AN ABSTRACT OF THE THESIS OF

Scott T. King for the degree of Master of Science in Computer Science presented on April 3, 2009.

Title: Joining Open Source Software Communities: An Analysis of Newbies' First Interactions on Project Mailing Lists.

Abstract approved: _____

Carlos Jensen

Open source software has become a powerful force in the world of computing. While once confined to the domain of technical specialists, people of all types have begun to adopt this software – from the casual web-surfer who uses Firefox, to the professional web developer who codes in PHP or Python.

People interested in both seeking and receiving information related to an open source software project are often directed to its mailing lists. Therefore, many newcomers, or "newbies", will have their first interactions with the project community there. These newbies are a sustaining force for open source software projects, making it worthwhile to investigate how these interactions play out and affect the newbies' future participation.

To gain insight into the first experiences newbies have interacting with an open source software community, we conducted a study of eight mailing lists across four open source software projects: MediaWiki, GIMP, PostgreSQL, and Subversion. We analyzed the discussion threads initiated by newbies on those lists for information such as poster gender, nationality, politeness, helpfulness, and timeliness of response. Among the most interesting results, we found that newbies were generally treated very well, with nearly 80% receiving replies to their first post. We also found that receiving timely responses, especially within 48 hours of posting, had a positive correlation with that newbie continuing to participate on the mailing list over time. Somewhat surprisingly, we discovered that a newbie's level of courtesy did not have a significant effect on whether or not that newbie received a reply.

Joining Open Source Software Communities:
An Analysis of Newbies' First Interactions on Project Mailing Lists

by
Scott T. King

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented April 3, 2009
Commencement June 2009

Master of Science thesis of Scott T. King presented on April 3, 2009

APPROVED:

Major Professor, representing Electrical and Computer Engineering

Director of the School of Electrical Engineering and Computer Science

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Scott T. King, Author

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my appreciation to the people who made this thesis possible.

First and foremost, thank you to my advisor, Dr. Carlos Jensen, who guided me through the rocky-yet-rewarding path of graduate school. I appreciate his academic advice, as well as his patience and flexibility in scheduling meetings with me outside of my work hours as this thesis entered its final stages.

Thank you to my committee members, Dr. Timothy Budd, Paul Paulson, and Dr. Harry Yeh, for graciously volunteering to participate.

A big thanks goes to Eric Betts, Jill Cao, Eunyoung Chung, Heather Lonsdale, Yunrim Park, Jordan Strawn, and Koji Yatani. Without all of their hard work helping me to rate thousands of mailing list posts, this study could not have gotten off the ground.

Thank you to Dr. Tom Rothamel for his expert Python advice.

Last, but definitely not least, I would like to thank my family for their support and encouragement through school and life.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Page

LIST OF FIGURES

Figure Page

LIST OF TABLES

LIST OF TABLES (Continued)

Joining Open Source Software Communities: An Analysis of Newbies' First Interactions on Project Mailing Lists

1 Introduction

People joining an open source software project often start by downloading the code and documentation, following the community's discussion on its mailing lists, and posting questions and comments to it. A project's mailing list is one of the main avenues for people of all experience levels to communicate about virtually any aspect of the software. As such, newcomers' first interactions with the community will often take place on the list. This thesis is about how people new to an open source software project make their introductory posts to these lists, how they are received by the community, and the effect that has on their future participation.

Open source software is essentially software whose source code is freely available to distribute and modify. While originally taking the form of programs shared by universities and research labs over Usenet in the early days of computing, the ubiquity of the Internet in recent years has allowed open source software to reach larger audiences of developers and end-users across the globe.

There are many reasons why open source software is important. First of all, it is widely used for many application areas. A prime example of this is the Apache web server, the most popular web server for over a decade (Wheeler, 2007). Even better known is the open source operating system GNU/Linux (commonly referred to as just "Linux") that has become the operating system of choice for servers, as well as the famous "One Laptop Per Child" project. MediaWiki, the engine behind the immensely popular Wikipedia, is open source. Even outside the realm of servers and the Internet, programs like GIMP and Blender have gained large user communities of artists and 3D modelers. OpenOffice.org, an open source alternative to Microsoft's Office suite, had been downloaded nearly 100 million times from its official website as of 2007 (OpenOffice.org). With its adoption by many end-users, businesses, and governments, it is safe to say that open source software is here to stay.

Newcomers to an open source software project, often called "newbies", play an important role in the community. They make up a pool of potential future developers and contributors to the project, vital to its long-term survival and evolution. With the vast majority of contributors participating voluntarily, having no financial incentive or contract, there is less to prevent them from leaving the project than might be the case for commercial closed-source software developers

tied to a contract and an employer. Therefore, a continual influx of people who are willing to learn how to contribute to the community provide the seeds of growth and sustainability.

Mailing lists are a primary medium for communication and coordination within open source software projects. They enable mass communication since emails sent to the list go out to everyone subscribed to it. The Apache website sums up the importance of mailing lists quite well:

> *"Mailing lists are crucial to the operation of the Apache Software Foundation. The Foundation, as well as each of the Apache Projects, uses mailing lists to coordinate development of the software and administration of the organization. Mailing lists also serve as a primary support channel where users can help each other learn to use the software." (Apache Software Foundation, 2009)*

With people interested in soliciting help and project information often directed to mailing lists, many newbies will have their first interactions with the community there. Since newbies are a sustaining force for open source projects, it is worth investigating how these interactions play out and how these affect the newbies' participation.

This study examines the first posts newcomers to eight open source software mailing lists make. Mailing lists from the MediaWiki, GIMP, PostgreSQL, and Subversion projects were chosen for examination to get a good mixture of projects of differing prominence, years of activity, and availability of raw list archive data. In addition, these projects were selected for the breadth of their target audiences; with user bases ranging from non-technical end-users to system administrators. Each project had a dedicated user list and a developer list, which we examined for newbie postings, and analyzed for information such as poster gender, nationality, politeness, helpfulness, and timeliness of response.

The results of the data collection provided us with interesting findings and insights into the world of communication of open source software participants. Focusing on the quintessential first step in the joining process, the first interaction with the community, and the one most newbies drop out after, this study can help to shape how open source software communities can encourage newbies to continue participating.

This study investigated the following research questions:

- Which factors determine a newbie's success in joining the mailing list community?
- Do newbies generally receive timely replies to their first post?

- How does a newbie's level of courtesy affect the replies he/she receives?
- How does a newbie's gender affect the replies he/she receives?
- How does a newbie's nationality affect the replies he/she received?
- How does the treatment of newbies differ between user lists and developer lists?

Other related aspects of newbies' first interactions on the mailing lists were explored as information was gathered and analyzed.

The structure of this thesis document is as follows: First, in section 2, we will describe the relevant work in a number of areas related to this thesis, including the origins of open source software projects, their characteristics, how people join them, and the roles people play in the projects. Next, in section 3, we describe our data-collection and parsing approach. In section 4 we describe at a high level the results of this data collection, the amount of data, characteristics of the eight mailing lists, etc. Section 5 investigates the research questions, presenting the data, and section 6 discusses those results. Finally, the conclusion summarizes the main findings and opportunities for future study.

2 Related Work

2.1 Overview

2.1.1 What is Open Source Software?

"Open source doesn't just mean access to the source code."

So begins the Open Source Definition by the Open Source Initiative (OSI), the de-facto authority on what constitutes "open source" (2006). Although many people think of open source software as simply being no-cost software or software with published source code, the community sees it as much more. Gacek and Arief (2004) point out the main criteria in the official definition as:

- Ability to freely distribute the software
- Availability of the source code
- The right to create derived works

Falling under this definition and the phenomenon of developers also being users are the two main characteristics common to open source software projects (Gacek and Arief, 2004).

2.1.2 A Brief History of Free and Open Source Software

The basic concept of open source software did not emerge as a revolutionary response triggered by dissatisfaction with a predominant closed source model. Rather, the open source software mentality was the norm for software in its earliest days. Computers were primarily research tools used by universities and research labs, who commonly shared code with each other (Nuvolari, 2005). This changed in the early 1980s as computers became more ubiquitous in business and the home. Companies saw that there was money to be made in the commercialization of software, and many developers started writing proprietary code.

Seeing moral and technological conflicts in the decreasing openness of software, Richard Stallman started the GNU operating system project in 1983 (Nuvolari, 2005, Stallman, 2008). GNU, which stands for "GNU's Not Unix", was meant to be a free operating system without the restrictive licenses of commercial software. With the GNU project, Stallman launched the Free Software Movement and founded the Free Software Foundation in 1985. Free software in the GNU sense is often described as being "free as in free speech, not as in free beer" (Free Software Foundation, Inc., 2007). Basically, free software can be run by anyone for any purpose, modified, and redistributed. Although the difference between the FSF's "free software" and

"open source software" is negligible in practice, there is a notable ideological one. Stallman writes:

> *"Nearly all open source software is free software; the two terms describe almost the same category of software. But they stand for views based on fundamentally different values. Open source is a development methodology; free software is a social movement. For the free software movement, free software is an ethical imperative, because only free software respects the users' freedom. By contrast, the philosophy of open source considers issues in terms of how to make software "better"—in a practical sense only. It says that non-free software is a suboptimal solution. For the free software movement, however, non-free software is a social problem, and moving to free software is the solution." (2007b)*

Stallman and other free software advocates see software usage as an ethical issue of right versus wrong, calling proprietary software "the enemy" (Stallman, 2007b). Despite ideological differences, the free and open source communities work largely on the same projects. For instance, anything covered by the GPL license is both free and open source. The two are so similar in practice that free and open source software are often referred to as a single group, "F/OSS" (Free/Open Source Software), or "FLOSS" (Free/Libre/Open Source Software).

In 1991, Linus Torvalds started the now-famous Linux project, developing an operating system kernel released under the GPL. He got help from around the internet, and in 1994 Linux 1.0 was released (Nuvolari, 2005). The operating system as a whole that used the Linux kernel also used Stallman's non-kernel GNU software, which is why it is also known as "GNU/Linux" (Stallman, 2007a). Eric Raymond was amazed by what he saw take place with Linux's success. This prompted him to become an open source advocate, writing an essay about closed source versus open source software development, "The Cathedral and the Bazaar" (Raymond, 1999).

Netscape became interested in the merits of open source development and invited Raymond to help plan the source code release of their web browser. At this time, a corporation treading down the path of open source was a major event. Raymond and others in the Linux community realized the code release by Netscape was a prime opportunity to sell companies on the merits of open source development. They met to discuss this in February 1998 and came up with the term "open source" that they would use to promote this non-proprietary style of development. The idea was to not focus on moral arguments as free software advocates did, but rather on the business and pragmatic side of the development model. Stallman was displeased with the term, as were some other free software advocates, creating a minor rift between open source and free software

developers. However, the term and promotion of open source development, as it was now known, spread around the Internet, and the OSI was founded (Tiemann, 2006).

2.1.3 Why Open Source Matters

Open source software powers many of the computer systems people rely on every day both directly and indirectly. Leading software in categories such as web servers, operating systems, databases, and web browsers are open source (Ghosh et al, 2006). Furthermore, their adoption is not limited to mom-and-pop businesses and home users on a tight budget, but is driven by medium and large-sized companies.

The open source Apache web server holds about 50% of the server market share with Microsoft's Internet Information Services software fifteen percentage points behind (Netcraft, 2008). Firefox, the most widely-used open source web browser has almost 30% of the market share in Europe (Paul, 2008) and approximately 20% worldwide (Keizer, 2008) despite Microsoft's Internet Explorer coming pre-installed on most of the world's computers.

From a software development standpoint, studying open source can teach us many lessons. Yamauchi et al. (2000) point out that the open source development process can produce useful, reliable software, even without the in-person communication and spontaneous discussion crucial to the success of traditional closed-source teams. This could provide insight into practices that can make any distributed software development more successful.

2.2 How Open Source "Works"

2.2.1 Characteristics of OSS Projects

Not only do closed source and open source projects differ in their philosophies and licenses, but they also tend to differ in their development models. Eric Raymond, co-founder of the OSI, famously compared the distinction between classic closed source software and open source software development to cathedrals and bazaars:

> *"I believed that the most important software… needed to be built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time…. The Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches… out of which a coherent and stable system could seemingly emerge only by a succession of miracles." (Raymond, 1999)*

Such a system did emerge, and Raymond became a prominent open source advocate. One of the prime differences between the cathedral-style of closed source and the bazaar-style of open source is the latter's mantra "release early, release often" (Raymond, 1999). Open source's concept of treating a program's users as its co-developers can be a great way to debug and improve code, providing a large community of people from which to receive input. Closed source firms, on the other hand, tend to rely on a typically smaller quality assurance department to find bugs before releasing software. In order to attract more participants, Raymond says the program does not need to work incredibly well or have tons of documentation, but rather just needs the "plausible promise" that it can become great (1999). This is an example of how open source welcomes new developers, whereas closed source restricts their joining. Brooks (1975) is known for saying "adding manpower to a late software project makes it later" because of newcomers' ramp-up time and mentorship needed from developers already familiar with the system. Under an open source model, though, newcomers often learn the system at their own pace with minimal distraction to other developers, relying on the code and discussion archives. The model again is geared towards inclusion.

In Ko et al's (2007) observations of software developers at Microsoft they found that programmers acquired the necessary information to do their jobs by talking to coworkers in person. Open source projects, however, tend to be highly distributed, and in-person meetings are usually not an option. Therefore long-distance communication methods, such as mailing lists and online chat become very important, being the primary means for developers to get the information they need. In the Ko study, developers' top information need was "did I make any mistakes in my new code?" Such information was obtained by debugging, compiling, and using computer tools. Both closed source and open source developers have access to these kinds of tools. The second most common information need was "what have my coworkers been doing?" Most often Microsoft developers found this information through in-person communication and follow-up emails. Open source developers primarily rely on the internet channels such as mailing lists, email, and IRC to obtain their coordination information.

Finally, when considering open source software projects, there can be a tendency to think of large projects with hundreds of contributors. This is understandable given the attention such projects get in the media, like Linux and Firefox. However, it is important to keep in mind that large projects are in the minority in open source. Illustrating this, a study of the hundred most active mature projects on SourceForge in 2002 (Krishnamurthy, 2002) found that individuals, not

large communities develop most OSS projects. Projects had a median of four developers, but most had only one. (Krishnamurthy calls this low-participation development the "cave" model as opposed to the bazaar model.) Only 29 of the 100 projects had more than five developers. Not only did most projects have few developers, but they also generated little discussion. Ten of the projects had no online forum or mailing list, and only three had over 1000 forum messages. Therefore, "open source" doesn't automatically mean "big", "popular", or "distributed", but merely refers to the definition as in the previous section. Because of the more interesting community dynamics and greater influence larger projects have, like many other researchers, we focused on those for our study.

2.2.2 The Importance of Newcomers

Ye and Kishida (2003) assert that newcomers, even passive end-users, are vital to the success of an open source software project. Without them, the entire project could come to a stop when active contributors leave. The departure of important contributors is a real threat in OSS since there are rarely contracts or obligations to force work on the project. Krogh et al's study of Freenet (2003) supports the importance of newcomers, noting a high turnover rate in developers, as well as the joining and specialization of new people being a main issue of concern to the developers. Even end-users are important because they are a source for potentially future contributors, and even when they do not contribute, they "play a role similar to that of the audience in a theatrical performance who offers values, recognition, and applause to the efforts of the actors" (Ye and Kishida, 2003).

2.2.3 The Joining Process

2.2.3.1 Closed Source Model

While not much has been written about the specific joining processes of developers to a new commercial/closed source software (CSS) project, there are some studies of note. Sim and Holt (1998) studied four employees of a commercial software company joining a new project. They coined the term "software immigrants" to describe these people, as they were not traditional novices given their extensive computer knowledge, but rather dealing with a new project. The company assigned mentors to these immigrants, acting as their main source of information about the project. Spending many hours each day with their mentors, they would learn about naming

conventions, how to set up the system, use the tools, etc. The mentors also played key roles in helping the newcomers socially integrate with the development team, introducing them on breaks. After 2-3 weeks, the immigrants were receiving task assignments. They were simple at first, usually easy bug fixes and optimizations. However, the immigrants were able to do more complicated tasks after about four months and do them independently of their mentors.

Also studying a closed-source environment with mentors at HP Labs, Berlin (1992) found that the project newcomers' main strategy for familiarizing themselves with code and doing their work was to copy and experiment with existing code. When they asked their mentors questions, the mentors would often provide them not only with the basic answer, but also vital information related to system design, rationale, etc. Newcomer-mentor sessions were very interactive and in-person. Though each was assigned a mentor, the newcomers felt a reluctance to ask questions when stuck. They reported feeling a need to figure things out on their own whenever possible. This stood in contrast to the expert mentors who were much quicker to consult each other. Berlin notes that as one becomes more familiar with a project and gains knowledge to offer other experts in return, a developer becomes less reluctant to seek their assistance.

Pigoski and Looney (1993) write about their experiences at a US Navy software maintenance organization. Unlike the other two studies, mentors were absent. The maintenance organization had large software systems handed over to them from defense contractors, with no involvement in the development process, making them an entire team of software immigrants. Their strategy was to first understand the problem domain by reading any existing documentation, operating the system, and discussing the system with the original developers when possible. Their next step was to learn the system's structure and organization, getting a grasp on what source files and libraries there were, rebuilding the system, producing call trees, and analyzing the code structure. Next came examining code to determine what the software was doing, and once that was underway, making updates was possible.

2.2.3.2 Open Source Model

Rather than looking at the differences in the proximity of open source and closed source developers or the difference in code accessibility, an interesting way to look at the distinction between OSS and CSS is the "role transformation" (Ye and Kishida, 2003). According to Ye and Kishida, the fundamental difference between OSS and CSS is the "role transformation of people involved in a project", stemming from the fact that anyone can join. All users in open source are

potential developers, but in closed source, the developers and users are two clearly separated groups. Ye and Kishida identify eight main roles of an OSS community member:

- Passive User: simply uses the software
- Reader: uses the software and reads source code to understand how it works
- Bug Reporter: finds and reports bugs, but does not fix them
- Bug Fixer: fixes bugs, needs to understand part of the source code
- Peripheral Developer: contributes features to the software, but sporadically
- Active Developer: contributes new features and bug-fixes regularly
- Core Member: guides and coordinates development
- Project Leader: has main vision, directs project (sometimes later replaced with core members)

These roles and people's ability to transition between them is commonly referred to as the "onion model" in the open source literature (Crowston and Howison, 2005, Jensen and Scacchi, 2007). As one becomes more involved with the project, he or she moves to a more inner layer of the onion.

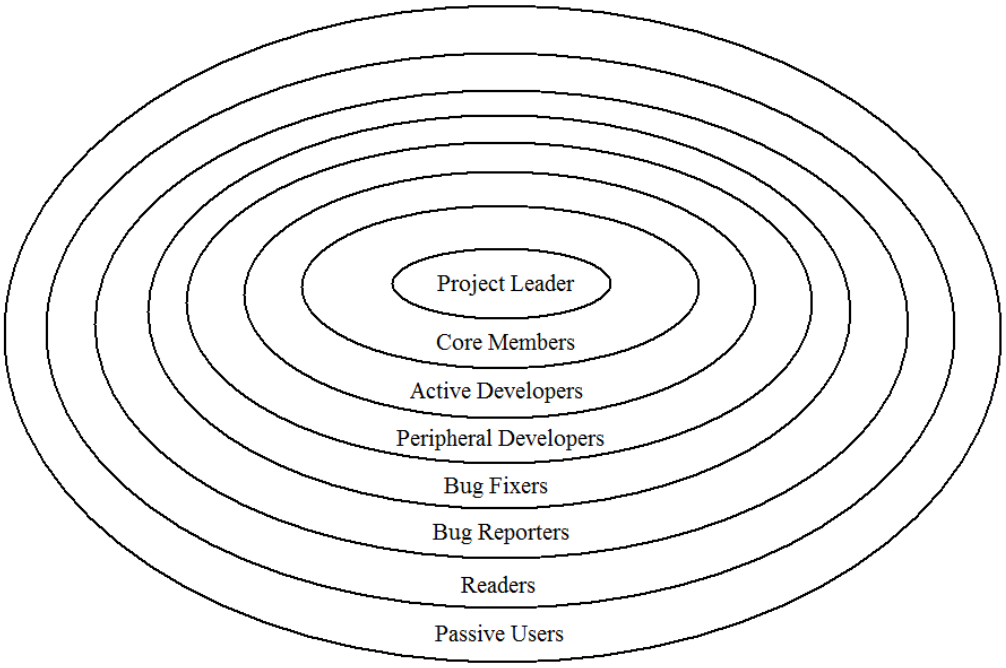

Figure 1. The onion model (Ye and Kishida, 2003).

Studying Freenet, an open source peer-to-peer file sharing network, Krogh et al. (2003) discovered developers-to-be would quite often lurk around (observe) the project and its mailing list(s) before actively participating. This lurking period would last anywhere on the order of weeks to months and was needed before they felt comfortable contributing to a technical discussion. In 40% of the cases examined, the first email by a future developer was to an ongoing technical discussion, rather than being an initiation of their own discussion. In what might be seen as a culture of humility, none of them gave unsolicited technical suggestions in their first post.

Herraiz et al. (2006) studied GNOME, an OSS project whose developer base includes people paid by various companies to work on it full-time. The study found a contrast between volunteer developers and the paid developers. The joining process of volunteers agreed with the onion model, but the process of paid developers was inconsistent with it, having all kinds of activity in a short period of time. Often developers would commit source code changes before ever sending a bug report. This study shows that even within open source, there is no one way to join.

Jensen and Scacchi (2007) studied role migration in the Mozilla, Apache, and NetBeans projects, uncovering the paths people follow to join. For Mozilla, a path for those without deep technical knowledge is a quality assurance path, contacting websites that don't display properly with Mozilla/Firefox, finding and reporting bugs, and working on documentation. A more technical path is for one to begin by submitting patches for bugs. These submissions can only be incorporated into the main source code by a developer with repository access, whom the person can contact through email and IRC. With enough of a reputation, the person can then become approved as a source code contributor, gaining code repository access. From there, he or she may become a module owner and/or code reviewer.

For Apache, there is a more linear path starting as an end-user before moving to become a developer. Developers are encouraged to also submit bug reports and documentation. If the Project Management Committee (PMC) is happy with the developer's contributions, it may offer the status of committer (developer with source code repository commit-access who reviews patches). The rest of the path's positions are by invitation only: PMC member, Apache Software Foundation (ASF) member, and finally the position of ASF board of directors member.

In the case of NetBeans.org, core members are primarily employees of Sun Microsystems, the project's main sponsor. In fact, some roles are not even open to those who do not work at the company. People new to the project are encouraged to start with quality assurance,

internationalization, documentation, and submitting patches. If one is shown to be reputable, he or she may gain commit access.

As we can see, Jensen and Scacchi's study shows us that two common characteristics of larger OSS projects are their hierarchical nature and the presence of corporation-employed core members. The ways of acquiring these roles are by volunteering, having them earned through work, having them assigned, and by being elected for them.

2.2.3.3 Open Source Tools

OSS developers utilize various tools to accomplish their work. Common tools include project web portals (such as SourceForge), code viewers (like ViewVC), source code control programs (usually CVS or the more recent Subversion), test support tools, mailing list managers (such as Mailman and MHonarc), bug trackers (like Bugzilla), and instant messaging (usually IRC) (Halloran and Scherlis, 2002). It should be noted that these tools are almost exclusively open source, themselves. Taking a bigger-picture view of the tools' functions, Halloran and Scherlis claim

> *"A ubiquitous, almost defining, trait of open source practice is that tool mediation is the norm. This enables leaders to shift the burden of policy enforcement from people to tools. Tools support authentication, regulation of commit privileges, audit and notification, and other policy-related functions." (2002)*

The mailing list is a particularly useful tool for coordinating OSS development and helping newcomers communicate with project veterans. A developer for Freenet asserted, "if you wanted to join an open-source project, the first thing you do is get on the mailing list" (Krogh et al, 2003). Another said "there would be a discussion about something, and I would just throw in my two cents about it. After a while, I was contributing to the discussion so much that everybody knew who I was and what I was doing there" (Krogh et al, 2003). Gutwin et al's 2004 study of NetBSD, Apache httpd, and Subversion, three widely used OSS projects with distributed development, found that mailing lists and text chat were the main source of group awareness (i.e. who is working on what parts of the project, what their plans are, etc.) They also found mailing lists to be useful for learning who the experts are on a topic. Since the messages are delivered to the whole group, the relevant people will make themselves known by joining the conversation. Text chat connects developers, like the mailing list, but with more off-topic social discussion and in real time.

2.2.4 Open Source Demographics

To gain insight as to who open source contributors are, David et al. (2003) conducted an online survey. The survey was posted to prominent websites and mailing lists in many countries across the globe. 1588 developers participated, the majority from Western Europe (53%) and North America (27%).

The results suggested open source software involvement is attractive to younger people. Nearly two-thirds of participants became involved with OSS between the ages of 16 to 25. On the other hand, less than 7% were 36 or older when they started. As for why they became involved with their first project in particular, the most common reason was the usefulness of the software to them personally (77%). The technical appeal of the project came in second (61%), followed by the participant him or herself launching the project (35%). (Survey participants could choose more than one reason as an answer for this question.) The general importance and visibility of the project fell far behind other reasons at 32%. Most participants had contributed to multiple projects (mode and median 3) with the ordinal ranking of reasons for working on their most recent project being similar to that of their first. These results support Krishnamurthy's findings of many projects having a single developer, as well as Raymond's famous assertion that "every good work of software starts by scratching a developer's personal itch" (1999).

The survey results also paint open source software development as an overwhelmingly male-dominated field. At 98.4%, virtually all of the participants identified themselves as male. Only 1.6% said they were female. Participants were well-educated, with 38% having an undergraduate level education as their highest education level, and another 37% having graduate (Masters or Doctoral) degrees. Yet another 6% held professional degrees (JD, MBA, etc.) High school was the highest education level completed for 19% of participants.

Regarding their occupational status at the time of the survey, 51.7% considered themselves an "employee", 28.8% "student", 15.9% "self employed", and 3.6% "not employed".

Almost half of participants (49.1%) were employed by an organization producing proprietary software within the last two years, while 30.6% had never been employed by such an organization. Counter to some people's misconceptions that one cannot make money working in open source, 14.6% of participants said they were paid for developing OSS, and 13.0% said they were paid for supporting it. Slightly over half (56.8%) said that they did not earn money from OSS involvement, directly or indirectly.

2.3 Related Studies

Lampe and Johnston (2005) showed that the feedback given to new members in an online community can have a strong effect on future participation. They studied Slashdot, a popular online tech community where members can leave comments, and some members can rate those comments. An analysis of server logs revealed that 11,079 users joined between November 1, 2004 and December 6, 2004. 1,763 (16%) of these users posted comments, making a total of 6,467 comments. Slightly over 55% of new members who made comments only made one. The new members who did not receive any ratings on their first comment were less likely to comment again than users who received ratings, whether positive or negative. New members who received replies to their first comment took less than a third of the time to post another comment than those who did not receive replies. These results suggest that feedback of any type is important for the continued participation of new members in an online community.

The 2003 Freenet study by Krogh et al. followed the project's developer discussion mailing list for a year. 11,210 email messages, spanning 1,714 threads were made by 356 people, for an average thread length of 6.5 messages. The creation of new threads by participants was common, with 78% of them starting at least one. However, 10.5% of them did not get a reply and did not appear on the mailing list again. Surprisingly, both developers and other list participants were equally likely to receive responses to their initial postings, and no statistically significant difference in mean thread length was found between threads instigated by developers and those started by others. Participation was very concentrated, with 4 developers (a mere 1.1% of the participants) posting half of the messages. Code commits to the project were also concentrated. About 8 percent of the participants had CVS commit access, but four out of 30 of them made 53% of the commits. Therefore, while the developer discussion list had many participants, few actually wrote code. As for message content, non-developers would ask more general questions, such as how to get the Freenet software running, and participants joining the project would report more bugs.

Crowston and Howison (2005) looked at centralization in the bug tracker data for OSS projects hosted by SourceForge. In this case, centrality was essentially a measure of how much participants on the bug forums communicated with a wide range of other people, as opposed to each interacting with just a few. At the time of their study, in 2002, there were more than 50,000 projects hosted. Of these, 140 met their basic study criteria of having at least seven developers

and over 100 bugs listed. One hundred and twenty of these had open access to bug reports and at least seven participants on the bug forums, so they studied those projects. One of their main findings was that centralization of interactions on the bug forums varied greatly among projects, following a normal distribution. Therefore simply being open source did not guarantee a "bazaar" model of communication or development. They also found that centralization was negatively correlated with the number of developers and users who contributed bug reports. Crowston and Howison's centralization findings were consistent with the developer mailing list interactions that they were able to examine for 52 of the projects.

Sowe et al. (2006) examined the "KDE", "mentors", and "user" mailing lists for the Debian Linux project from January 2001 to September 2004. They focus on a group of participants called "knowledge brokers" who connect the developers and users, posting information to each community. 136 people were identified who posted information to all lists, posting by far the majority of their messages on the user list (about 2,500 total for the KDE and mentors lists, but 10,290 for the user list). Interestingly, the mode number of posts by those on each list was one and the medians ranged from 2 to 5.5, suggesting the post count for this group was quite concentrated. In fact, one person posted 40% of the replies to questions asked in the user list. Narrowing the definition of a "broker" down to those who posted at least 10 messages, they identified 15 knowledge brokers. All of them saw themselves as both knowledge seekers and knowledge providers on the lists, and two-thirds of them thought other participants were cooperative.

A study of the Apache field support system on Usenet by Lakhani and von Hippel (2003) between 1996 and 1999 found that of 11,510 participants, 43% were solely information providers, 78% were solely information seekers, and only 21% were both providers and seekers in their posts to the forum. Consistent with the 2006 work of Sowe et al., a small portion of the information providers posted a very large portion of the answers to questions. Two percent of the providers posted 50% of the answers. Posts by the information seekers were much more distributed among participants, with 24% of them asking 50% of the questions. The study also found that 39% of information seekers received no reply on the forum. However, a survey of those people over a 4.5 month period examined showed that 40% of them received at least one reply privately by email. Lakhani and von Hippel divided the information seekers into two groups: "frequent seekers" who asked at least four questions over the four-year study period and "other seekers" who did not. Only 17% of survey respondents falling into the "other seeker"

category said that answers received completely solved their problem, but a much larger 44% said the answers merely gave them information that pointed them towards a solution to their problem. Thirty-nine percent said that the answers did not solve their problem. Interestingly, frequent seekers reported that the answers they received helped to solve and completely solved their problems more than the other seekers. Whether this is due to frequent seekers being more known on the forum and therefore receiving better or more thoughtful replies, or if it's due to frequent seekers becoming frequent seekers because of the helpfulness of the first replies is uncertain.

Bird et al. (2006) analyzed the Apache web server, PostgreSQL, and Python commit logs and mailing lists for three factors: Technical commitment ("how committed is the developer to the success of this project?"), skill level ("How knowledgeable/skillful is this developer relative to this specific project?"), and individual reputation ("What is the status of the individual in this community?") They propose a sort of race going on with each participant between the time for acquisition of skill and reputation needed to become a developer (officially), and loss of interest in doing so. The median time between the beginning of mailing list participation and first patch submission varied greatly. For Apache and PostgreSQL the time was 2 months, but for Python it was half a year. Social status, measured by social networks also correlated with the chance of attaining developer status.

3 Methodology

3.1 Data Harvesting

We obtained mbox files of four open source projects' user and developer mailing lists at their official websites or archives. The projects were chosen for their name recognition and archive availability. We collected data from January 1, 2005 through August 31, 2007 and divided that data into intervals: People who did not post between January 1, 2005 and October 31, 2006 but posted a valid message between November 1, 2006 and December 31, 2006 were classified as newbies for the latter interval. Posts made from March 1, 2007 through August 31, 2007 were analyzed to see whether or not these newbies stayed with the project.

"mbox" is the most common format for storing email messages, with each mbox file containing any number of messages (Qmail Documentation, 1998). We used Python's mailbox and email modules to extract relevant data from the mbox files. The following are the message headers whose information we extracted along with brief descriptions (Crocker, 1982):

- "From": Contains the identity of the message's sender
- "Subject": The message's topic/summary
- "In-reply-to": Information identifying which other message is being replied to, if any
- "References": Contains IDs of other messages referenced
- "Date": The date the message was sent
- "Message-ID": A unique identifier for the message

We also extracted the body (text) of the message, though there is no specific header denoting it. Rather, the body is the last block of text after the headers.

Unfortunately, some mbox files contained instances of malformed headers and improper formatting. We dealt with malformed headers that resulting in invalid dates, from data, and message IDs by tossing them out. Improper formatting was a trickier issue. In mbox format, messages are supposed to be separated by "From" headers. If a line of text in the body starts with "From ", it is supposed to be escaped to ">From " to avoid parsers mistakenly believing it is the start of a new message. Some mailing lists did not handle this correctly, creating instances where the parser would think the mbox file contained many large messages that did not actually exist. To catch major instances of this phenomenon, we had the parsing program flag all suspicious

messages (longer than 15,000 characters that contained at least seven “Message-ID” strings). Suspicious messages were then examined manually to determine which messages were truly instances of improper “From” escaping. We manually corrected these instances in the mbox files and reprocessed until no such improper messages were detected.

3.2 Email Address Grouping

A concern when analyzing mailing lists is that of people posting to a list from multiple email addresses. For instance, Mark Phippard from the Subversion user mailing list sent 6 emails from “markp@softlanding.com” during the period we examined, and 140 from another address, “markphip@gmail.com”. This raises an issue when trying to determine how many posts Mark Phippard made because assuming each distinct email address represents a distinct person is erroneous. We needed a way to match users to all addresses they posted from. There is clearly no one-hundred percent accurate way of doing this since those who wish to remain unidentified on the web, by spoofing email addresses and/or assuming multiple unrelated aliases, may always do so. However, there are ways of matching posters to email addresses in a reasonably accurate fashion. Since the “From” header in emails can contain both the sender’s address and his or her name, Sowe et al. (2006) matched addresses to people by attributing addresses with the same name to the same person. Bird et al. (2006) took into consideration that addresses can still be matched to the same person, even if the names are slightly different. For instance, a person named “Steven Harding” might also identify himself as “Steve Harding”, both names representing a single person, not two. We took a similar approach as Bird, with some modifications.

First we processed all (email address, name) pairs for each project’s mailing lists with a Python script to group them into suspected clusters, with each cluster representing a user. The program’s main steps are as follows:

1. Normalize the names by removing leading and trailing whitespace, periods, and titles (Jr., Dr., Sr., etc.)
2. Break the names up into three fields: First, Middle, and Last. The first name is the first word in the name, using space delimiters. The last is the last word (if not also the first), and the middle name is everything in between.
3. Merge two addresses’ clusters if the addresses meet any of the following criteria:
    a. The email addresses are identical.

b. The first and last names are identical. They can be in reverse order, so for example “Steven Harding” would match “Harding, Steven”.
c. The first and last names of one (address, name) pair are contained in the other’s address. For instance, “steven.q@harding.com” would match an address with the name “Steven Harding”.
d. The full names are similar. We defined the similarity S of two names, x and y, as

$$S_{x,y} = 1.0 - (D_{x,y} / \max (\text{length}_x, \text{length}_y))$$

where $D_{x,y}$ is the Levenshtein (edit) distance between x and y (Black, 2008). Since $D_{x,y}$ can range from zero (meaning x and y are exactly the same) to the maximum length of x or y (meaning x and y share none of the same characters), we divide by the maximum length to obtain a normalized value over the interval [0,1]. If this value is at least 0.85, we consider the names similar. This rule groups names with high similarity, such as “Steve Harding” with “Steven Harding” (S = 0.93).
e. The portion of the email addresses before the “@” symbol are equal and at least six characters long.
f. The portion of the email addresses before the “@” symbol are equal but less than six characters long, and the similarity between the full names is at least 0.75 as defined above.

4. For the sake of not creating huge clusters from some (address, name) pairs matching several dozen others inaccurately, some names were excluded from matching, if the name was the sole cause of the match. For instance generic full names like “webmaster”, “admin”, and “developer”, as well as common full names like “Sam”, “Martin”, and “Christopher”, would not trigger a match. Full names like “Sam Davidson” or “Martin Lee” would.

Once the addresses were matched with individual people programmatically, each cluster of addresses was manually processed to break them up into more accurate groups. The strategy was for the program to err on the side of assigning too many potential addresses to a person, but for the manual processing to take a closer look at the results, splitting the clusters into groups that were clear matches.

3.3 Message Threading

In order to determine how long conversations on a specific topic continued and which posters replied to which other posters, grouping messages into threads was necessary. We experimented with various threading methods, such as using the "In-Reply-To" and "References" headers, using the "Subject" headers, and Zawinski's Threading Algorithm (2002). In the end, we decided to thread the messages using Kuchling's implementation of Zawinski's Algorithm with minor modifications (2006). Some people might think it odd to not just use the "In-Reply-To" and "References" headers to construct threads since they should form a chain of messages from any message all the way up to the first post in the thread. While this is true in theory, it is not the case in practice (Zawinski, 2002). Reference headers can be truncated due to size limitations. More troubling is that they can be completely inaccurate due to posters using "Reply" buttons in email programs to save the effort of typing a list's email address for their new topic. This would lead the reference chain to indicate that the message is a reply to another message when it is actually a completely new thread. A person might also reply to a message with fresh reference headers, causing a reply to be classified as the start of a new thread. Grouping messages in a thread by "Subject" headers is an alternative, but it suffers from email programs' inconsistent handling of "Re:" tags and subject renaming by people replying to messages. Zawinski's Algorithm considers "In-Reply-To", "References", and "Subject" headers together to make reasonable judgments about threading. Its reliability (it was used in Netscape) and performance in our own tests led us to choose it.

3.4 Categorizing Posts

As mentioned earlier, we categorized a "newbie" on a mailing list as someone who posted a message to that mailing list in the November 1, 2006 to December 31, 2006 interval, and who had not posted a message in the preceding 22 months. We assumed that the absence of posts for 22 months would be a reasonable indication that a person was not previously a part of the project (and if they had, they would have been gone for long enough that many people in the community would not recognize them.) After identifying the newbies, we examined each one's first post to the list. These posts, as well as their first three responses, were classified for both participant and message-related attributes.

3.4.1 Participant Attributes

For each post, participants were categorized according to nationality and gender. We categorized nationality broadly, with each person being either a US poster or a non-US poster. Although one's true nationality cannot be discerned simply from a mailing list post, there are several clues that we considered. The most obvious clue was the email domain name suffix. People with addresses whose suffix corresponded to a non-US entity according to the Internet Assigned Numbers Authority (IANA) were categorized as non-US participants (Internet Corporation for Assigned Names and Numbers). A couple of non-US address examples are "svnlgo@mobsol.be" and "timo.lilja@hut.fi". Another clue was the presence of non-English email software, as evidenced by reply lines such as "am 04.11.2006 um 10:46 schrieb Julien Pons:". The other clues were in the message signatures, which sometimes included the posters' locations (often in the context of their place of business) and their telephone numbers. International numbers are formatted differently from the numbers people in the US are used to, for example "+41 44 272 91 61" (Kropla, 2006). If any of the clues indicated that the poster was non-US, that poster was categorized as such. However, if the clues indicated that a poster was from the US or if there was no basis for saying otherwise, that poster was categorized as a person of "US/unknown" nationality.

A participant's gender was classified as male, female, or unknown. Like nationality, there are no completely accurate ways of determining gender for all of the participants, but we examined names to make reasonable judgments. We wrote a computer program to compare each first name found in the "From" email header against a list of the 1,000 most common male and 1,000 most common female names according to the US Census Bureau, as well as a shorter list of top male and female German names (US Census Bureau, 2008, About.com, 2008). These lists were chosen for the breadth of the name types they spanned, as well as availability. Since the US is ethnically diverse, the top names are not only English names, but non-English names as well. If a poster's name was found exclusively in the male name list, the program categorized the poster as male. If her name was found exclusively in the female list, she was categorized as female. Posters whose names were found in both lists (i.e. unisex names like "Jamie" and "Jessie") or were not found in either list (i.e. uncommon names and internet nicknames) had their genders generally classified as unknown. Exceptions were made for unisex names that were much more common for one gender than the other, such as James which was over 300 times as common for males than females (US Census Bureau, 2008). Though the program classified gender, its results

were only treated as strong suggestions. These suggestions could be overwritten during the manual review of the posts given more solid evidence.

3.4.2 Message Attributes

Each post was also rated for general type, tone, and helpfulness. A message's general type, along with the criteria were as follows:

- Valid message: The message is legitimate and on-topic for the list.
- Not English/Unintelligible: The message is not in English or the rater is unable to make sense of it.
- Subscription message: The message appears to be a failed subscription/unsubscription to the list.
- Spam: The message is a commercial advertisement.
- Off topic (provocative): The message is off the topic of discussion and attempts to provoke more off-topic replies. "Baiting" is a term that comes to mind.
- Off topic (other): The message is off the topic of discussion, but innocently so. It is not meant to provoke.

Raters chose the category that fit each message best. They also did this for the poster's overall tone, on 5-point scale:

- 1 (friendly/personal): Message exceeds the basic standards of politeness, possibly relating on a personal level
- 2 (supportive/polite): Message meets basic standards of politeness, is generally positive
- 3 (neutral): Message is neither polite nor rude
- 4 (aggressive/rude): Message is unfriendly, rude, and/or insulting
- 5 (profane/flame): Message contains profanity/slurs in an offensive way, and/or personal attacks

Finally, each message was rated for its helpfulness:

- Helpful: Message provides useful/specific information for the topic of discussion

- Not helpful: Message fails to provide useful/specific information for the topic of discussion
- Not sure: Rater is unsure of the post's helpfulness
- Not applicable: Message is the first in the thread, or the helpfulness cannot be rated due to the nature of the thread

Because these criteria are subjective, each message was rated by three people. Threads containing posts that received completely different ratings on general message type or tone were discussed by at least three raters in person to determine a final unanimously agreed-upon rating. In the case of two out of the three raters agreeing on a classification, the one picked by the majority became the final rating.

4 Data Collection

This study examined four open source projects: MediaWiki, GIMP, PostgreSQL, and Subversion. We consider PostgreSQL and Subversion to be developer-oriented projects since the target end-users are software developers. MediaWiki and GIMP are here considered non-developer-oriented projects since the targeted end-user is someone who is not necessarily a programmer. The projects were chosen with this variety of target audiences in mind, the availability of mailing list archives, and their popularity. Each of these projects had a developer list and a separate user mailing list. The user lists are "mediawiki-l", "gimp-user", "pgsql-general", and "svn-users". They can be thought of as the general catch-all lists for people to use to communicate about non-developer aspects of the projects. The other mailing lists are aimed at developers and more technical discussions.

Table 1. Overview of mailing lists and projects studied. Posts, Posters, and Percentage of Posts Made by Core Members statistics calculated for the period between Nov. 1, 2006 and Dec. 1, 2007. For this study, we define a "core member" as a poster who ranks in the top 10% for posts made.

Mailing List	Description of Project	Posts	Unique Posters	% of Posts Made by Core Members
mediawiki-l	MediaWiki: Wiki software, the engine behind Wikipedia	8,422	806	66%
wikitech-l		7,094	399	71%
gimp-user	GNU Image Manipulation Program (GIMP): Photo/image editing program, often compared to Adobe Photoshop	2,431	397	56%
gimp-developer		2,267	224	64%
pgsql-general	PostgreSQL: An object-relational database system	19,606	1,827	70%
pgsql-hackers		16,360	568	83%
svn-users	Subversion: A version control system, seen by some as a successor to CVS	13,721	2,331	63%
svn-dev		10,586	559	83%

In the two months analyzed, 643 newbies started threads by posting valid messages on the eight mailing lists. The results in this study are based off of the first four posts in the threads started by those newbies. 101 additional newbies replied to someone else's thread as their first

post rather than start their own thread, and they are excluded from our study. This was a necessary measure because, in addition to being a small minority of newbies (accounting for only 13.6% of all newbies) it is difficult to classify which future reply is associated with the newbie, and which is directed at one of the other posters. Replies on threads that a newbie initiated are less ambiguous. This is important because we want to examine how replies to newbies affect their future posting.

Table 2. Newbies posting to lists between November 1, 2006 and December 31, 2006. The group of newbies initiating a thread as their first post is the group considered in this study. Newbies who posted a reply to someone else's thread as their first post are ignored.

Mailing List	Newbies Initiating a Thread as First Post	Newbies Replying to a Thread as First Post	Total
mediawiki-l	83 (95.4%)	4 (4.6%)	**87**
wikitech-l	19 (61.3%)	12 (38.7%)	**31**
gimp-user	39 (75.0%)	13 (25.0%)	**52**
gimp-developer	10 (76.9%)	3 (23.1%)	**13**
pgsql-general	148 (86.5%)	23 (13.5%)	**171**
pgsql-hackers	36 (76.6%)	11 (23.4%)	**47**
svn-users	271 (90.9%)	27 (9.1%)	**298**
svn-dev	37 (82.2%)	8 (17.8%)	**45**
TOTAL	**643 (86.4%)**	**101 (13.6%)**	**744**

Across all lists, 250 posters (26.5%) were identified as being of clearly non-US persons. 695 (73.5%) were identified as of US or indeterminate nationality. This is in contrast to the 2003 FLOSS Survey in which 23.49% were from the US and the country with the most respondents was Germany at 25.17% (David et al.)

Only 16 females were identified out of 945 total posters participating on the newbie-initiated threads analyzed. Of these 16, only one was not a newbie[1]. 582 males were identified, leaving

[1] Only one was not a newbie in terms of having posted before. Another female was a newbie who replied to a newbie in her first post, but newbies who reply to threads, rather than starting them, are not in this study.

347 posters of indeterminate gender. Disregarding people of unknown gender (36.7% of all posters), 2.68% of posters were female. Taking into consideration only the developer lists, 2 out of 176 posters (1.14%) were female. Assuming posters on the developer lists are, in fact, developers in one way or another, this is only a slightly lower percentage than found in David et al.'s 2003 survey of OSS developers (1.61% female). See Table 4.

This study did not find female participation being more or less common statistically on user lists versus developer lists. To calculate this, we used Fisher's exact test, a test with similar uses to a chi-square test for independence, but for small samples (McDonald, 2008). Female posters were not found as more common on user versus developer lists (p-value = .748). Similarly, female newbies were not found to be more common on one type of list over the other (p-value = 1).

Table 3. Number of newbies by demographic.

Mailing List	Male	Female	Unknown Gender		Non-US	US/Unknown Nationality	Total Newbies
mediawiki-l	45 (54.2%)	6 (7.2%)	32 (38.6%)		24 (28.9%)	59 (71.1%)	**83**
wikitech-l	15 (78.9%)	0 (0.0%)	4 (21.1%)		8 (42.1%)	11 (57.9%)	**19**
gimp-user	27 (69.2%)	1 (2.6%)	11 (28.2%)		13 (33.3%)	26 (66.7%)	**39**
gimp-developer	1 (10.0%)	0 (0.0%)	9 (90.0%)		2 (2.0%)	8 (8.0%)	**10**
pgsql-general	74 (50.0%)	1 (0.7%)	73 (49.3%)		44 (29.7%)	104 (70.3%)	**148**
pgsql-hackers	19 (52.8%)	1 (2.8%)	16 (44.4%)		12 (33.3%)	24 (66.7%)	**36**
svn-users	170 (62.7%)	4 (1.5%)	97 (35.8%)		70 (25.8%)	201 (74.2%)	**271**
svn-dev	22 (59.5%)	1 (2.7%)	14 (37.8%)		12 (32.4%)	25 (67.6%)	**37**
TOTAL	**373 (58.0%)**	**14 (2.2%)**	**256 (39.8%)**		**185 (28.8%)**	**458 (71.2%)**	**643**

Table 4. Number of posters by demographic. Includes newbies from Table 3.

Mailing List	Male	Female	Unknown Gender		Non-US	US/Unknown Nationality	Total Posters
mediawiki-l	62 (54.9%)	6 (5.3%)	45 (39.8%)		29 (25.7%)	84 (74.3%)	**113**
wikitech-l	26 (76.5%)	0 (0.0%)	8 (23.5%)		13 (38.2%)	21 (61.8%)	**34**
gimp-user	44 (65.7%)	2 (3.0%)	21 (31.3%)		20 (29.9%)	47 (70.1%)	**67**
gimp-developer	5 (31.3%)	0 (0.0%)	11 (68.8%)		4 (25.0%)	12 (75.0%)	**16**
pgsql-general	123 (54.7%)	1 (0.4%)	101 (44.9%)		62 (27.6%)	163 (72.4%)	**225**
pgsql-hackers	43 (62.3%)	1 (1.4%)	25 (36.2%)		18 (26.1%)	51 (73.9%)	**69**
svn-users	243 (66.8%)	5 (1.4%)	116 (31.9%)		88 (24.2%)	276 (75.8%)	**364**
svn-dev	36 (63.2%)	1 (1.2%)	20 (35.1%)		16 (28.1%)	41 (71.9%)	**57**
TOTAL	**582 (61.6%)**	**16 (1.7%)**	**347 (36.7%)**		**250 (26.5%)**	**695 (73.5%)**	**945**

5 Data Analysis

This section will investigate the research questions, presenting results related to newbies' success in joining the mailing list communities, whether or not newbies received timely replies, and the effects of courtesy, gender, and nationality on the lists.

5.1 Newbies' Success Rates in Joining a Mailing List Community

To determine how many newbies persisted in their attempts to join the community we examined the postings made during the 3-month period from March 1, 2007 through May 31, 2007 (about 3 months out from our initial observations) for new posts from our previously identified newbies. Few (13.5%) posted again in that time. Looking farther into the future (from June 1, 2007 and August 31, 2007), this group was again halved (6.4% of the original group). This indicates a significant drop-off of newbie participation over time ($\chi 2 = 18.359$, p-value < .001). The data clearly shows that most newbie posters to mailing lists do not contribute to the long-term discussion. It should also be noted that a small group (3% of the original newbies) did not post during the March-May interval, but reappeared during the June-August interval. It is possible that these newbies were still lurking in the community after their first postings.

There are many natural reasons for these low numbers. The majority of newbies posting on mailing lists are posting to get help on a specific technical problem, and not because of a conscious long-term plan to join the community. Therefore, once these help-seeking newbies resolve their problem, with or without the help of the mailing list community, a large majority of them are likely to disappear until some new issue emerges.

Comparing the portions of newbies who continued to post on the mailing lists during the first interval (3-6 months) and the second interval (6-9 months), no distinction could be made between those in user lists and those in developer lists ($\chi 2 = .048$, p-value = .827) or between those in developer projects versus non-developer projects ($\chi 2 = 2.771$, p-value = .096).

Table 5. Number of newbies posting after their initial post. Shows the number of newbies who posted during the two-month study phase, approximately 3-6 months after their initial post, as well as those who posted again 6-9 months after their initial post. Corresponds to Figure 2 below.

Mailing List	Newbies	Newbies Posting 3-6 Months Later	Newbies Posting 6-9 Months Later
mediawiki-l	83	13 (15.7%)	7 (8.4%)
wikitech-l	19	3 (15.8%)	2 (10.5%)
gimp-user	39	6 (15.4%)	3 (7.7%)
gimp-developer	10	2 (20.0%)	2 (20.0%)
pgsql-general	148	25 (16.9%)	15 (10.1%)
pgsql-hackers	36	3 (8.3%)	1 (2.8%)
svn-users	271	30 (11.1%)	9 (3.3%)
svn-dev	37	5 (13.5%)	2 (5.4%)
TOTAL	**643**	**87 (13.5%)**	**41 (6.4%)**

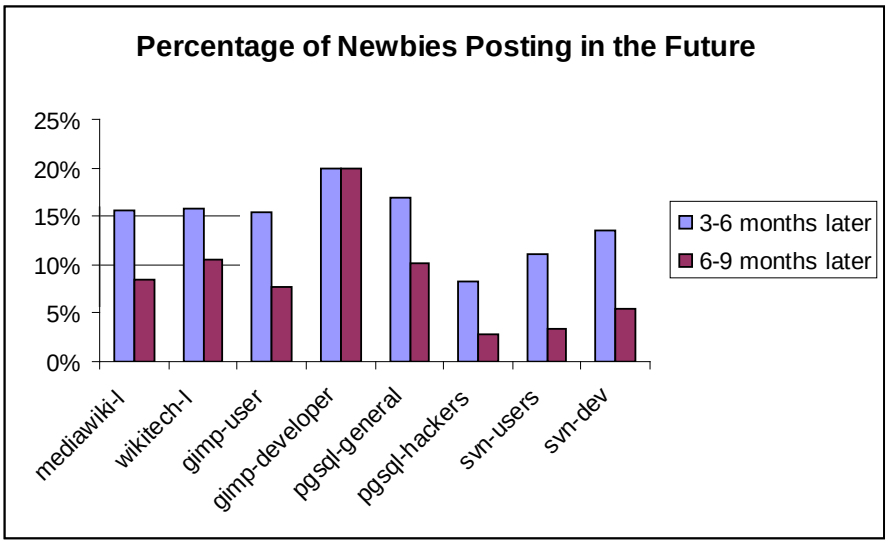

Figure 2. Percentage of newbies posting after their initial post. Shows those who posted approximately 3-6 months after their initial post (March-May interval), as well as those who posted again 6-9 months after their initial post (June-August interval). This does not include the 3% who posted 6-9 months later, but not 3-6 months later.

5.2 Factors That Affect Success

To determine which properties or factors affected the likelihood of a newbie staying with the community, we applied logistic regression, a technique that can predict a dependent variable given independent variables (McDonald, 2008). In our case, the dependent variable is whether or not a newbie continued posting 3-9 months after his/her initial post.

The logistic regression model tries to find a statistical model using all statistically significant independent variables, and determine how much of the overall sample variance these variables account for. When dealing with uncontrolled, natural observations of users, the observed variables tend to account for only a small percentage of the overall variance due to the vast number of external factors and motivations inherent in normal human behavior. Our model showed that the variables we tracked accounted for 3% of the variance in the future participation of the newbies in our sample. While this seems low, it is important to remember that only a small portion of the newbies have any intention whatsoever of staying on the list long-term. Many simply have a technical question or an issue getting the software installed and configured. These casual posters with no intent of ever joining are noise in the model.

Analysis showed that the statistically significant factors were whether or not the newbie received a reply within 48 hours ($p = .012$) and whether or not the newbie posted to a Subversion mailing list ($p = .024$). If a newbie received a reply within 48 hours, he or she was more likely to continue posting in the future. This is consistent with Lampe and Johnston's study of Slashdot, which found that feedback to newbie posts was correlated with future participation (2005). Being on a Subversion list had a negative correlation with future participation. One possible explanation for this is the technical or infrastructure nature of the project, leading to an increased frequency of the one-question posters mentioned above.

5.3 Helpfulness of Replies to Newbies

As stated in section 3.4.2, three people rated each post in order to determine its helpfulness in addressing a newbie's questions. The rating categories were "helpful", "not helpful", "not sure", and "not applicable" (this one used to denote posts made by the newbie him or herself.). For analysis, these categories were combined into two: "clearly helpful", consisting of the helpful

rating, and "not clearly helpful", consisting of everything else.[2] In the cases where two out of three raters agreed on a rating, that rating prevailed as the final rating for use in the study. The rater agreement for the posts' helpfulness is as follows:

Table 6. Rater agreement for helpfulness ratings.

Mailing List	Posts With Total (3 / 3) Rater Agreement	Posts With Majority (2 / 3) Rater Agreement
mediawiki-l	146 (76.4%)	45 (23.6%)
wikitech-l	42 (70.0%)	18 (30.0%)
gimp-user	72 (60.0%)	48 (40.0%)
gimp-developer	22 (59.5%)	15 (40.5%)
pgsql-general	327 (73.6%)	117 (26.4%)
pgsql-hackers	82 (71.3%)	33 (28.7%)
svn-users	487 (71.4%)	195 (28.6%)
svn-dev	78 (69.0%)	35 (31.0%)
TOTAL	**1256 (71.3%)**	**506 (28.7%)**

The data indicates that replies to newbies' posts were generally helpful, with the majority of replies to newbies' posts (69.3%) being rated as such. At least half of replies rated were helpful on every list, except for gimp-developer. It is interesting to note that the developer-oriented projects, PostgreSQL and Subversion, had a statistically-significantly larger proportion of helpful replies than the other lists ($\chi 2 = 15.595$, p-value < .001). While the reason for this cannot be determined from this study, one possible answer might lie in the traffic levels in the lists analyzed. The greater traffic of the developer-oriented project lists could presumably mean more people available to consider the newbies' posts, leading to an increased likelihood someone knowledgeable could see the question and respond.

There were no statistically significant differences in the helpfulness of replies between the user lists and developer lists of the projects we examined ($\chi 2 = 1.482$, p-value = .223).

[2] Therefore, when we say a reply to a newbie is "helpful" in this study, we are referring to this "clearly helpful" category.

Table 7: Helpfulness of replies to newbies' first posts.

Mailing List	Helpful Replies	Non-helpful Replies	Percent Helpful
mediawiki-l	56	27	67.5%
wikitech-l	11	11	50.0%
gimp-user	32	31	50.8%
gimp-developer	8	10	44.4%
pgsql-general	177	50	78.0%
pgsql-hackers	44	12	78.6%
svn-users	213	95	69.2%
svn-dev	32	18	64.0%
TOTAL	**573**	**254**	**69.3%**

5.4 Politeness of Replies to Newbies

As above, three people rated each post in order to determine the post's tone. The original categories were "1 (friendly/personal), "2 (supportive/polite)", "3 (neutral)", "4 (aggressive/rude)", and "5 (profane/flame)". These categories were then combined into three groups: "polite", consisting of the "1" and "2" ratings, "neutral", consisting of the "3" rating, and "rude", consisting of the "4" and "5" ratings. In the cases where two out of three raters agreed on a rating, that rating prevailed as the final rating for use in the study. In the cases where all raters disagreed, at least three raters discussed the posts in question and decided upon a rating that they all agreed with. The rater agreement for the posts' tone is as follows:

Table 8: Rater agreement for tone ratings. Ratings on which all raters disagreed initially were discussed until a unanimous classification was agreed upon, and so they show up as 3/3 agreement.

Mailing List	Posts With Total (3 / 3) Rater Agreement	Posts With Majority (2 / 3) Rater Agreement
mediawiki-l	105 (55.0%)	86 (45.0%)
wikitech-l	36 (60.0%)	24 (40.0%)
gimp-user	64 (53.3%)	56 (46.7%)
gimp-developer	21 (56.8%)	16 (43.2%)
pgsql-general	256 (57.7%)	188 (42.3%)
pgsql-hackers	50 (43.5%)	65 (56.5%)
svn-users	446 (65.4%)	236 (34.6%)
svn-dev	56 (49.6%)	57 (50.4%)
TOTAL	**1034 (58.7%)**	**728 (41.3%)**

Out of the 827 replies analyzed, only 6.8% were polite. The vast majority, 91.8%, were neutral in tone. One and a half percent were rude. Therefore, the flame wars and elitism often feared by prospective newbies posting on mailing lists were almost non-existent, or at least not aimed at them. An interesting finding is that newbies on developer lists received more polite replies on average than newbies on user lists ($\chi 2 = 3.902$, p-value = .048). Furthermore, newbies on lists for non-developer projects received more polite replies on average than those on developer projects ($\chi 2 = 6.025$, p-value = .014).

Table 9. Tone of replies to newbies' first posts.

Mailing List	Polite Replies	Neutral Replies	Rude Replies	Total
mediawiki-l	9 (10.8%)	74 (89.2%)	0 (0.0%)	**83**
wikitech-l	3 (13.6%)	18 (81.8%)	1 (4.5%)	**22**
gimp-user	8 (12.7%)	54 (85.7%)	1 (1.6%)	**63**
gimp-developer	0 (0.0%)	17 (94.4%)	1 (5.6%)	**18**
pgsql-general	12 (5.3%)	213 (93.8%)	2 (0.9%)	**227**
pgsql-hackers	5 (8.9%)	47 (83.9%)	4 (7.1%)	**56**
svn-users	12 (3.9%)	293 (95.1%)	3 (1.0%)	**308**
svn-dev	7 (14.0%)	43 (86.0%)	0 (0.0%)	**50**
TOTAL	**56 (6.8%)**	**759 (91.8%)**	**12 (1.5%)**	**827**

5.5 Timeliness of Replies to Newbies

The time between newbies' first posts and the first reply they received indicates that OSS communities do not usually ignore newbies. In fact, the majority of newbies on each list examined received a reply within 24 hours. Across all lists, only 23.3% of newbies did not receive any replies. A particularly interesting finding is that newbies posting on user lists were significantly less likely to receive a reply than those posting on developer lists ($\chi 2 = 3.958$, p-value = .047).

Table 10. Time elapsed before newbies received a reply to their post.

Mailing List	24 Hours or Less	Between 1 day and 1 week	Later than 1 week	Received No Reply
mediawiki-l	48 (57.8%)	3 (3.6%)	1 (1.2%)	31 (37.3%)
wikitech-l	11 (57.9%)	1 (5.3%)	0 (0.0%)	7 (36.8%)
gimp-user	31 (79.5%)	4 (10.3%)	1 (2.6%)	3 (7.7%)
gimp-developer	7 (70.0%)	3 (30.0%)	0 (0.0%)	0 (0.0%)
pgsql-general	97 (65.5%)	32 (22.3%)	2 (1.4%)	17 (11.5%)
pgsql-hackers	28 (77.8%)	4 (11.1%)	1 (2.8%)	3 (8.3%)
svn-users	150 (55.4%)	36 (13.3%)	2 (0.7%)	83 (30.6%)
svn-dev	20 (54.1%)	9 (24.3%)	2 (5.4%)	6 (16.2%)
TOTAL	**392 (61.0%)**	**92 (14.3%)**	**9 (1.4%)**	**150 (23.3%)**

5.6 Who Replies to Newbies

Did the core members (defined for this study as those ranking in the top 10% for posts made) stick mainly to talking amongst themselves, or would they reply to newbies' posts? To answer this, it was necessary to compare the number of threads started by newbies[3] who received replies from core members to the number of threads started by non-newbies who received such replies.

No significant relation was found between being a newbie and a non-newbie when it came to receiving a response from a core member ($\chi 2 = 0.615$, p-value = .433). Considering only threads started by newbies, no significant relation was found between posting on a user list versus a developer list regarding receiving a core member reply ($\chi 2 = 2.538$, p-value = .111).

Table 11. Number of core member replies to newbies versus non-newbies.

Mailing List	Threads Started by Newbies (23.4%)				Threads Started by Non-Newbies (76.6%)		
	Received Core Member Reply	No Core Member Reply	Total		Received Core Member Reply	No Core Member Reply	Total
mediawiki-l	41 (49.4%)	42 (50.6%)	83		147 (60.7%)	95 (39.3%)	242
wikitech-l	10 (52.6%)	9 (47.4%)	19		111 (51.6%)	104 (48.4%)	215
gimp-user	29 (74.4%)	10 (25.6%)	39		32 (61.5%)	20 (38.5%)	52
gimp-developer	10 (100.0%)	0 (0.0%)	10		17 (63.0%)	10 (37.0%)	27
pgsql-general	127 (85.8%)	21 (14.2%)	148		419 (73.8%)	149 (26.2%)	568
pgsql-hackers	28 (77.8%)	8 (22.2%)	36		213 (80.1%)	53 (19.9%)	266
svn-users	174 (64.2%)	97 (35.8%)	271		280 (61.1%)	178 (38.9%)	458
svn-dev	30 (81.1%)	7 (18.9%)	37		215 (78.2%)	60 (21.8%)	275
TOTAL	**449 (69.8%)**	**194 (30.2%)**	**643**		**1434 (68.2%)**	**669 (31.8%)**	**2,103**

[3] For this subsection, "newbie" can be thought of as a first-time-poster: If a newbie started another thread later, that thread is counted as being started by a non-newbie.

5.7 Effects of Newbie Tone

While one might expect a polite newbie post to evoke the most helpful responses from the community, the data only weakly supports this. Polite newbies received a greater percentage of helpful replies than newbies who started their threads with neutral-tone messages in 5 out of 8 lists, but statistically, the effect of newbie tone was only marginally significant ($\chi 2 = 3.042$, p-value = .081). It should be noted however that given the very small number of rude replies, they are omitted from the statistical analysis. We examine only polite versus neutral replies for comparative purposes.

Table 12. Helpfulness of replies by newbies' tones. Omits one rude newbie who received 2 helpful replies and 1 non-helpful reply.

Mailing List	Polite Newbies (36.4%)			Neutral Newbies (63.6%)		
	Helpful Replies	**Non-helpful Replies**	**Total**	**Helpful Replies**	**Non-helpful Replies**	**Total**
mediawiki-l	28 (75.7%)	9 (24.3%)	37	28 (60.9%)	18 (39.1%)	46
wikitech-l	7 (58.3%)	5 (41.7%)	12	4 (40.0%)	6 (60.0%)	10
gimp-user	11 (73.3%)	4 (26.7%)	15	21 (43.8%)	27 (56.3%)	48
gimp-developer	0 (0.0%)	3 (100.0%)	3	8 (53.3%)	7 (46.7%)	15
pgsql-general	55 (87.3%)	8 (12.7%)	63	122 (74.4%)	42 (25.6%)	164
pgsql-hackers	14 (70.0%)	6 (30.0%)	20	28 (84.8%)	5 (15.2%)	33
svn-users	95 (68.3%)	44 (31.7%)	139	118 (69.8%)	51 (30.2%)	169
svn-dev	9 (81.8%)	2 (18.2%)	11	23 (59.0%)	16 (41.0%)	39
TOTAL	**219 (73.0%)**	**81 (27.0%)**	**300**	**352 (67.2%)**	**172 (32.8%)**	**524**

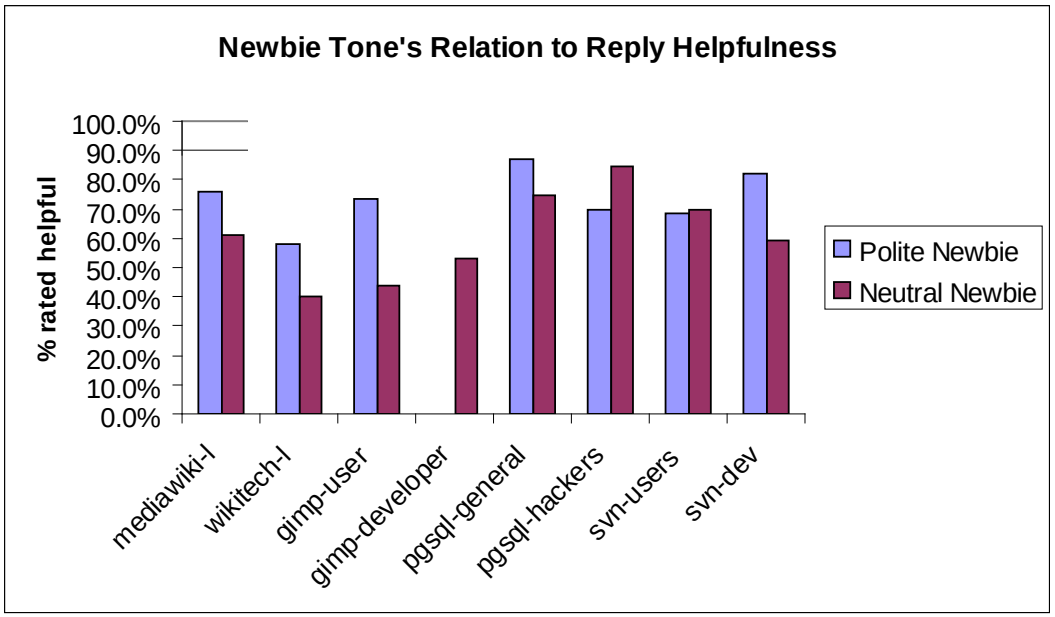

Figure 3. Relation of reply helpfulness to newbie tone.

Common sense would cause one to expect that polite posts would be reciprocated with polite replies. However, no distinction could be made between polite newbies and neutral newbies in the tone of replies received ($\chi 2 = 1.273$, p-value = .259).

Table 13. Tone of replies by newbies' tones. Omits 10 rude replies (3 of which were to polite newbies) in addition to one newbie who received 2 rude replies and 1 neutral reply.

Mailing List	Polite Newbies (36.2%)				Neutral Newbies (63.8%)		
	Polite Replies	Neutral Replies	Total		Polite Replies	Neutral Replies	Total
mediawiki-l	6 (16.2%)	31 (83.8%)	37		3 (6.5%)	43 (93.5%)	46
wikitech-l	1 (8.3%)	11 (91.7%)	12		2 (22.2%)	7 (77.8%)	9
gimp-user	2 (13.3%)	13 (86.7%)	15		6 (12.8%)	41 (87.2%)	47
gimp-developer	0	0	0		2 (11.8%)	15 (88.2%)	17
pgsql-general	7 (11.1%)	56 (88.9%)	63		5 (3.1%)	157 (96.9%)	162
pgsql-hackers	1 (5.6%)	17 (94.4%)	18		4 (12.1%)	29 (87.9%)	33
svn-users	5 (3.6%)	134 (96.4%)	139		7 (4.2%)	159 (95.8%)	166
svn-dev	3 (27.3%)	8 (72.7%)	11		4 (10.3%)	35 (89.7%)	39
TOTAL	**25 (8.5%)**	**270 (91.5%)**	**295**		**33 (6.4%)**	**486 (93.6%)**	**519**

The newbies' tones had no significant correlation with whether or not they received a reply ($\chi 2 = .08$, p-value = .777). This information, combined with the above finding that newbie tone and reply helpfulness are slightly correlated, suggests that perhaps people will reply to a post despite its tone but give more thought to those which are polite.

Table 14. Time elapsed before newbies received a reply to their post, by newbie tone. Omits 1 rude newbie (who received a reply in 24 hours or less).

Mailing List	Polite Newbie (36.3%)			Neutral Newbie (63.7%)		
	24 Hours or Less	More Than 24 Hours	No Reply	24 Hours or Less	More Than 24 Hours	No Reply
mediawiki-l	21 (63.6%)	1 (3.0%)	11 (33.3%)	26 (54.2%)	2 (4.2%)	20 (41.7%)
wikitech-l	5 (45.5%)	1 (9.1%)	5 (45.5%)	6 (75.0%)	0 (0.0%)	2 (25.0%)
gimp-user	8 (72.7%)	1 (9.1%)	2 (18.2%)	23 (82.1%)	4 (14.3%)	1 (3.6%)
gimp-developer	1 (100.0%)	0 (0.0%)	0 (0.0%)	6 (75.0%)	2 (25.0%)	0 (0.0%)
pgsql-general	27 (61.4%)	11 (25.0%)	6 (13.6%)	70 (67.3%)	23 (22.1%)	11 (10.6%)
pgsql-hackers	10 (83.3%)	1 (8.3%)	1 (8.3%)	17 (73.9%)	4 (17.4%)	2 (8.7%)
svn-users	64 (58.7%)	19 (17.4%)	26 (23.9%)	86 (53.1%)	19 (11.7%)	57 (35.2%)
svn-dev	8 (72.7%)	1 (9.1%)	2 (18.2%)	12 (46.2%)	10 (38.5%)	4 (15.4%)
TOTAL	**144 (62.1%)**	**35 (15.1%)**	**53 (22.8%)**	**246 (60.4%)**	**64 (15.7%)**	**97 (23.8%)**

5.8 Effects of Newbie Nationality

No statistical distinction could be made between the helpfulness of replies to non-US newbies and the helpfulness of replies to US and indeterminate nationality newbies ($\chi 2 = .728$, p-value = .393).

Table 15. Helpfulness of replies by newbies' nationality.

Mailing List	US/Unknown Nationality (74.0%)			Non-US Nationality (26.0%)		
	Helpful Replies	**Non-helpful Replies**	**Total**	**Helpful Replies**	**Non-helpful Replies**	**Total**
mediawiki-l	45 (67.2%)	22 (32.8%)	67	11 (68.8%)	5 (31.3%)	16
wikitech-l	7 (53.8%)	6 (46.2%)	13	4 (44.4%)	5 (55.6%)	9
gimp-user	21 (48.8%)	22 (51.2%)	43	11 (55.0%)	9 (45.0%)	20
gimp-developer	6 (42.9%)	8 (57.1%)	14	2 (50.0%)	2 (50.0%)	4
pgsql-general	124 (80.0%)	31 (20.0%)	155	53 (73.6%)	19 (26.4%)	72
pgsql-hackers	32 (80.0%)	8 (20.0%)	40	12 (75.0%)	4 (25.0%)	16
svn-users	171 (70.1%)	73 (29.9%)	244	42 (65.6%)	22 (34.4%)	64
svn-dev	23 (63.9%)	13 (36.1%)	36	9 (64.3%)	5 (35.7%)	14
TOTAL	**429 (70.1%)**	**183 (29.9%)**	**612**	**144 (67.0%)**	**71 (33.0%)**	**215**

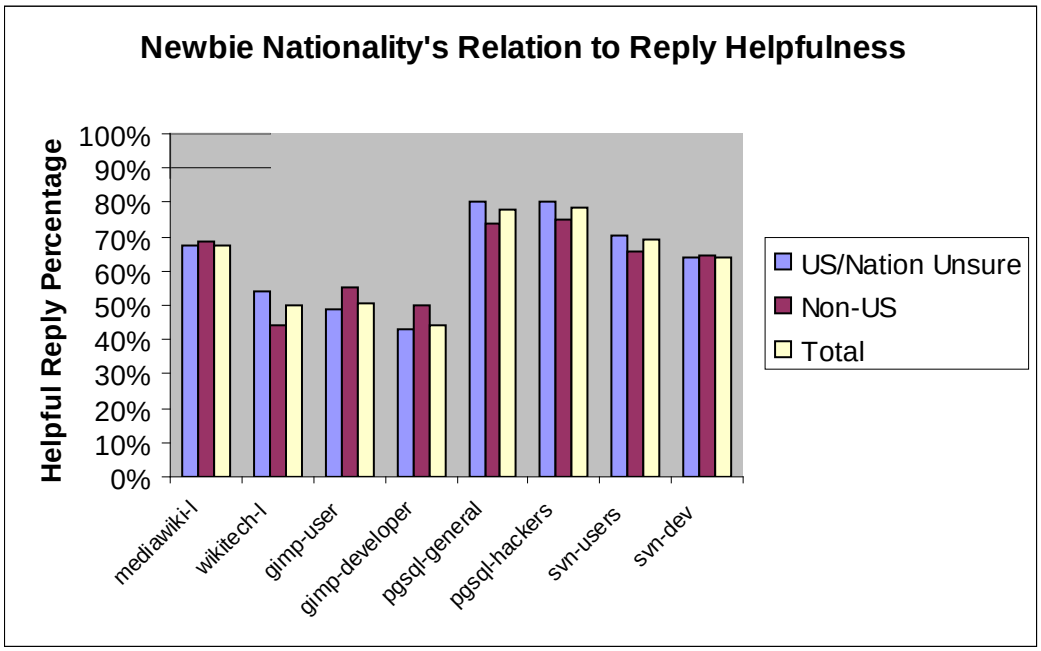

Figure 4. Percentage of helpful replies by newbies' nationality.

Newbies who were clearly of non-US nationality received a lower raw percentage of polite replies than other newbies in 5 out of 8 lists. Statistically, however, no distinction could be made between the US/unknown nationality newbie and non-US newbie groups in this respect ($\chi 2$ = .041, p-value = .839).

Table 16. Tone of replies by newbies' nationality. Omits 12 rude replies, 6 to US/Unknown nationality newbies and 6 to Non-US newbies.

Mailing List	US/Unknown Nationality (74.4%)			Non-US Nationality (25.6%)		
	Polite Replies	Neutral Replies	Total	Polite Replies	Neutral Replies	Total
mediawiki-l	8 (11.9%)	59 (88.1%)	67	1 (6.3%)	15 (93.8%)	16
wikitech-l	2 (15.4%)	11 (84.6%)	13	1 (12.5%)	7 (87.5%)	8
gimp-user	3 (7.1%)	39 (92.9%)	42	5 (25.0%)	15 (75.0%)	20
gimp-developer	0 (0.0%)	14 (100.0%)	14	0 (0.0%)	3 (100.0%)	3
pgsql-general	10 (6.5%)	145 (93.5%)	155	2 (2.9%)	68 (97.1%)	70
pgsql-hackers	2 (5.6%)	34 (94.4%)	36	3 (18.8%)	13 (81.3%)	16
svn-users	10 (4.1%)	233 (95.9%)	243	2 (3.2%)	60 (96.8%)	62
svn-dev	6 (16.7%)	30 (83.3%)	36	1 (7.1%)	13 (92.9%)	14
TOTAL	**41 (6.8%)**	**565 (93.2%)**	**606**	**15 (7.2%)**	**194 (92.8%)**	**209**

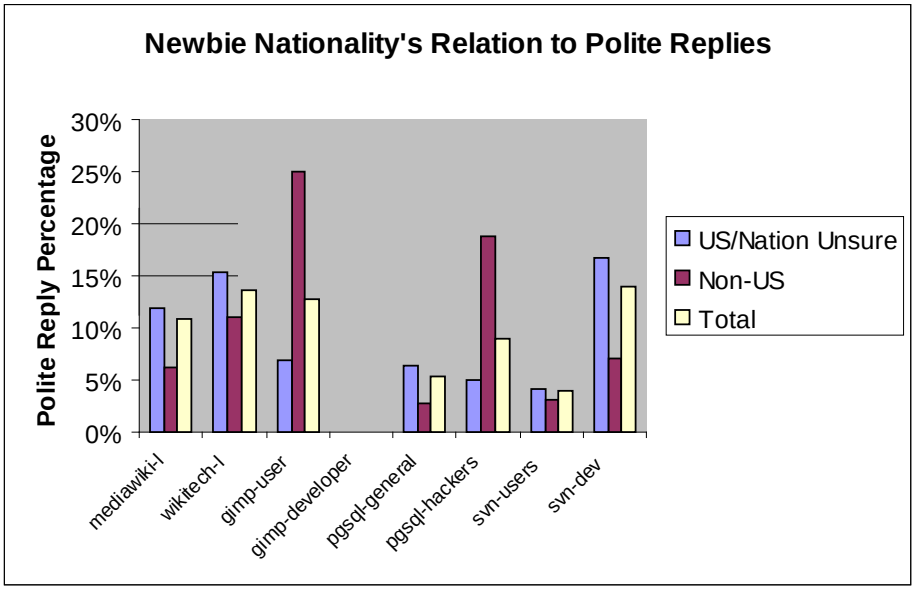

Figure 5. Percentage of polite replies by newbies' nationality.

A greater percentage of non-US newbies were not replied to, and the ones who were received a larger portion of later replies than their US/unknown nationality counterparts. However, comparing the US/unknown newbies and the non-US newbies who did not receive replies, this difference was not statistically significant ($\chi 2 = 2.610$, p-value = .106).

Table 17. Time elapsed before newbies received a reply to their post, by nationality.

Mailing List	US/Unknown Nationality (74.4%)			Non-US Nationality (25.6%)		
	24 Hours or Less	More Than 24 Hours	No Reply	24 Hours or Less	More Than 24 Hours	No Reply
mediawiki-l	38 (64.4%)	3 (5.1%)	18 (30.5%)	10 (41.7%)	1 (4.2%)	13 (54.2%)
wikitech-l	8 (72.7%)	0 (0.0%)	3 (27.3%)	3 (37.5%)	1 (12.5%)	4 (50.0%)
gimp-user	21 (80.8%)	2 (7.7%)	3 (11.5%)	10 (76.9%)	3 (23.1%)	0 (0.0%)
gimp-developer	5 (62.5%)	3 (37.5%)	0 (0.0%)	2 (100.0%)	0 (0.0%)	0 (0.0%)
pgsql-general	68 (65.4%)	22 (21.2%)	14 (13.5%)	29 (65.9%)	12 (27.3%)	3 (6.8%)
pgsql-hackers	20 (83.3%)	2 (8.3%)	2 (8.3%)	8 (66.7%)	3 (25.0%)	1 (8.3%)
svn-users	117 (58.2%)	28 (13.9%)	56 (27.9%)	33 (47.1%)	10 (14.3%)	27 (38.6%)
svn-dev	14 (56.0%)	8 (32.0%)	3 (12.0%)	6 (50.0%)	3 (25.0%)	3 (25.0%)
TOTAL	**291 (63.5%)**	**68 (14.8%)**	**99 (21.6%)**	**101 (54.6%)**	**33 (17.8%)**	**51 (27.6%)**

5.9 Effects of Newbie Gender

Gender is an interesting factor on these mailing lists, given the large ratio of males to females and the potential effects this might have. Would males see open source software as their domain and show negative sexism against the female newbies? Or oppositely, would they make a special effort to treat females better since they are so rare? Something to keep in mind in investigating this issue is that the sample of females was very small. Therefore, instead of the chi-square test for determining independence of factors, we use Fisher's exact test. Fisher's exact test is more accurate for small samples than the chi-square test (McDonald, 2008). Examining the helpfulness of replies to male versus female newbies, no distinction could be made (p-value = .765).

Table 18. Helpfulness of replies by newbies' gender. Omits newbies of unknown gender.

Mailing List	Male Newbies (97.3%)				Female Newbie (2.7%)		
	Helpful Replies	Non-helpful Replies	Total		Helpful Replies	Non-helpful Replies	Total
mediawiki-l	35 (74.5%)	12 (25.5%)	47		3 (60.0%)	2 (40.0%)	5
wikitech-l	5 (38.5%)	8 (61.5%)	13		0	0	0
gimp-user	22 (53.7%)	19 (46.3%)	41		1 (100.0%)	0 (0.0%)	1
gimp-developer	0 (0.0%)	2 (100.0%)	2		0	0	0
pgsql-general	94 (80.3%)	23 (19.7%)	117		1 (50.0%)	1 (50.0%)	2
pgsql-hackers	23 (76.7%)	7 (23.3%)	30		1 (100.0%)	0 (0.0%)	1
svn-users	138 (70.8%)	57 (29.2%)	195		2 (100.0%)	0 (0.0%)	2
svn-dev	19 (67.9%)	9 (32.1%)	28		2 (100.0%)	0 (0.0%)	2
TOTAL	**336 (69.1%)**	**137 (28.2%)**	**473**		**10 (76.9%)**	**3 (23.1%)**	**13**

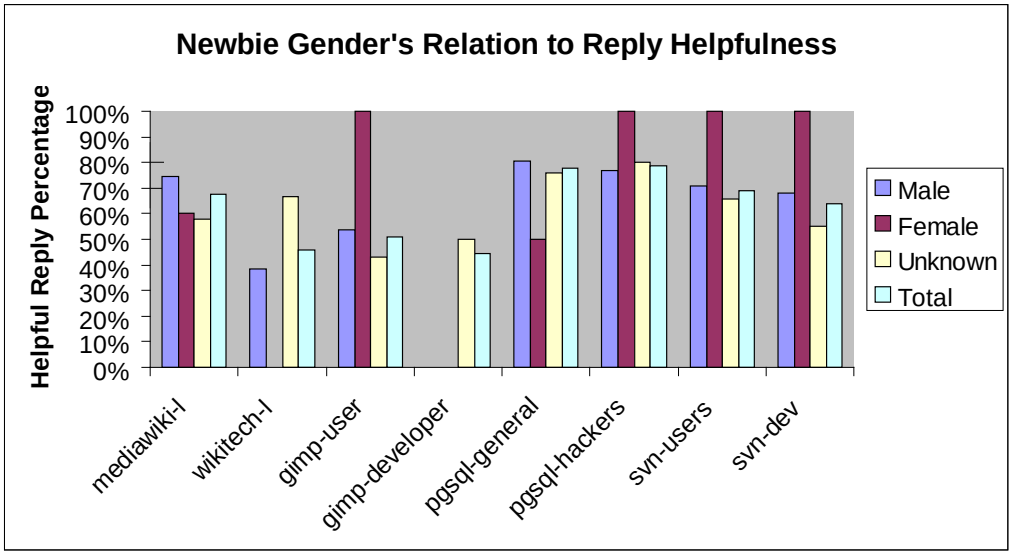

Figure 6. Helpfulness of replies by newbies' gender.

Furthermore, examining the tone of replies to male versus female newbies, no distinction could be made either (p-value = 1). Therefore, this study found no benefit or drawback to appearing to be a female newbie on an open source software list regarding courtesy.

Table 19. Tone of replies by newbies' gender. Omits 3 rude replies and newbies of unknown gender.

Mailing List	Male Newbies (97.3%)				Female Newbies (2.7%)		
	Polite Replies	**Neutral Replies**	**Total**		**Polite Replies**	**Neutral Replies**	**Total**
mediawiki-l	5 (10.6%)	42 (89.4%)	47		0 (0.0%)	5 (100.0%)	5
wikitech-l	3 (23.1%)	10 (76.9%)	13		0	0	0
gimp-user	4 (10.0%)	36 (90.0%)	40		1 (100.0%)	0 (0.0%)	1
gimp-developer	0 (0.0%)	2 (20.0%)	2		0	0	0
pgsql-general	8 (6.8%)	109 (93.2%)	117		0 (0.0%)	2 (100.0%)	2
pgsql-hackers	4 (13.8%)	25 (86.2%)	29		0 (0.0%)	1 (100.0%)	1
svn-users	8 (4.1%)	186 (95.9%)	194		0 (0.0%)	2 (100.0%)	2
svn-dev	4 (14.3%)	24 (85.7%)	28		0 (0.0%)	2 (100.0%)	2
TOTAL	**36 (7.7%)**	**434 (92.3%)**	**470**		**1 (7.7%)**	**12 (92.3%)**	**13**

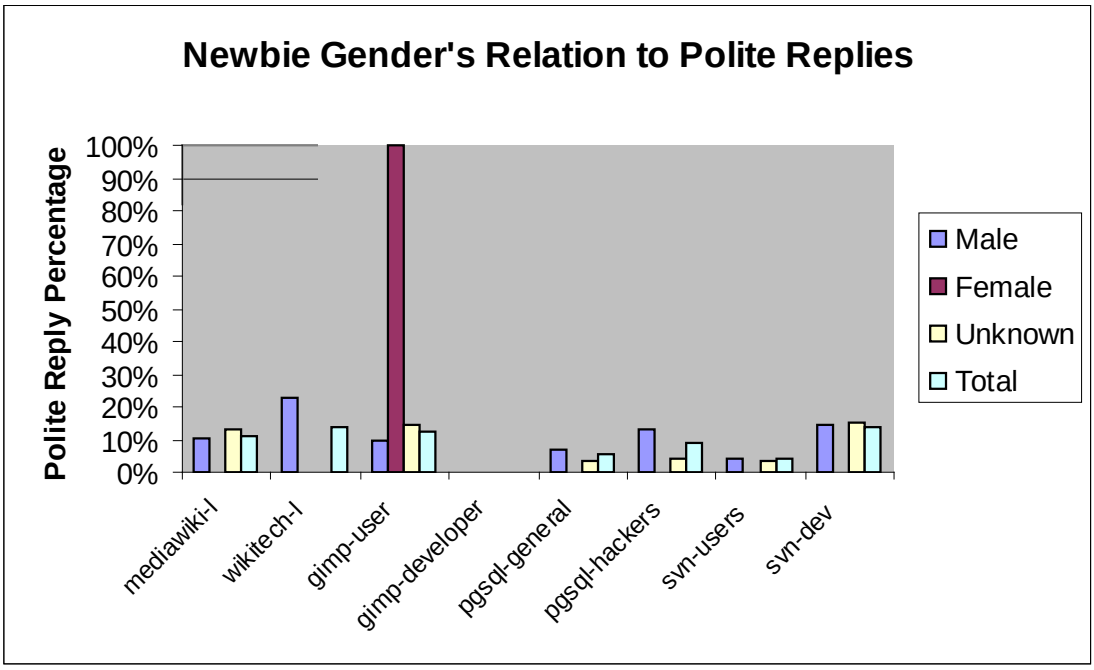

Figure 7. Polite reply ratio by newbies' gender.

While the raw data shows a smaller percentage of females as receiving replies than males, the difference was not statistically significant, in part likely due to the small female sample size (Fisher's exact test, p-value = .330).

Table 20. Time elapsed before newbies received a reply to their post, by gender. Omits newbies of unknown gender, but not rude replies.

Mailing List	Male Newbies (96.4%)			Female Newbies (3.6%)		
	24 Hours or Less	**More Than 24 Hours**	**No Reply**	**24 Hours or Less**	**More Than 24 Hours**	**No Reply**
mediawiki-l	29 (64.4%)	3 (6.7%)	13 (28.9%)	2 (33.3%)	1 (16.7%)	3 (50.0%)
wikitech-l	8 (53.3%)	1 (6.7%)	6 (40.0%)	0	0	0
gimp-user	22 (81.5%)	3 (11.1%)	2 (7.4%)	1 (100.0%)	0 (0.0%)	0 (0.0%)
gimp-developer	0 (0.0%)	1 (100.0%)	0 (0.0%)	0	0	0
pgsql-general	54 (73.0%)	14 (18.9%)	6 (8.1%)	1 (100.0%)	0 (0.0%)	0 (0.0%)
pgsql-hackers	16 (84.2%)	2 (10.5%)	1 (5.3%)	0 (0.0%)	1 (100.0%)	0 (0.0%)
svn-users	91 (53.5%)	26 (15.3%)	53 (31.2%)	2 (50.0%)	0 (0.0%)	2 (50.0%)
svn-dev	11 (50.0%)	7 (31.8%)	4 (18.2%)	1 (100.0%)	0 (0.0%)	0 (0.0%)
TOTAL	**231 (61.9%)**	**57 (15.3%)**	**85 (22.8%)**	**7 (50.0%)**	**2 (14.3%)**	**5 (35.7%)**

6 Discussion

Though not emphasized in this study, most newbies posted on the mailing lists seeking help with a problem with the software, and did not necessarily have any interest in following the project long-term. With this caveat in mind, one question raised by this study is of how long a person needs to contribute to an OSS project before he or she can be considered to have joined or become a regular. Does contributing a single bug fix, but nothing else, qualify someone as having successfully joined the project? What about a few mailing list comments over the course of a week? The line is quite vague and subjective. Therefore, when we considered people as having successfully become part of the mailing list community by posting 3-6 months after their initial post, and again 6-9 months after their initial post, we cannot claim that this is a definitive answer to the question. It is a valid argument that a person who stops by the list for a month and posts some insightful messages has become a part of the community in a more meaningful way than someone who makes a less-useful comment every few months. However, it is reasonable to assume that those still asking questions and participating in discussions after 6 months probably have overcome the initial hurdles of installation and configuration, and therefore are engaging in more meaningful discussion.

This study did not consider whether or not the newbies actually ended up becoming developers or submitted bug reports. This information would be interesting for future investigation. It is possible that even those newbies who did not post again on the mailing lists during the March-August intervals we studied were silently lurking, and/or submitting code or bug reports to the project.

When considering the results, it is important to keep in mind that only newbie-initiated threads were examined. The possibilities exist that participants known in the mailing list communities would receive more replies, more polite replies, and/or more helpful replies. On the other hand, it is also possible that newbies are the ones who receive more since the communities might make special efforts to encourage newcomers. Such information about mailing lists in general is outside of this study's scope.

Mediocre rater agreement on the tone/politeness category could have affected the results. An interesting observation was made with respect to the issue of how newbie tone related to the number of replies received. Considering all lists, there was no relation found ($\chi 2 = .08$, p-value = .777). However, if we consider only the lists with the top 3 values for rater agreement for tone

(wikitech-l, svn-users, and pgsql-general), a somewhat different result is found ($\chi 2 = .496$, p-value = .481). One reason for the low rater agreement for tone is likely the diverse ethnicities of the raters. Two raters were raised in Korea, one in China, and a plurality in the United States. Since different cultures have differing standards of courtesy, the diversity created an extra obstacle for the raters in being synchronized, despite training and discussion periods. Another problem for agreement was probably the high subjectivity of the tone category itself.

This study indicates that three to four years after the David et al.'s 2003 FLOSS Survey, females are likely still a tiny part of the open source community, representing less than 2%. People participating in open source software projects are helpful to newcomers, at least on the mailing lists. Flames, putdowns, and elitism are likely rare, although this could depend on the specific project. Blatant rudeness ignored, a newbie's degree of courtesy in his or her initial post had little effect on the replies in this study. Only the helpfulness of the replies was slightly affected. Taken as a whole, this study suggests that newbies on OSS mailing lists do not need to fear the community and can expect a positive experience. Again, though, there may be projects that exist with different online cultures than the four examined. Also, there might be more aggression and rudeness among non-first-time posters that newbies could pick up on by lurking on the lists. Thus, while OSS participants were generally polite to newbies, it is possible that newbie expectations and perceptions of politeness could be colored by how the regulars engage with each other.

7 Conclusion

Out of the 643 newbies who initiated threads on the list, only 87 (13.5%) were active on the mailing list after a 3-6 months gap. Even fewer (6.4% of the original 643) were active after 6-9 months. The majority of newbies posted their question/issue and then were not heard from again, although it is possible they continued to lurk.

Most of the replies that newbies got to their first post were helpful (69.3%). PostgreSQL and Subversion, the developer-oriented projects, had a significantly greater proportion of helpful mailing list replies than did the other projects. No significant difference was found between user and developer lists in the proportion of helpful replies, though.

Aggression and rudeness towards newbies was rare on the lists analyzed. Polite replies were uncommon, with only 6.8% being rated as polite. However, an overwhelming 91.8% majority was neutral in tone, and only 1.5% were rude. Contrary to notions of developer elitism, newbies on the developer lists received significantly more polite replies on average than those on the user lists. On the other hand, newbies posting on lists for non-developer projects received significantly more replies on average than those posting on developer project lists.

Most newbies received replies to their posts within 24 hours. 14.3% received their first reply between one day and one week after their post, and 1.4% received the first reply after one week. Only 23.3% of newbies did not receive any replies on the list, although newbies posting on user lists were significantly less likely to receive a reply than those posting on developer lists.

In a blow to the notion of rampant elitism on OSS mailing lists, the top 10 percent of active posters on the lists, defined as core members for this study, responded equally to newbies and non-newbies. Posting on a user list as opposed to a developer list made no significant difference in whether a newbie received a reply from a core member.

Newbies' politeness was surprisingly found to have minimal effect in eliciting replies. Polite newbies received a greater percentage of helpful replies than neutral-tone newbies, but only with marginal statistical significance (p-value = .081). The politeness of replies to a newbie's post was found to be independent of the politeness of the newbie's post, as was the newbie's likelihood of receiving a reply. In the data analysis, rude posts were omitted due to their incredibly low occurrence.

The nationality of newbies was not found to have a significant effect on the replies in the areas we analyzed. Between the US/unknown-nationality group and the non-US group, reply helpfulness, reply tone, and the occurrence of not receiving responses were statistically indistinguishable.

As with nationality, gender was not found to have a significant effect on the replies, although low female participation should be taken into account. Reply helpfulness, reply tone, and the chance of not receiving a reply were not found to be related to newbie gender.

Bibliography

About.com. (2008). *Top German Names for Boys and Girls*. Retrieved April 25, 2008 from http://german.about.com/library/blname_topDE.htm

Algorithms and Theory of Computation Handbook, CRC Press LLC, 1999, "Levenshtein distance", in Dictionary of Algorithms and Data Structures [online], Paul E. Black, ed., U.S. National Institute of Standards and Technology. 14 August 2008. Retrieved April 9, 2008 from http://www.nist.gov/dads/HTML/Levenshtein.html

Apache Software Foundation. (2009). *Mailing Lists*. Retrieved March 18, 2009 from http://www.apache.org/foundation/mailinglists.html

Berlin, L. (1992). Beyond Program Understanding: A Look at Programming Expertise in Industry. Retrieved April 2008 from http://www.hpl.hp.com/techreports/92/HPL-92-142.pdf

Bird, et al. Mining email social networks. *Proceedings of the 2006 International Workshop on Mining Software Repositories*, 28th International Conference on Software Engineering, Shanghai, China, 2006, 137-143.

Brooks, F. (1975). *The Mythical Man-Month*. Reading, MA: Addison-Wesley.

Crocker, D. (1982). *RFC822: Standard for ARPA Internet Text Messages*. Retrieved April 7, 2008 from http://www.w3.org/Protocols/rfc822/

Crowston, K. and Howison, J. The social structure of Free and Open Source software development. *First Monday 10*, 2 (2005), http://firstmonday.org/issues/issue10_2/crowston/index.html

David, P., Waterman, A., and Arora, S. (2003). *The Free/Libre/Open Source Software Survey for 2003*. Retrieved May 14, 2008 from http://www.stanford.edu/group/floss-us/

Free Software Foundation, Inc. (2007). *The Free Software Definition*. Retrieved April 9, 2008 from http://www.gnu.org/philosophy/free-sw.html

Gacek, C. and Arief, B. The many meanings of open source. *IEEE Software 21*, 1 (2004), 34-40.

Garson, G. D. (2009). *Logistic Regression*. Retrieved March 24, 2009 from http://faculty.chass.ncsu.edu/garson/PA765/logistic.htm

Ghosh, et al. (2006). Economic impact of FLOSS on innovation and competitiveness of the EU ICT sector. Retrieved April 2008 from http://ec.europa.eu/enterprise/ict/policy/doc/2006-11-20-flossimpact.pdf

Gutwin, C., Penner, R, Schneider, K. Group Awareness in Distributed Software Development. *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*, Chicago, Illinois, 2004, 72-81.

Halloran, T. and Scherlis, W. High Quality and Open Source Software Practices. *Meeting Challenges and Surviving Success: The 2nd Workshop on Open Source Software Engineering*, 24th International Conference on Software Engineering, Orlando, Florida, 2002.

Herraiz et al. The Processes of Joining in Global Distributed Software Projects. *Proceedings of the 2006 international workshop on Global software development for the practitioner*, 28th International Conference on Software Engineering, Shanghai, China, 2006, 27-33

Internet Corporation for Assigned Names and Numbers. (Date Unknown). *Root Zone Database*. Retrieved February 2008 from http://www.iana.org/domains/root/db/index.html

Jensen, C. and Scacchi, W. Role Migration and Advancement Processes in OSSD Projects: A Comparative Case Study. *Proceedings of the 29th International Conference on Software Engineering*, Minneapolis, Minnesota, 2007, 364-374.

Keizer, G. (2008). *Firefox on track to crack 20% share in July*. Retrieved July 2008 from http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=9091959&intsrc=hm_list

Ko, A., DeLine, R., and Venolia, G. Information Needs in Collocated Software Development Teams. *Proceedings of the 29th International Conference on Software Engineering*, Minneapolis, Minnesota, 2007, 344-353.

Krishnamurthy, S. Cave or Community?: An Empirical Examination of 100 Mature Open Source Projects. *First Monday, 2002*, Available at SSRN: http://ssrn.com/abstract=667402

Krogh, G., Spaeth, S., and Lakhani, K. Community, joining, and specialization in open source software innovation: a case study. *Research Policy 32*, 7 (2003), 1217-1241.

Kropla, S. (2006). *International Dialing Codes*. Retrieved February 2008 from http://www.kropla.com/dialcode.htm

Kuchling, A. M. (2006). *JWZ's Threading Algorithm*. Retrieved February 28, 2008 from http://www.amk.ca/python/code/jwz

Lakhani, K. and von Hippel, E. How open source software works: "free" user-to-user assistance. *Research Policy 32*, 6 (2003), 923-943.

Lampe, C. and Johnston, E. Follow the (Slash) dot: Effects of Feedback on New Members in an Online Community. *Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work*, Sanibel Island, Florida, 2005, 11-20.

Lewis, D. and Knowles, K. Threading Electronic Mail: A Preliminary Study. *Information Processing & Management 33*, 2 (1997), 209-217.

McDonald, J. H. (2008). *Chi-square test of independence*. Retrieved March 3, 2009 from http://udel.edu/~mcdonald/statchiind.html

McDonald, J. H. (2008). *Fisher's exact test of independence*. Retrieved March 3, 2009 from http://udel.edu/~mcdonald/statfishers.html

Netcraft. (2008). *January 2008 Web Server Survey*. Retrieved April 9, 2008 from http://news.netcraft.com/archives/2008/01/28/january_2008_web_server_survey.html

Nuvolari, A. Open source software development: Some historical perspectives. *First Monday 10*, 10 (2005), http://firstmonday.org/issues/issue10_10/nuvolari/index.html

OpenOffice.org. (2007). *Stats Project*. Retrieved March 18, 2009 from http://stats.openoffice.org/index.html

Open Source Initiative. (2006) *The Open Source Definition*. Retrieved April 9, 2008 from http://www.opensource.org/docs/osd

Paul, R. (2008). *All the rage in Europe: Firefox market share climbs higher*. Retrieved July 2008 from http://arstechnica.com/news.ars/post/20080427-all-the-rage-in-europe-firefox-marketshare-climbs-higher.html

Pigoski, T. M. and Looney, C. S. Software maintenance training: Transition experiences. *Proceedings of the 1993 Conference on Software Maintenance*, Montreal, Canada, 1993, 314-318.

Qmail Documentation. (1998). *mbox*. Retrieved April 9, 2008 from http://www.qmail.org/man/man5/mbox.html

Raymond, E. The Cathedral and the Bazaar. *Knowledge, Technology, and Policy 12*, 3 (1999), 23-49.

Sim, S. and Holt, R. The Ramp-Up Problem in Software Projects: A Case Study of How Software Immigrants Naturalize. *Proceedings of the 20th International Conference on Software Engineering*, Kyoto, Japan, 1998, 361-370.

Sowe, S., Stamelos, I., and Angelis, L. Identifying knowledge brokers that yield software engineering knowledge in OSS projects. *Information and Software Technology 48*, 11 (2006), 1025-1033.

Stallman, R. (2007). *Linux and the GNU Project*. Retrieved April 9, 2008 from http://www.gnu.org/gnu/linux-and-gnu.html

Stallman, R. (2007). *Why "Open Source" misses the point of Free Software*. Retrieved April 9, 2008 from http://www.gnu.org/philosophy/open-source-misses-the-point.html

Stallman, R. (2008). *Richard Stallman's Personal Home Page*. Retrieved April 9, 2008 from http://www.stallman.org

Tiemann, M. (2006). *History of the OSI*. Retrieved April 9, 2008 from http://www.opensource.org/history

U.S. Census Bureau. (2008). *Genealogy*. Retrieved April 25, 2008 from http://www.census.gov/genealogy/names/names_files.html

Wheeler, D. (2007). Why *Open Source Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers!* Retrieved March 18, 2009 from http://www.dwheeler.com/oss_fs_why.html

Yamauchi et al. Collaboration with Lean Media: how open-source software succeeds. *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work*, Philadelphia, Pennsylvania, 2000, 329-338.

Ye, Y. and Kishida, K. Toward an understanding of the motivation of open source software developers. *Proceedings of the 25th International Conference on Software Engineering*, Portland, Oregon, 2003, 419-429.

Zawinski, J. (2002). *Message Threading*. Retrieved February 28, 2008 from http://www.jwz.org/doc/threading.html