THE POSITION OF QUALITY ASSURANCE CONTRIBUTORS IN FREE/LIBRE OPEN SOURCE SOFTWARE COMMUNITIES

BARHAM, Adina

Doctoral Dissertation

Graduate School of Social Sciences, Hitotsubashi University

SD112005

The development of quality assurance (QA) practices by software companies and the growth of FLOSS are two major phenomena in the domain of software development over the last 30 years. As FLOSS products became more widely adopted, it was unavoidable that the two phenomena would cross paths. In recent years, more and more FLOSS communities have adopted formal QA procedures in their development process in an attempt to improve stability and usability. This study investigates the relationship between QA and FLOSS, and more specifically to investigate the impact of QA adoption on the structure of FLOSS communities.

Communities are the driving force behind FLOSS development. A change in the structure of communities can therefore be an important factor in a project's success or failure. Hence, monitoring the evolution of FLOSS communities is important for both practitioners and researchers. Academic studies of FLOSS communities have focused on a variety of topics ranging from community structure and communication patterns, career advancement and role migration, to developers' motivations and the role of voluntary contributions. However, studies of QA adoption by communities are lacking.

The stereotyped image of FLOSS communities is of a bunch of super hackers developing software for their own gratification. However, research shows that FLOSS communities are comprised of individuals with diverse skill sets who contribute for various reasons ranging from hobbyist interest to professional advancement. Individuals without advanced programming skills have hitherto been able to contribute to FLOSS projects by translating or writing documentation. The emergence of QA activities in FLOSS development may offer a new area in which such non-programmers can contribute. Therefore, it is important to analyse who is contributing and how they are communicating regarding QA in FLOSS communities.

This study offers the first systematic empirical research on the impact of QA adoption on FLOSS communities. With the growing popularity of FLOSS products and the diversification of its user base, communities have realised the need for a more rigorously controlled development process that can ensure higher quality. As a consequence, methodologies previously used exclusively in proprietary software development such as formal quality assurance processes are increasingly present in the FLOSS landscape. These transformations in the development process suggest that FLOSS is heading towards a hybrid model that integrates elements from both the open and proprietary paradigms. The question of how and how far FLOSS communities achieve quality standards has already attracted the attention of academics; they have conducted a variety of studies focusing on topics including the QA practices that are employed under the FLOSS model, comparisons of quality levels between proprietary

and open systems, sustainability, fault prediction, and the relationship between social network metrics and quality. Thus a new question arises with respect to whether the implementation of formal QA procedures is affecting the structure of FLOSS communities. However, little research has sought to position QA activities within project communities. One reason for this gap might be the novelty of dedicated QA teams within FLOSS projects. The main objective of this research is to start filling this gap by establishing the impact of formal QA practices adoption on the structure of FLOSS communities. This goal will be achieved by analysing communities that have implemented a formal QA step in their development process and investigating how this emergent layer in the community affects its structure.

This thesis seeks to answer two main research questions:

Q1: Is QA a separate layer in FLOSS communities?

FLOSS communities are usually described as following a concentrical layered model that resembles an "onion". In this FLOSS onion model, communities are formed of four main layers: passive users, active users, developers and core developers; passive and active users are together considered to be the periphery. Although this model fails to capture some complexities of FLOSS communities, the onion model is still regarded as generally valid. This research aims to investigate how QA contributors fit into this onion model. More specifically, it asks whether QA contributors form a separate layer or whether members who belong to other layers perform QA related activities. It would be interesting to note, if all projects taken into consideration share the same structure with respect to the QA contributors. Previous research on the Firefox project has shown that peripheral members perform some QA tasks, for example posting 20 to 25% of bugs on the issue tracker; this research will deepen our understanding of this question by establishing the extent of peripheral involvement for several other major FLOSS projects.

Q2: What are the communication patterns between QA members as well as with other project participants?

Previous research has shown that information access correlates with productivity and participants who have better access to information are able to contribute more efficiently. Hence it would be useful to know whether there are participants who control the information flow; such dependence on a few participants may be risky for a project's long-term sustainability. This project will analyse communications between QA members to identify the central figures, in other words members with high activity levels within the community, and observe their evolution over time within the project. This requires us to analyze communication across different channels, most notably e-mail lists and issue trackers.

Based on existing literature regarding QA, the following working definition of QA is used for the purposes of this thesis: Testing, contributing code to automated testing tools or any test related activity, triaging bugs or any activity performed on the projects issue tracker, participating on the QA dedicated communication channels. QA team members are considered members who perform any of the tasks described in this definition. The number of FLOSS projects that are implementing formal QA activities in

their development process is increasing. A preliminary assessment of the top 100 FLOSS projects listed on Ohloh shows that more than a quarter of these projects include some form of QA in their development.

The approach chosen for conducting this research is the case study approach. The research consists of two phases:

- **Phase I:** A FLOSS community that has implemented QA practices is analysed as a preliminary case study. The community analysed is Mozilla, which is a large and mature community with a long history in developing successful FLOSS products such as Firefox and Thunderbird. QA mailing list data and issue tracker data are retrieved, cleaned and analysed using first simple statistics and then social network analysis techniques. A number of hypotheses regarding the position of QA activities within FLOSS communities are formed based on the findings.

- **Phase II**: Four other case studies are carried out. The projects studied differ from Mozilla in size and history and are: Ubuntu, Plone, KDE, and LibreOffice. The hypotheses proposed in Phase I are examined and adjusted in the light of the findings for those case studies.

The findings of the first phase suggest that in the case of Mozilla a smaller percentage of peripheral members (occasional contributors) are active on the mailing list as opposed to the issue tracker where a larger number of members have only one non-repeated act of communication. Activity on the QA mailing lists seems to be independent from activity on the issue tracker presenting peaks that are not directly related to time progression. Furthermore, a small group of people seems to be highly active in comparison to the rest of the community on both issue tracker and mailing lists. Social network analysis of the Mozilla community shows a large group of people spanning both issue tracker and mailing lists. However, almost two thirds of the connections are created by single acts of communication from one member to another. Furthermore, the existence of a highly active small group of people is also supported by the social network analysis of the Mozilla community. As regards information flow within the Mozilla community, the risk of a small group of people brokering information seems to be very low. As far as the communication carried out only on the QA lists, the patterns seem to display similarities to the whole network communication but on a smaller scale, i.e. one group of people working together where a small subgroup is highly active. However, the analysis of the Mozilla community revealed that issue tracker data and QA mailing lists' data is insufficient to determine whether QA represents a separate layer in the community.

In the second phase of research, in addition to retrieving communication data conducted on issue trackers and QA mailings lists, all mailing lists associated with each project were retrieved as far as possible. Furthermore, a list of community members contributing code to the project was downloaded from Ohloh, a public directory of FLOSS and contributors. This list was used for both data cleaning and contributor layer identification. Thus, data triangulation is used in the sense that data associated with various venues is downloaded. The analysis of these four case studies included general statistics methods and social network analysis techniques applied in a similar manner as for the Mozilla preliminary case study. The findings seem to validate some of the

hypotheses proposed in the first phase of the research. For example, all four communities contained a large group containing most of the projects' participants that spanned mailing lists and issue trackers. Furthermore, all communities had a small number of participants with a higher than average activity that displayed strong connections among themselves and were connected to a large number of members. However, some case studies displayed particularities. For example, while in the Mozilla, Plone and KDE cases the QA contributors seemed to merge with other layers in the case of Ubuntu the QA group tended to be more separate. In other words, a smaller number of members contributing to QA activities in the latter two communities were contributing with other activities to the project while members contributing with QA activities in Mozilla, Plone and KDE seemed to bring a variety of contributions to the project by submitting code and participating to a large number of non-QA mailing lists. These findings may suggest that in some cases less technically knowledgeable individuals are finding a new way to contribute to FLOSS development aside from documentation and localisation. Further study can be conducted in this direction considering the targeted user base for these projects. Ubuntu and LibreOffice may display a more segregated and rigorous QA due to the fact that the intended end-users for these software products are not necessarily technically "savvy".

With respect to the first research question, the five communities studied have dedicated communication channels, wikis, and other resources for providing QA related information. However, only the Ubuntu and LibreOffice communities displayed a somewhat separate QA layer where a large percentage of its members do not appear to be contributing code or communicating on other mailing lists. In the other communities a much smaller percentage of users were performing exclusively QA related activities; instead, they had multiple roles within the project. However, it is possible that non-QA activities were performed in different time frames than the ones in which the contributors were part of the QA team. Further study is required to clarify this point.

With respect to the second research question, all communities' graphs displayed a large group of people spanning both mailing lists and issue tracker. Previous research has suggested that FLOSS networks contain a small number of individuals with significantly higher connections than the network's average (called hubs). Our case studies supported this argument, as we found a very small number of individuals with a high degree compared to the network's average. Similarly, in the QA teams studied a small number of vertices had a higher degree than average. Within these teams, participants did not direct all communication efforts to one or a few members who then conveyed that information to members of other groups. Instead, QA team members seemed to communicate not only among themselves but also directly with members of other groups. Furthermore, fewer than 1% of community members had a higher than average betweenness centrality value, and those values were not particularly high, which suggests that information flow in the networks was not particularly vulnerable to a small number of individuals leaving the community.

Considering the diversity of FLOSS projects, some might argue that this data sample is not sufficient for drawing conclusions applicable to all FLOSS communities. However, this thesis does not aim to create a theory or a fail proof method that can be applied to all FLOSS communities but to lay the foundation for future research.

The following section outlines the structure of this thesis as follows:

1. **Chapter 1: Introduction**. This chapter introduces the main goals of this thesis and is divided in four sub-chapters:

    1.1. **Significance and Contributions of this Thesis**. This sub-chapter states the significance and the main contributions of this thesis to the field.

    1.2. **Research Questions**. In this sub-chapter the main research questions are stated and explained.

    1.3. **Outline of the Research Project**. This sub-chapter summarises the research conducted in this thesis.

    1.4. **Thesis Structure**. This sub-chapter details the chapter and sub-chapter structure of this thesis.

2. **Chapter 2: Literature Review**. This chapter reviews the appropriate literature and is divided in four main sub-chapters:

    2.1. **Free/Libre Open Source Software**. This sub-chapter presents a short history of FLOSS and essential concepts characterising the FLOSS development paradigm.

    2.2. **Software Quality Assurance and FLOSS**. This sub-chapter presents a review of literature focusing on Software Quality Assurance from both the standpoint of proprietary as well as FLOSS development.

    2.3. **Communities and FLOSS Development**. This sub-chapter contains a review of literature focusing communities under the FLOSS development model including structure and dynamics.

    2.4. **Summary**. This sub-chapter summarises the current chapter.

3. **Chapter 3: Research Methodology**. This chapter describes the research methodology and the processes used for data retrieval and analysis in the following sub-chapters:

    3.1. **The Case Study Approach**. This sub-chapter describes the research approach in the context of research objectives. Justifications are made for the choice of the case study methodology with respect to potential criticism of said method.

    3.2. **Data**. This sub-chapter presents the datasets included in this research as well as data models, tools used for data retrieval, chosen venues and data time span.

    3.3. **Social Network Analysis**. This sub-chapter presents essential Social Network Analysis metrics and concepts as well as justifies the choice of said method in the context of this research.

3.4. **Summary**. This sub-chapter summarises the current chapter.

4. **Chapter 4: Preliminary Research and Pilot Case Study Research**. This chapter presents the first stage of this research and is organised in the following sub-chapters:

   4.1. **Working Definition of Quality Assurance**. This sub-chapter presents the working definition of quality assurance used in this thesis for identifying project participants contributing with QA activities.

   4.2. **Preliminary Study of QA Adoption in FLOSS Projects**. This sub-chapter presents a preliminary assessment of QA presence within popular FLOSS projects.

   4.3. **Pilot Case Study: Mozilla**. This sub-chapter presents the justifications for choosing the Mozilla community as a pilot case study and presents general statistics about the QA teams within this community as well as a detailed social network analysis.

   4.4. **Summary**. This sub-chapter summarises the current chapter.

5. **Chapter 5**: Case studies. This chapter presents the second phase of the research and is organised in the following sub-chapters:

   5.1. **Ubuntu**. This sub-chapter presents general statistics about the QA team as well as a detailed social network analysis of the Ubuntu community.

   5.2. **Plone**. This sub-chapter presents general statistics about the QA team as well as a detailed social network analysis of the Plone community.

   5.3. **KDE**. This sub-chapter presents general statistics about the QA team as well as a detailed social network analysis of the KDE community.

   5.4. **LibreOffice**. This sub-chapter presents general statistics about the QA team as well as a detailed social network analysis of the LibreOffice community.

6. **Chapter 6**: Conclusions and Discussion. This chapter presents the main conclusions of this research and is structured in the following sub-chapters:

   6.1. **Comparative Analysis**. This sub-chapter presents a comparative analysis of all the case studies in order to verify the hypothesis presented in Chapter 4.

   6.2. **Answers to the Research Questions**. This sub-chapter presents the general conclusions of this research, and answers to the research questions.

   6.3. **Discussion and Limitations**. This sub-chapters addresses research limitations,

potential threads to validity and proposes future research directions.

7. **Appendix A**: This appendix contains tables associated with the preliminary analysis of the top 100 projects on Ohloh.

8. **Appendix B**: This appendix contains tables that enumerate all the mailing lists associated to each community and which mailing lists were retrieved and stored for each case study.

9. **Appendix C**: This appendix contains tables that detail the activity of QA member participants on all communication venues taken into consideration for this study. In addition, this appendix contains tables that describe activity on a yearly basis for each of the case studies.