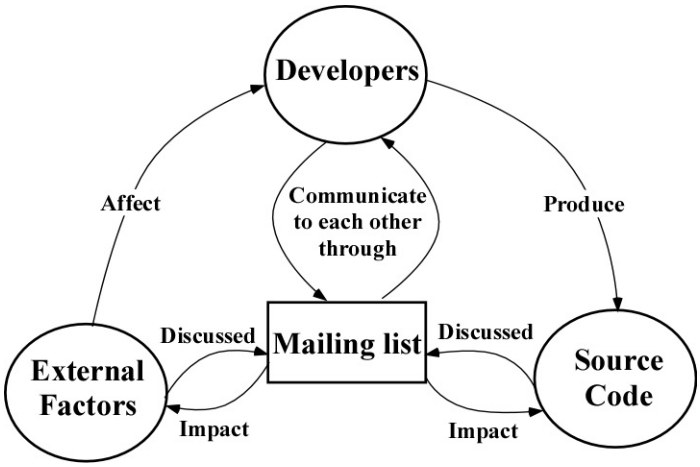

Fig. 1. The central role of Mailing lists in open source projects

		Number of				
Project name	**Start date**	**Messages**	**Participants**	**Age (months)**	**Threads**	**Application Domain**
Deskbar Applet	Oct-05	1,098	106	39	340	Search interface
Ekiga	Aug-06	5,389	690	29	1,200	Teleconferencing
Eog	Mar-01	458	106	93	233	Image viewer
Epiphany	Dec-02	5,735	905	73	1,608	Web browser
Evince	Jan-05	1,358	415	48	566	Document viewer
Evolution	Jan-00	53,927	6,026	96	15,718	Email client
Games	Feb-03	1,590	190	71	531	Computer games
Gdm	Mar-00	2,578	675	105	1,040	Display manager
Gedit	Apr-00	2,237	530	104	919	Text editor
Multimedia	Oct-00	1,646	273	98	507	Multimedia library
Network	Aug-03	673	105	65	267	Network tools
Power Manager	Jan-06	1,059	199	36	305	Power management
Themes	Jan-98	1,310	221	132	447	Window manager
Utils	Oct-04	358	106	51	279	Utility applications
Control Center	Dec-99	1,478	168	97	311	Configuration
Libsoup	May-06	83	24	32	41	HTTP library
Metacity	Sep-05	262	48	40	59	Window manager
Nautilus	Apr-00	22,488	2,384	105	5,582	File manager
Orca	Jan-06	11,930	516	36	3,598	Screen reader
Screensaver	Oct-05	139	25	39	30	Screensaver
Seahorse	Jun-07	252	34	19	116	Encryption management
System tools	Nov-99	1,832	327	98	792	System admin tools

Table 1. General overview of the GNOME mailing lists studied

– **Developers:** We characterized the communication style of mailing list's participants, i.e., the developers from the development mailing lists.

  We found that a small number of developers play a central role in driving the mailing list activity. We also found that these developers remain stable throughout the lifetime of a project.

– **Source code:** We explored the impact of mailing list activity on the source code activity, i.e., changes.

  We found that there is a high correlation between mailing list activity and source code activity.

– **External Factors:** We examined the effect of external factors, such as competing products on mailing list activity.

  We found that competing products shape and drive many of the discussions on mailing lists.

Overview of Paper. The rest of the paper is organized as follows: Section 2 discusses the motivation for using the GNOME project as a case study and presents statistics about the project. We present and analyze our findings in Section 3. The threats to validity are discussed in Section 4 and the related work is presented in Section 5. Section 6 concludes the paper.

2 GNOME as a Case Study

In this section, we detail the case study project used in our study. The GNOME project is composed of approximately two million lines of code and has more than 500 different contributors from all over the world [9]. The GNOME project is composed of many small projects that cover a wide range of applications, e.g., email client, text editor, and file manager. The main source of communication for GNOME developers is the developer mailing list for each project. These projects vary in size, age, user and developer base. We expect these differences in size, age, and domain to have an impact on the mailing lists of these projects. Therefore, studying the mailing lists of the different projects can lead to interesting and generalizable findings and open new directions for future research.

Table 1 presents a general overview of the mailing lists used for this study. The `Project name` column lists the name of the GNOME module. The `Start date`, `Number of Messages`, `Number of Participants`, `Age` and `Number of Threads` columns list the month and year of the first commit to the project's trunk (derived by examining the source control repository for the project), the total number of messages, the number of participants, the age, and the number of threads of the GNOME projects, respectively. In addition, the `Application Domain` column lists the application of the project. All calculations are based on the participation from the start date listed till the end of 2008, inclusive.

3 Results and Analysis

We now study the three aspects outlined in Figure 1 using the GNOME mailing list data. Subsections 3.1 and 3.2 cover the developers aspect, subsection 3.3 covers the source code aspect and subsection 3.4 covers the external factors aspect. We start each subsection by presenting our motivation to explore the aspect. We then describe the approach that we used to perform our exploration. Finally, we present our results and outline our main findings.

Since most of the GNOME mailing lists have low activity, we will often use the Evolution and Nautilus projects to more closely explore many of our findings since the two projects account for more than 65% of the total messages. We highlight the results that generalize for the rest of the 20 projects, where applicable.

3.1 Communication Style in Mailing Lists

Is mailing list activity mostly driven by a few participants (a dominant group) or is the participation evenly distributed? Does the dominant group engage in discussions with others or is it mostly involved in internal discussions?

Motivation. The Pareto principle (also known as the 80-20 rule), which states that the majority of the effects come from a minority of the causes, has applications in many fields. For instance, research shows that 20% of the code contains 80% of the bugs [8]. We hypothesize that there exist a few key participants (who we call the *dominant group*) in mailing lists, that are responsible for most of the messages posted on the mailing list. Most likely, they are members who are very knowledgeable about the project and use their knowledge to support newcomers and casual participants (who we call the *casual group*). It is important for us to investigate whether these experts exist on mailing lists for two reasons: 1) one can address his/her questions directly to such experts to receive a more accurate and speedy response and 2) the discussions of these experts can be used for future reference by others who are less knowledgeable about the project.

In addition, if such a group exists, we would like to know if they actively engage in discussions with others who are outside of the dominant group. If in fact they do engage with others then we can safely assume that newcomers and less experienced developers will benefit from these experts. If we determine otherwise, i.e. that the dominant group is a closed group, then newcomers and other participants may be better off reading previous discussions and learning from them rather than attempting to establish direct contact with the dominant group members.

Approach. We measured the number of messages contributed by the top 10% most active participants, who we call the dominant group. We found evidence that in fact there does exist a dominant group for each of the 22 GNOME mailing lists. The dominant group contributes a large amount of the messages posted.

Then, we examined the active discussion threads and classified these active threads into threads with:

- **Dominant group members only:** A high number of such threads implies that the dominant group is a closed group that does not engage with others.

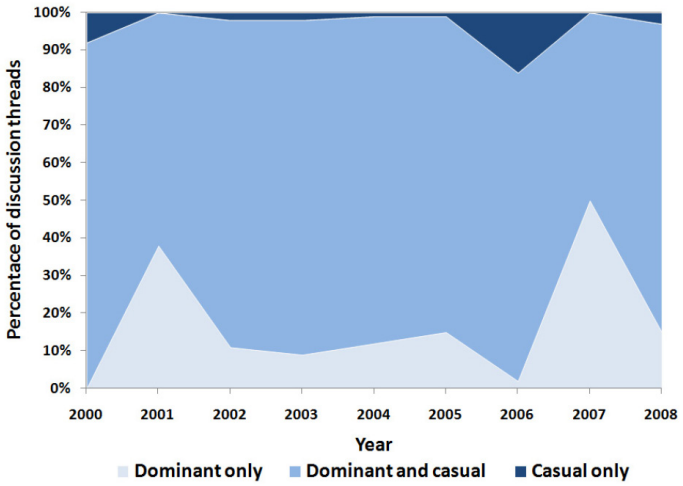

Fig. 2. Distribution of discussion types in the Evolution project

- **Dominant and casual members together:** A high number of such threads is a good indicator of a stimulating mailing list where expert and casual participants actively engaging in discussions.
- **Casual group members only:** A high number of this type of discussion would indicate that the casual members are not integrated into the mailing list.

Results. In addition to finding out that there exists a dominant group in each mailing list, we quantified their contribution. We found that on average the dominant group accounts for approximately 60% of the messages. *This finding is consistent across all of the 22 GNOME projects*. We did not observe a consistent finding when we considered the top 20% of the participants (i.e. we did not find evidence of the Pareto principle).

We plot the number of threads for the Evolution and Nautilus projects in Figures 2 and 3, respectively. In both projects, we found that the majority of the active discussions involve dominant and casual group members. On average, in 82% of the discussions dominant and casual group members were present. In 16% of the discussions, dominant group members were discussing exclusively and in the remaining 2% of the discussions the casual members discussing exclusively. We believe that it is a sign of a productive mailing list when the two groups actively engage in discussions, with the dominant group members most likely playing a supporting role for the casual group members.

However, in some cases a high percentage in discussions that involve dominant and casual members may not be desired. For example, some dominant group members may be overwhelmed by a high number of questions from casual members (since casual members may make unreasonable requests from more knowledgeable dominant group members). Whether a high number of discussions between casual and dominant group members is indicative of a productive mailing list depends on the product domain and the mailing list's members' knowledge.

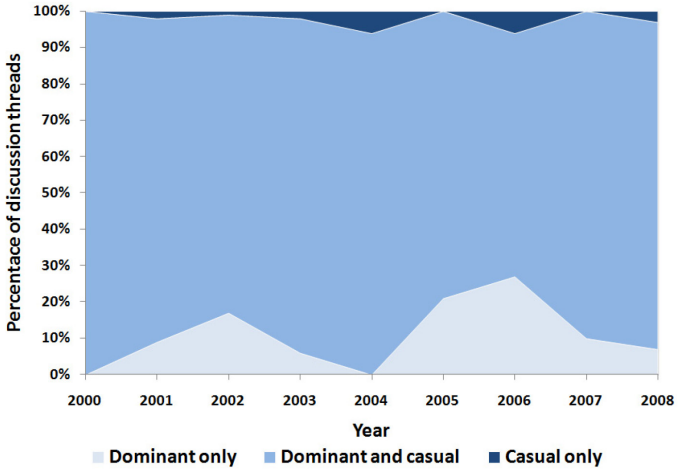

Fig. 3. Distribution of discussion types in the Nautilus project

> 10% *of mailing list participants (the dominant group) contribute* 60% *of the messages in a mailing list. The dominant group is very active and is engaging with outside-members, i.e. casual members.*

3.2 Stability of Mailing List Participants

Do dominant group members change over time? If so, how much are they changing by? How is their stability compared to rest of the mailing list participants?

Motivation. As we have seen in the previous subsection, the dominant group plays an important role in the mailing list. They contribute the majority of messages posted and are involved in approximately 96% of active discussions. For this reason, it is quite important that dominant group members do not change frequently. We study the stability of the dominant group. In particular, we measure the variation in the dominant group over time. A relatively stable dominant group (i.e. one that does not change frequently) is desirable because it means that dominant group members spend enough time in the project and achieve a higher level of expertise to better support casual group members.

Approach. To measure the stability of members in the dominant group, we performed two studies:

- **Dominant group change over time:** We measured the change between two consecutive years. This gives us a measure of how much a dominant group changes by from one year to the next.
- **Dominant group change compared to casual group change:** We measured the change of the casual group for two consecutive years and compared it to the change in the dominant group.

	Evolution		Nautilus	
Year	**Dominant**	**Casual**	**Dominant**	**Casual**
2000 - 01	0.68	0.11	0.73	0.20
2001 - 02	0.74	0.11	0.55	0.20
2002 - 03	0.63	0.16	0.40	0.21
2003 - 04	0.74	0.16	0.85	0.23
2004 - 05	0.84	0.16	0.76	0.24
2005 - 06	0.70	0.19	0.95	0.24
2006 - 07	0.35	0.17	0.88	0.19
2007 - 08	0.80	0.15	0.77	0.16
Average	**0.69**	**0.15**	**0.73**	**0.21**

Table 2. Cosine distance of dominant and casual groups of the Evolution and Nautilus projects

We used the Cosine Distance (CD) similarity metric to measure the similarity between the groups in two consecutive years. The CD metric outperforms other simple measures such as intersection or proportion which only measure the existence of a participant but not their level of contribution. The CD similarity is defined as:

$$CD(P,Q) = \frac{\sum_x P(X)Q(X)}{\sqrt{\sum_x P(X)^2}\sqrt{\sum_x Q(X)^2}}, \quad (1)$$

where $P(X)$ and $Q(X)$ represent the two input distributions to be compared. A value of 0 for the CD metric means that the group has changed drastically across two years with no members in common. A value of 1 for the CD metric indicates that the group is the exact same (i.e. is it a very stable group).

The Cosine Distance metric takes as input two participation distributions – one for each of the years under study. Each distribution has the contribution of each of the participants for that year. So when comparing the dominant group for the year 2000 and year 2001, the 2000 and 2001 participation distribution for the dominant group is used. One major challenge we faced when conducting this study was the use of multiple aliases by developers [4]. We used heuristics based on regular expressions to address this challenge as detailed in our previous work [2].

Results. The calculated CD values for the Evolution and Nautilus projects are shown in Table 2. It is observed that the dominant group is more stable than the casual group. On average, the dominant group is 3 times more stable than the casual group. *These two findings are observed across all of the 22 GNOME projects*. The same stability of social structures were also observed with the FLOSS projects [22]. This is a positive sign about the health of the dominant groups of many of these projects. Dominant group members, who are critically important to the mailing list of the project are stable enough to pass their knowledge to newcomers and casual group members.

	Type of change		
Project	**Add**	**Remove**	**Modify**
Evolution	**0.83**	0.60	0.61
Nautilus	0.32	0.53	**0.85**

Table 3. Correlation between the number of messages per year and the type of source code change

The participants in the dominant group are very stable over time. On average, they are about 3 times as stable as casual participants.

3.3 Source Code Activity and Mailing List Activity

Can mailing list activity be used to infer information about source code activity (amount of work done on the source code)?

Motivation. Since mailing lists are the main source for developer communication [10], we expect that mailing lists contain useful information about the source code of a project. We want to explore if we can infer the types of source code changes and the level of activity done on the source code through the mailing list activity. Because developers often use the mailing list to discuss their source code changes and get assistance or feedback on these changes [14], we hypothesize that there will be high correlation between the mailing list activity and the code activity. Or in other words, the more work done on the source code, the more it will be discussed on the mailing list and vice-versa.

Approach. We mined the SVN source control repository and extracted the number of lines added, removed and modified per year for each project. We defined a Code Activity (CA) metric, defined as:

$$CA(Y) = A_Y + R_Y + M_Y, \tag{2}$$

where A_Y, R_Y and M_Y refers to the number of lines of source code added, removed and modified in year Y, respectively. We used this metric and measured the correlation between it and the mailing list activity, i.e., the number of messages per year. Furthermore, we examine the correlation between the number of messages and the type of the performed change (add, delete, modify).

Results. The number of messages per year and the Code Activity for the Evolution and Nautilus projects are plotted in Figures 4 and 5, respectively. It can be observed that there is a high correlation between the number of messages on the mailing list and the Code Activity metric. This finding shows that developers do rely heavily on the mailing list to discuss source code changes. As for the correlation between the level of mailing list activity and the type of change, we present the results in Table 3. We found that in the Evolution project, the highest correlation was between the number of

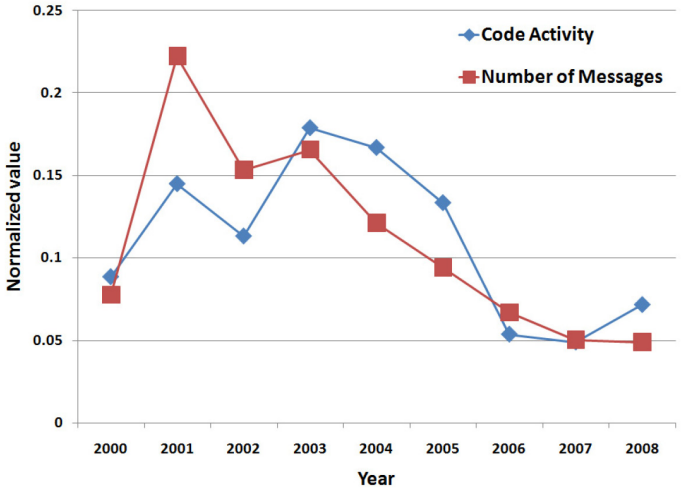

Fig. 4. Number of messages and Code Activity for the Evolution project

messages and the lines of code added ($\rho = 0.83$). On the other hand, in the case of the Nautilus project, we found that the highest correlation is between the number of messages and the lines of code modified ($\rho = 0.85$). It seems that in the Evolution project, participants are discussing code additions more than they are discussing code removal or modifications, while for the Nautilus project, code modifications are being discussed more than code additions and removals. We believe that further investigation is needed here to better understand the rationale for this discrepancy between both projects and whether it indicates different development and communication styles.

To verify, we measured the occurrence of terms that indicate code additions and code modifications in the mailing lists of the two projects. Since most commonly, code additions involve the introduction of new features, we classified the terms "new features" and "feature request" as indicators of code additions. Code modifications are usually carried out to fix bugs which are found during the testing phase and applied via patches. For this reason, we associate the terms "bug", "patch", "testing", and "maintain" to code modifications. We observed that in the Evolution mailing list, the terms associated with the addition of new features were mentioned in 57% more messages than on the Nautilus mailing list. On the other hand, the terms associated with code modifications were mentioned in 75% more messages in the Nautilus mailing list compared to the Evolution mailing list. The findings are consistent with our correlation results shown in Table 3.

Mailing list activity is closely related to source code activity. In addition, mailing list discussions are good indicators of the types of source code changes being carried out on the project

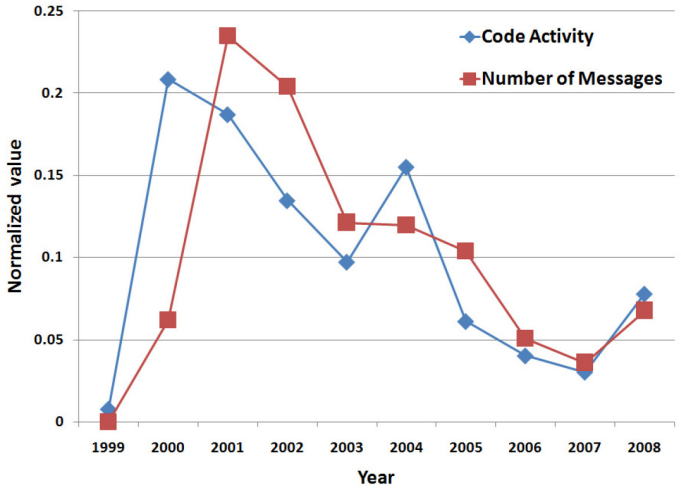

Fig. 5. Number of messages and Code Activity for the Nautilus project

3.4 Effect of External Factors on Mailing List Activity

Can we observe the effect of external factors on mailing list activity?

Motivation. One of the benefits of studying mailing lists is that they can provide us with knowledge about issues that indirectly affect a project, i.e., external factors. Market competition and management changes are examples of external factors. Such knowledge about external factors is often hard to uncover as it is not recorded in the source code or documentation. However, this knowledge is very important since it helps explain certain observed behaviors, such as an increase in bugs or the lack of interest in a project (and maybe its eventual death). We attempt to observe the effect of external factors on mailing list activity.

Approach. Due to space limitation, we perform the study of external factors on the Evolution project only. However, we note that our approach can be applied to any other project. We study the mailing list activity trend and perform two types of analysis: quantitative and qualitative analysis. In the quantitative analysis study, we treat the bodies of all email messages as a bag-of-words and compare the occurrence of the names of competing mail clients ("gmail", "outlook", and "thunderbird") to the occurrence of the terms: "evolution" and "evo" (a short hand form often used to refer to the evolution project). A rise in the number of times a term occurs indicates that it is being discussed more, hence it has a greater impact. In the qualitative study, we read through several email postings to better understand and clarify our quantitative findings.

Results.

Quantitative analysis: Looking at Figure 6, we observe that the activity on the Evolution mailing list is increasing from 2000 to 2001. This increase can be attributed to the creation of Ximian at the end of 1999, which was created to continue the development

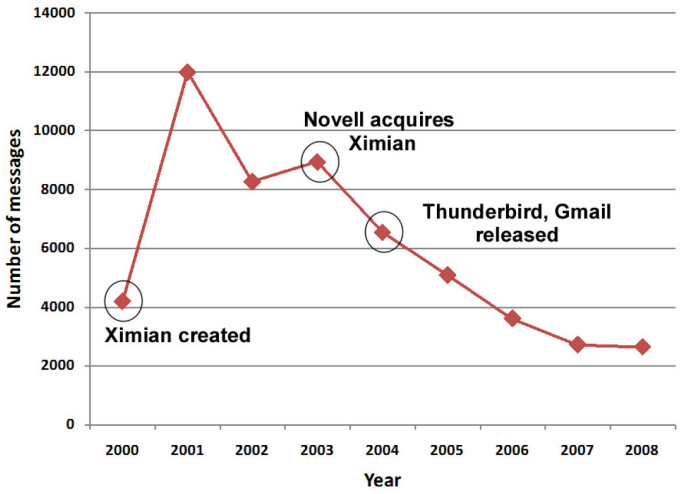

Fig. 6. Messages per year on the Evolution mailing list

of the Evolution project [9]. This acquisition increased the attention and support for the Evolution project, hence the continuing increase in mailing list activity.

Then, from the year 2001 on, we observe a steady decline in mailing list activity (except for a small increase in activity in the year 2003). Market competition, along with organizational changes may have caused this decline. The results of the quantitative study (which measures the frequency of occurrence of terms in the message bodies per year) are shown in Figure 7. We observe a steady decrease in the use of the terms "evolution" and "evo", suggesting that the Evolution project is being discussed less frequently. At the same time, there is a steady increase in the number of times its market competitors "gmail", "outlook" and "thunderbird" are being mentioned.

Qualitative analysis: We read through several mailing list posting to better understand our aforementioned quantitative findings. The following quotations are excerpts from discussions that took place when a declining level of activity was observed:

> "*...Furthermore, I can't find where in the Tools menu to change this: the option is no longer present on any of the dialog boxes. Which is why I'm sending this with Thunderbird...*"
> "*...Unless Ximian implements some features that aren't important to Ximian but are important to its users, evo will be relegated to "toy" status. I'm currently struggling to remain with my current distro of SuSE+Ximian in my business, but the lack of meaningful support in both components is forcing my hand to look around for another solution...*"

We believe that these excerpts show that the Evolution mail client was and is losing market share due to competition from other competing mail clients, such as Thunderbird, with many of the postings pointing people to competing products.

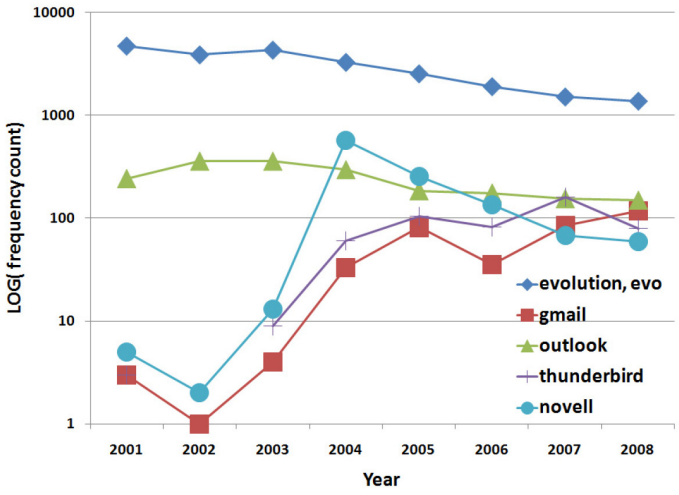

Fig. 7. Frequency of terms in the Evolution mailing list

As for the spike in activity on the Evolution mailing list in the year 2003, we believe this can be attributed to Novell's acquisition of Ximian in late 2003 [9]. We counted the occurrence of the term "novell" in the mailing list and found that the number of times the term "novell" was mentioned on the Evolution mailing list spiked from 13 in 2003 to 574 in 2004 (as depicted in Figure 7). This spike is most likely due to hype surrounding Novell's acquisition, which quickly dies off in the coming years.

This study on external factors suggests that mailing lists can be leveraged to study the effect of external factors on a project. Furthermore, such information can be used to explain design decisions that happened in the past.

External factors affect mailing list activity.

4 Threats to Validity

In our stability analysis, we used the names of developers as identifiers. Although we used heuristics to resolve multiple aliases [2] (i.e. participants who use multiple email address and names), we were not able to deal with some rare cases. Additionally, in our study we assume that all mailing list participants are developers. This assumption is true for the vast majority of the cases (especially since we are considering developer mailing lists), but in some cases, it is possible that a participant on the developer mailing list is not engaged in any developmental effort.

In our studies on source code activity and external factors, we measure the frequency of key terms that we associate with specific topics (i.e. the term "maintain" with the topic maintenance). Although our list is not exhaustive and does not contain all the terms that may be associated with the respective topic, we believe that the terms used in

our study are the most common and cover the majority of the terms that would be used to refer to the topic.

Finally, our findings may not generalize to all open source projects.

5 Related Work

Previous work used mailing lists to study the social structure of developers. Bird *et al.* [4, 5] used mailing lists to study the social networks created by developers and non-developers. In their follow-on work [7], they extracted the sub-community structure from these social network and studied their evolution over time. Ogawa *et al.* [12] used Sankey diagrams to visualize evolving networks in mailing lists and concluded that social behavior can be related to events in a project's development.

In addition, several studies used mailing lists to study developer morale, work times and the code review process. Rigby and Hassan [15] performed a psychometric study on the Apache httpd mailing list to identify the personality types of open-source software developers and gain insight on the level of optimism in pre- and post release phases. Tsunoda *et al.* [17] used mailing lists to analyze developer work times and found that the ratio of committer messages sent during overtime periods is increasing every year. Weissgerber, Neu and Diehl [18] used mailing lists to study the likelihood of a patch getting accepted.

Furthermore, other studies used mailing lists to study developer coordination, motivation and knowledge sharing. Yamauchi *et al.* [19] studied the coordination mechanisms used by OSS developers to achieve smooth coordination. They found that spontaneous work coordinated afterward is effective, rational organizational culture helps achieve agreement among OSS members and communications media, such as CVS and mailing lists, moderately support spontaneous work. Lakhani and von Hippel [21] used mailing lists to study the motivating factors of OSS participants to perform mundane tasks. They found that direct learning benefits is one of the main motivators for these participants to conduct such tasks. Sowe *et al.* [20] studied knowledge sharing between developers in mailing lists. They found that developers share knowledge a lot.

Other work combined the information extracted from mailing lists with information from other repositories (e.g. the source code repository). Robles and Gonzalez-Barahona [16] used information from multiple historical archives to assist in accurately identifying actors. Baysal and Malton [1] used the similarity between mailing list and source code archives to identify architectural changes. Bird *et al.* [6] combined the use of mailing lists and the source code repository to study the time it takes for developers to be invited into the core group of a project.

Our work recognizes the central role played by mailing lists and, to the best of our knowledge, is the first to perform an exploratory study using a large number of mailing lists. The study on the communication style of participants and their stability is novel and complements previous work. For example, previous work on social network analysis, developer morale, work times and evolution could have treated dominant and casual group differently and put more emphasis on the dominant group findings. Doing so would enhance the impact of their findings and provide a better understanding of the phenomena being observed. The findings from our source code activity and ex-

ternal factors studies can assist researchers who use mailing lists in combination with source code repositories (e.g. [1, 13]) better understand the relationship between the two. Further, taking into account the effect of external factors may help explain some unexpected observations.

6 Conclusions

In this paper, the central role of mailing lists was studied through an exploratory study. The study centered around three aspects: developers, source code and external factors.

Our findings indicate that a small number of participants (dominant group) account for the majority of the messages posted on mailing lists. The dominant group is very active and engaging with others and its composition is very stable (3 times more stable than casual members). In addition, we found that mailing list activity is closely related to source code activity and mailing list discussions are good indicators of the types of source code changes being carried out on the project. Lastly, we showed that external factors affect mailing list activity.

References

1. O. Baysal and A. J. Malton. Correlating social interactions to release history during software evolution. In *MSR '07: Proceedings of the Fourth International Workshop on Mining Software Repositories*, page 7, 2007.
2. N. Bettenburg, E. Shihab, and A. E. Hassan. An empirical study on the risks of using of off-the-shelf techniques to process mailing list data. In *ICSM'09: Proceedings of the 25th International Conference on Software Maintainance*, 2009.
3. C. Bird, A. Gourley, and P. Devanbu. Detecting patch submission and acceptance in oss projects. In *MSR '07: Proceedings of the Fourth International Workshop on Mining Software Repositories*, 2007.
4. 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, 2006.
5. C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks in postgres. In *MSR '06: Proceedings of the 2006 international workshop on Mining software repositories*, pages 185–186. ACM, 2006.
6. C. Bird, A. Gourley, P. Devanbu, A. Swaminathan, and G. Hsu. Open borders? immigration in open source projects. In *MSR '07: Proceedings of the Fourth International Workshop on Mining Software Repositories*, page 6, 2007.
7. C. Bird, D. Pattison, R. D'Souza, V. Folkiv, and P. Devanbu. Latent Social Structure in Open Source Projects. In *FSE '08: Proceedings of the 2008 ACM SIGSOFT symposium on the Foundations of Software Engineering*, pages 24–35, 2008.
8. B. Boehm and V. R. Basili. Software defect reduction top 10 list. *Computer*, 34(1):135–137, 2001.
9. 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.
10. D. M. German. Using software trails to reconstruct the evolution of software: Research articles. *J. Softw. Maint. Evol.*, 16(6):367–384, 2004.

11. L. Hossain, A. Wu, and K. K. S. Chung. Actor centrality correlates to project based coordination. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 363–372, 2006.
12. M. Ogawa, K.-L. Ma, C. Bird, P. Devanbu, and A. Gourley. Visualizing social interaction in open source software projects. *Asia-Pacific Symposium on Visualization*, 0:25–32, 2007.
13. 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.
14. 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, 2008.
15. P. C. Rigby and A. E. Hassan. What Can OSS Mailing Lists Tell Us? A Preliminary Psychometric Text Analysis of the Apache Developer Mailing List. In *MSR '07: Proceedings of the Fourth International Workshop on Mining Software Repositories*, page 23, 2007.
16. 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.
17. M. Tsunoda, A. Monden, T. Kakimoto, Y. Kamei, and K.-i. Matsumoto. Analyzing oss developers' working time using mailing lists archives. In *MSR '06: Proceedings of the 2006 international workshop on Mining software repositories*, pages 181–182, 2006.
18. P. Weissgerber, D. Neu, and S. Diehl. Small patches get in! In *MSR '08: Proceedings of the 2008 international working conference on Mining software repositories*, pages 67–76, 2008.
19. Y. Yamauchi, M. Yokozawa, T. Shinohara, and T. Ishida. Collaboration with lean media: how open-source software succeeds. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 329–338, 2000.
20. S. K. Sowe, I. Stamelos, and L. Angelis. Understanding knowledge sharing activities in free/open source software projects: An empirical study. *J. Syst. Softw.*, 81(3):431–446, 2008.
21. K. R. Lakhani, E. von Hippel, and K. R. Lakhani. How open source software works: Free user-to-user assistance. *Research Policy*, 32:923–943, 2003.
22. J. Howison, K. Inoue, and K. Crowston. Social dynamics of free and open source team communications. In *Second Intl Conf on Open Source Systems*, pages 319–330, June 2006.