SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

Special Focus

August 2013, Vol. 56 ******:1–******:13
doi: **************

How commercial involvement affects open source projects: three case studies on issue reporting

MA XiuJuan[1,2], ZHOU MingHui[1,2]* & RIEHLE Dirk[3]

[1]*School of Electronics Engineering and Computer Science, Peking University, Beijing* 100871*, China;*
[2]*Key Laboratory of High Confidence Software Technologies, Ministry of Education, Beijing* 100871*, China;*
[3]*Friedrich-Alexander-University of Erlangen-Nurnberg, Martensstr. 3,* 91058*, Erlangen, Germany*

Received March 5, 2013; accepted June 28, 2013

Abstract Whereas most research on Internetware has focused on new technologies for keeping track of a changing Internet, little attention has been paid to the software development process. A large portion of the software running the Internet is open source software. Open source software is developed both by volunteers and commercial companies, often jointly. Companies get involved in open source projects for commercial reasons, and bring with them a commercial software development process. Thus, it is important to understand how commercial involvement affects the software development process of open source projects. This article presents case studies of three open source application servers that are being developed jointly by a volunteer community and one primary software company. We are interested in better understanding developer behavior, specifically task distribution and performance, based on whether the developer is an external contributor, e.g., a volunteer working in their spare time, or a commercial developer from inside the primary backing company who is being paid for their time. To achieve this, we look at issue reporting as an example of commercial involvement in open source projects. In particular, we investigate the distribution of tasks among volunteers and commercial developers by studying the source of reported issues and quantify the task performance on user experience via the issue resolution speed. We construct measures based on historical records in issue tracking repositories. Our results show that, with intensified commercial involvement, the majority of issue reporting tasks would be undertaken by commercial developers, and issue resolution time would be reduced, implying a better user experience. We hope our methods and results provide practical insights for designing an efficient hybrid development process in the Internetware environment.

Keywords open source software development, commercial involvement, hybrid development, contributor participation, issue reporting

Citation Ma X J, Zhou M H, Riehle D. How commercial involvement affects open source projects: three case studies on issue reporting. Sci China Inf Sci, 2013, 56: ******(13), doi: **************

1 Introduction

The successful development of several extraordinary open source software (OSS) systems, such as Apache HTTP Server and Linux Kernel, has demonstrated the potential for constructing software by mostly voluntary contributions distributed across the world. Participants in OSS development work together

*Corresponding author (email: zhmh@pku.edu.cn) through software repositories, such as issue tracking systems, running over the Internet, and cooperate to resolve issues and produce good quality software. Over the past two decades, more and more companies have begun to build various business models around OSS projects in the expectation of benefiting from both open source and commercial development [1]. In OSS-commercial hybrid projects [2], participants from both commercial companies and the wider Internet community collaborate together to construct innovative and high-quality software.

However, it is a challenge to lead hybrid development efficiently. The involvement of a company, particularly, its for-profit business interests, and the formal way in which they arrange employees and the development process, may have an unpredictable impact on the behavior of OSS contributors, e.g., changing the task distribution and performance, attracting or hindering new comers. For example, "there were many rumors about various people leaving JBoss after its acquisition by Red Hat, including Marc Fleury [the founder]"1), which would constitute a great loss to the project. This inspired us to study how commercial involvement affects contributors' participation in hybrid development practices. In particular, we address whether the task preference of *external* participants (e.g., external users, participants voluntarily contributing in their spare time) changes because of the presence of *internal* developers (who are from inside the backing company, working on behalf of the commercial entity). We also consider whether the task performance would be improved, e.g., would user experience receive more attention and improve as a result. These issues have been subjected to only limited research in the academic community.

Specifically, as suggested by Linus' Law: "Given enough eyeballs, all bugs are shallow" [3], user issue reports have commonly been regarded as a crucial contribution in OSS development [4]. These user reports, which are also the source of the user innovation generally expected in most software production [5], will help to ensure and improve the functionality and quality of software. Hence, in particular, we look at issue reporting as an example of commercial involvement in open source projects, and investigate the following questions:

1) Is the task of issue reporting most commonly undertaken by external participants or internal developers?

2) Are external user issue reports resolved more quickly than developers' reports?

There are various ways for companies to engage in hybrid projects. For example, as discussed in [6], in many present open source projects (e.g., MySQL, Alfresco, SugarCRM, Mulesoft, Jaspersoft, Pentaho), a single company performs all the development, similar to the case of Google with Android. The classification of *internal* and *external* participants in such projects might have less significance, as all the development tasks are undertaken by *internal* developers. However, this study aims to highlight the problems faced in another typical class of hybrid projects, where the OSS and commercial software development are combined and expected to form a new and more efficient software engineering approach, as illustrated in [2].

Answers to these questions can help us understand the task preference and the effect on user experience in practice, which will help to improve methods and tools for hybrid development. The open, dynamic and uncontrollable nature of the Internet has led to a new software paradigm, i.e., Internetware [7,8]. Thus, we believe that understanding hybrid approaches will help to determine the best practice for Internetware development. Indeed, software that presents the typical characteristics of Internetware, e.g., an application server, may also be implemented in this way.

Therefore, we conduct an empirical study of three large OSS-commercial hybrid projects developing JavaEE application servers: JBossAS, Apache Geronimo, and JOnAS. These are the same projects we selected in a previous study on Internetware development [9]. We follow an experimental procedure of data filtering similar to that of the Apache study [10]: retrieving and processing historical data from issue tracking repositories, and constructing quantitative measures based on the extracted data to answer our specific research questions. Moreover, we investigate the three projects at the granularity of "epochs", representing time intervals over which the nature of the commercial involvement and business goals are relatively stable. In JBossAS and Geronimo, the epochs represent the different extent of company support, whereas in JOnAS they reflect different technical requirements and, therefore, different business

1) http://www.theserverside.com/discussions/thread.tss?thread_id=43410.

strategies. JOnAS is an excellent contrast to the other two projects because, although implementing the same functionality, it started as a hybrid, and thus may shed light on how much the initial conditions affect hybrid project practices.

Our results show that, with intensified commercial involvement, the majority of issues are reported by internal developers, and user issues are resolved more quickly than internal issues. This implies that the user experience has been enhanced. The results suggest that the current issue tracking systems in these projects can be improved in several ways, e.g., setting some fields in issue reports to highlight user issues, or improving the issue resolution process by adding special triage steps.

This paper makes the following main contributions.

1) Hybrid projects and issue-reporting practices, such as survival curves to quantify issue resolution speed.

2) We quantify the influence of commercial involvement on issue reporting in OSS projects. For example, commercial involvement is associated with a shift in the majority of issue reporters from external users to internal developers. User reports will thus be resolved more quickly than developer reports in some projects, whereas in other projects the opposite is the case.

3) We provide practical insights into the methods and tools that may help to design an efficient hybrid development with a high level of user experience.

The rest of the paper is organized as follows. Section 2 introduces some related work, and Section 3 describes the context of the three projects and illustrates our data filtering approach. Section 4 presents the results of the three case studies, and we discuss the limitations of our work in Section 5. Finally, we present our conclusion in Section 6.

2 Related work

Whereas there has been considerable research into OSS development, little has focused specifically on hybrid projects and the impact of commercial involvement in issue reporting.

Contributors' participation in OSS development has been studied by many researchers. For example, Refs. [11–13] investigated why the participants in OSS contribute without material compensation, and how such apparently unstructured and distributed organizations survive and succeed. Von Krogh et al. [14] studied the strategies and processes by which newcomers join the OSS community, while Zhou et al. [15] observed the impact of the project environment on newcomers' participation. Our work is most closely related to the study of Apache web server and Mozilla web browser [2]. This quantified aspects of OSS development practices, such as developer participation, issue reporters, and issue resolution speed, using email archives of the source code change history and problem reports. The results were framed as seven hypotheses that outline key aspects of OSS development. We have also replicated a case study of the Apache study on Internetware development [9]. In this paper, we confirmed some of the hypotheses on issue reporting in [2] in the context of hybrid projects, and also found some substantial differences. The new context of hybrid projects required us to discriminate between volunteer and commercial participation. In addition, the dynamic nature of commercial involvement and project strategies, combined with a longer (than in the Apache study) time frame and required us to separate each project into separate epochs and introduce other methodological innovations.

In the Apache study [10], the authors also suggested that OSS practices may be successfully combined with commercial practices in hybrid projects. However, the quantification of commercial influence is rarely addressed. Current research focuses mainly on theoretical models for hybrids, e.g., how projects may benefit from the coordination mechanism of OSS [16], how the stakeholders make economic profit from OSS [17], and how to define an innovative software engineering paradigm for large corporations [18]. There are also some empirical studies on successful hybrid projects. These enable us to understand how to improve upon existing software development practices, e.g., the analysis of business strategies [1,19] and development methodology [20]. In particular, Wagstrom et al. [21] provided an in-depth analysis of commercial involvement in the GNOME and Eclipse software projects, and measured the impact of two kinds of commercial involvement, namely community-focused and product-focused, on the participation of volunteers. Unlike in prior work, we focus on the impact of commercial involvement on contributors' participation in the practice of issue reporting.

Moreover, there are several studies that consider the issue reporting practice in other contexts. Hendry [22] analyzed the role of users in software development, and showed that many users offered creative input according to personal experience, scenario, and observed use. Ko et al. [23] investigated the interaction between users and developers of Mozilla, and found that reports that did lead to changes were made by a relatively small group of experienced, frequent reporters. In our study, we further research the role of users in the context of hybrid development, which requires us to introduce methodological innovations.

3 Methodology

We employ a quantitative approach to analyze the impact of commercial involvement on contributors' participation in issue reporting. We first present the context of the projects in Subsection 3.1, and the quantitative approach in Subsection 3.2.

3.1 Context

The three projects we investigated, JBossAS, Apache Geronimo, and JOnAS, started around 2000 and remain active. Currently, JBossAS is hosted by Red Hat and is the most popular open source application server (AS). Geronimo is an Apache project that is heavily supported by IBM. JOnAS is a development of the OW2 Consortium, which was established by the collaboration of Bull and other organizations.

There are three reasons we selected these projects. First, all three projects are ASs that have been developed over a similar time period. This allows us to control for external factors such as the application domain and changes in the technology landscape and the world economy. Second, as companies have changed the way they shape the community over the last decade, allowing us to observe the effect of such changes within a single project, thus controlling for project context. Actually, we can isolate nine epochs over which community strategies are relatively stable for the three projects. Finally, we have several years of experience working on ASs. The first two authors participated in the development of JOnAS, with one being on the community council since 2007. This means we are intimately aware of AS technology and how it has changed over time.

To make comparisons, we first determine the scope of each project in Subsection 3.1.1, divide the time-line of each project into relatively homogeneous periods (epochs), as described in Subsection 3.1.2, and then introduce the business strategies of each of the companies involved in the three projects in Subsection 3.1.3.

3.1.1 *Scope*

Each AS conforms to the standard specifications of JavaEE AS. In the history of JavaEE AS, there have been four remarkable evolutions in J2EE technology, namely the "Initial stage (J2EE 1.3 specification)", "J2EE 1.4 specification", "JavaEE 5 specification" and "Architecture Refactoring". The latter two started at a similar time, so we group them into a single "JavaEE 5 and Architecture Refactoring epoch". New specifications, with new functionalities and the effect of simplifying development, have been published and implemented. Meanwhile, since the end of 2006, great effort has been made to refactor the increasingly large product to achieve a loose coupling of AS components (see, e.g., [24]). Over time, all three projects grew in size organically and by incorporating other projects. To ensure that we compare similar functionalities in the JOnAS repository with JBossAS and Geronimo projects, we follow the suggestion of a JOnAS core team member, and consider a combination of repositories for "Carol", "CMI", "Easybeans", "JASMINe", "Jotm", and "JOnAS" to constitute the JOnAS project.

3.1.2 *Epochs*

During their extended lifespan, the three projects have encountered significant changes in the nature of commercial involvement (e.g, the change from open source to commercial, from one kind of company

Table 1 Epochs in the three projects

		Time span	JEE V	Backing cmpny.	Business strategy
JBossAS	Epoch 1	1999.10–2001.03	1.3	Open source	
	Epoch 2	2001.06–2004.02	1.3	JBoss Group LLC.	Offering professional services
	Epoch 3	2004.06–2006.02	1.4	JBoss Inc.	Service Business
	Epoch 4	2006.10–2010.09	5	Red Hat	Service Business
Geronimo	Epoch 1	2003.08–2005.04	1.4	open source	
	Epoch 2	2005.08–2010.09	1.4, 5	IBM	Expert Technical Support for Geronimo
JOnAS	Epoch 1	1999.10–2004.03	1.3	Bull	Initial application server
	Epoch 2	2004.06–2006.10	1.4	Bull	Introduction of JOnAS appliance
	Epoch 3	2007.02–2010.09	5	Bull	Introduction of a new generation of JOnAS

to another, or changes in company strategy). Therefore, we divide the time-line of each project into relatively homogeneous periods, from which we can draw comparisons to observe the impact of commercial involvement.

JOnAS was initiated as a hybrid project, with Bull as its primary backer from the beginning. It experienced the above-mentioned changes in JavaEE technology, which changes might have caused the strategy changes described below.

The time-line of JBossAS and Geronimo was divided according to the time at which commercial backing commenced. Both projects started as OSS and received company support a few years later. JBossAS has experienced three changes. The JBoss Group LLC was founded in April 2001. Then in March of 2004, JBoss Inc. was founded, before Red Hat acquired JBoss Inc. in April of 2006. Geronimo started receiving support from IBM in May of 2005.

It is worth noting that the changes in J2EE technology are somewhat similar, and possibly related to changes in commercial involvement. However, the dates of technology changes are not very precise, and the project behavior cannot react immediately to a change of commercial involvement. Hence, we exclude a slice of time immediately prior and just after these change points to reduce potential noise introduced by the transition. For example, prior to and after the acquisition, some activities that are not relevant to normal project operations might add unwanted noise. We use SVN commit logs and news on each project's website to identify and understand these transitions. As a result, for JBossAS, we ignore two months during the foundation of JBoss Group LLC, three months during the foundation of JBoss Inc., and seven months during the acquisition by Red Hat. For Geronimo, we ignore three months during the acquisition by IBM. For JOnAS, we ignore three months at the boundary between the two epochs.

Therefore, we have a total of nine epochs. Of these, seven belong to some kind of hybrid project, and two belong to pure OSS projects. The specific time spans are shown in the second column of Table 1.

3.1.3 *Strategies of commercial participation*

By comparing epochs within and among projects, we observe differences in development practices with different types of commercial involvement. Each change in commercial involvement also involves a specific shift in strategy. These strategies are determined by the relevant business requirements and affect development practices, e.g., issue reporting.

The strategy in JBossAS can be distinguished by the requirements to offer professional services and build a "Professional Open Source" business model[2]. In Epoch II, JBoss Group LLC aimed to provide support and services for JBossAS server technology directly from the core developers, and generate revenues that rewarded its employees and, most importantly, JBoss' developers[3]. In Epoch III, the actions of JBoss Inc. are reflected by the employment of leading JBoss developers. In Epoch IV, Red Hat acquired JBoss Inc. to ensure the continuity of an independent open source AS and integrated it as

2) http://www.jboss.com/pdf/dhbrown0704.pdf.

3) http://web.archive.org/web/20031206230933/, http://www.jbossgroup.com/index.html?module=html&op=userdisplay &id=en/company/index.

a division of Red Hat[4]. Both JBoss Inc. and Red Hat sell services, such as, production and development support, patches and updates, multi-year maintenance policies, and software assurance[5], but Red Hat appears to be more engaged in the commercial dissemination. For example, to provide marketing and support for JBossAS, Red Hat has organized an annual JBoss World conference since 2005. This is now co-hosted with the Red Hat Summit, and attracts numerous participants.

IBM provides expert technical support for Apache Geronimo[6]. IBM works continuously with the community to ensure that new features are added to Geronimo for the advancement of community goals. Additionally, all Geronimo bug fixes made by IBM are contributed to the Apache Geronimo community[7]. IBM is an active participant in the Apache Geronimo community—of 63 committers listed on the Geronimo website, more than half work for IBM[8]. Geronimo is used by IBM to test and obtain innovations that are transferable to IBM's commercial products. For example, based on Geronimo, IBM announced WAS CE, a free edition of its WebSphere application server[9].

Bull uses JOnAS as an open source enterprise middleware solution for its customers[10]. Bull offers customers advanced professional support for JOnAS, in addition to free online support[11]. Over its three epochs, the strategy behind JOnAS has evolved over time, based mainly based on the changes in J2EE technology. In the first epoch, i.e., the "Initial Epoch", J2EE technology emerged and thrived. The JOnAS team focused on developing an early version through cooperation with several French organizations, including France Telecom, Lifl, and INRIA[12]. In the second "J2EE 1.4 Specification Epoch", the namesake specification was implemented while cooperating with those organizations. Furthermore, Bull appears to be effective at dissemination, because many third-party developers have started to build applications and products based on JOnAS, such as "NeoLoad" and "Liferay Portal"[13]. In the "JavaEE 5 and Architecture Refactoring Epoch", the JOnAS strategy shifted to the development of an innovative version conforming to new specifications based on OSGi™[14], as well as cooperation with many other organizations to broaden its influence; e.g., SerLi, University of Fortaleza, and Peking University[15], contributed to the development of JOnAS 5.x in this epoch.

Furthermore, when a company participates in a hybrid project, it generally conducts software development in the manner to which it is accustomed. Although volunteer participation is likely to be highly appreciated, companies are usually inclined to transfer their existing developers and practises to these projects. For example, the three projects assign responsible owners for projects and sub-projects in their JIRA[16] issue—tracking systems, which implies that the task ownership is enforced. In JBossAS, the issue tracking system and community forum have played different roles since JBoss Inc. was founded. After Red Hat acquired JBoss, JIRA was no longer "the place to ask for help", while the forum became more active place in terms of users who wished to "ask a question" or have a discussion[17]. Earlier, it was more common to report issues through the issue tracking system.

Table 1 summarizes the types (3rd column) and strategies (last column) of the three projects over their different epochs.

3.2 Quantitative investigation

We follow the experimental procedure of data filtering similar to that of the Apache study [10], which is a classic study on contributors' participation in OSS projects. We retrieved and processed historical data

4) http://en.wikipedia.org/wiki/Red_Hat.
5) http://www.redhat.com/jboss.
6) http://www-01.ibm.com/software/webservers/appserv/geronimo/.
7) http://www-01.ibm.com/software/webservers/appserv/geronimo/features/?S_CMP=wspace.
8) http://geronimo.apache.org/committers.html.
9) http://en.wikipedia.org/wiki/Apache_Geronimo.
10) http://www.Bull.com/opensource/index.html.
11) http://www.Bull.com/news/021004jonas.html.
12) http://wiki.jonas.ow2.org/xwiki/bin/view/Main/Contributors.
13) http://wiki.jonas.ow2.org/xwiki/bin/view/Main/3rdPartyProducts.
14) http://www.osgi.org/Main/HomePage.
15) http://wiki.jonas.ow2.org/xwiki/bin/view/MoU/.
16) A type of issue tracking system. http://www.atlassian.com/software/jira.
17) http://community.jboss.org/wiki/HelpBugReport.

Table 2 Number of issue records for the three projects

Project	From	To	Number
JBossAS	2001.03.27	2010.09.14	24 855
Geronimo	2003.8.20	2010-09.14	6 929
JOnAS	2003.02.06	2010-09.14	1 581

from problem tracking repositories, and construct quantitative measures in Section 4. To investigate task distribution in issue reporting we identify internal and external reporters in Subsection 3.2.2. Table 2 gives a brief summary of the raw data we obtained.

3.2.1 *Issue tracking system*

JBossAS formerly used Sourceforge to track issues, but switched to JIRA in 2004. JOnAS started with GForge and switched to JIRA in October 2009, whereas Geronimo used JIRA from the very beginning.

We retrieved all web pages from the above mentioned systems. Irrelevant pages, such as index pages, were removed from further consideration. We also merged issues from multiple systems to remove duplicates, e.g., some issues of JBossAS were in both the JIRA and SourceForge trackers. Eventually, for each issue, we obtained the tuple of module, issue id, reporter, reported date, assignee, status, resolution, priority, type (*Bug, Feature Request, Patch or some others*), component, title, closer (if there was no closer, we set this attribute to "NULL"), and close date.

3.2.2 *Identifying external participants*

We used the email domain of the contact email addresses provided in JIRA to identify reporters of the corresponding commercial companies (internal developers, i.e., internal reporters). We considered the remaining reporters to be external. For example, reporters with domains "*@jboss.org*", "*@jboss.com*", or "*@redhat.com*" to be internal for JBossAS, while with email domains "*@Bull.net*", "*@Bull.com*", "*@ow2.org*" were considered to be internal for JOnAS, and "*@ibm.com*" were classed as internal for Geronimo.

There were some additional issues. For example, internal developers may not use the company's email domain when placing a report in JIRA. To validate the results, we verified our derived affiliation through the project mailing list, manually searched for each developer on the web, and checked whether the first change a developer made was to modify the "assignee" field to a known internal developer, as this behavior may point to a team leader or colleague of the internal developer. For Geronimo, we also combined the committers list on the project website to identify reporters who work for IBM.

4 Results

In this section, we provide answers to the research questions proposed in Section 1. We first investigate the major sources of issue reporting in Subsection 4.1. We then quantify the resolution speed of user reports to investigate whether the user experience improved following commercial involvement.

4.1 Major sources of issue reporting

A popular view is that the quality of OSS is ensured by the many users who download, use the software, and then report issues or even submit patches. As Mockus et al. [2] found, a larger group, beyond the core team, will discover more defects. These are the extensive user contributions from which companies involved in hybrid projects wish to benefit. However, when engaging in the projects, companies will bring in their own arrangements, which could affect the original OSS development practice. This might hinder user contributions and pose a challenge to the cooperation between internal and external participants. In this section, we identify who found and report issues during the epochs representing pure-OSS and hybrid development. This will help us understand how to take advantage of, or learn lessons from, the practice.

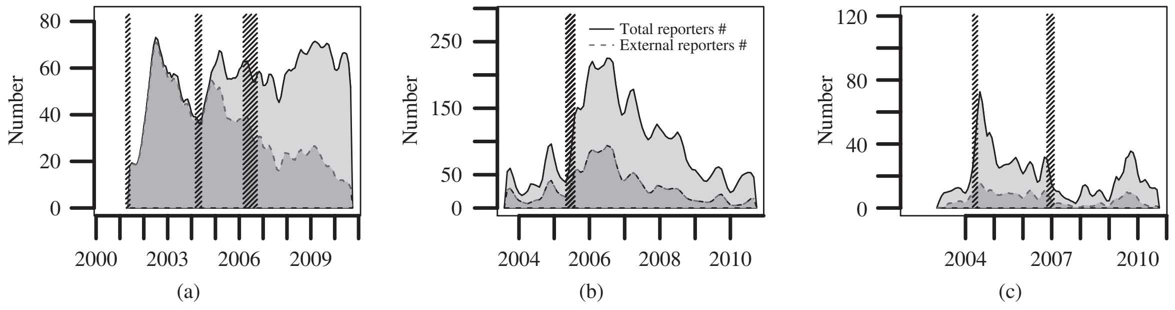

Figure 1 Number of external reporters in (a) JBossAS, (b) Geronimo, and (c) JOnAS.

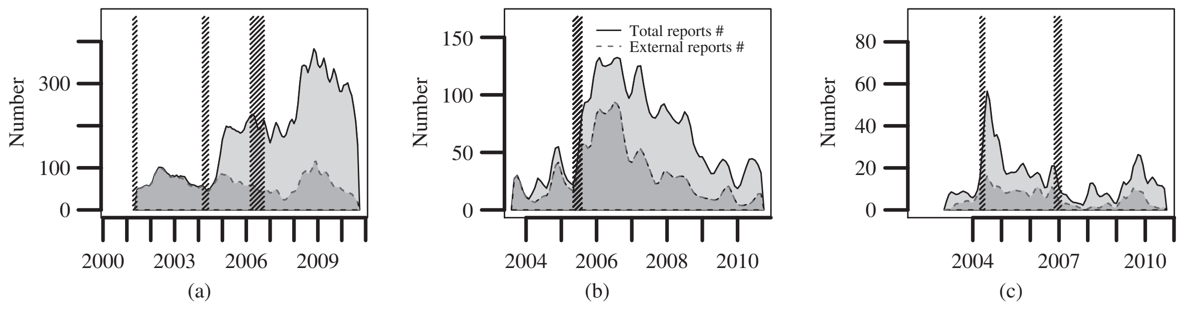

Figure 2 Number of external reports in (a) JBossAS, (b) Geronimo, and (c) JOnAS.

Figure 1 shows the number of reporters every month for the three projects. We distinguished the reporters as external and internal participants. Note that JBossAS did not start using issue tracking until the second epoch. The earliest reports we could retrieve were from March 27th, 2001, when the second epoch started. Therefore, we could only inspect the issue reporting behavior in the hybrid epochs of JBossAS. Although there was no first-epoch commercial involvement in Geronimo, we determined which issues had been reported by developers who were later hired by IBM, so as to distinguish the core team from peripheral participants.

We can see from Figure 1 that, after the onset of commercial involvement, most of the reporters seem to be internal rather than external reporters. For example, this is clearly the case in Epoch 4 of JBossAS, Epoch 2 of Geronimo, and all three epochs of JOnAS (note that there was some commercial involvement from the beginning of the JOnAS project). This reflects the fact that, with commercial involvement, internal developers will get to work full-time on the project, whereas external participants may only work part-time. Moreover, many companies channel participation through selected gatekeepers. This saves the internal developers the effort of responding to every request, and ensures consistency in feedback, so a single person may appear to be doing the work of many. This will also cause the number of external reporters to decrease.

Because external reporters will be more casual, and may have a different level of productivity than internal reporters, we determined the number of external and internal issues, as shown in Figure 2. We found a similar trend to that of the source of reported issues (Figure 1), except that the total number of issues increased more sharply than the number of reporters, a result of the higher productivity of internal reporters.

Simply looking at the absolute numbers of reporters/issues may reflect changes in the overall development activity more than in the external participation. We then considered the percentage of issues reported by external participants. As shown in Figure 3, in general, in the hybrid epochs of the three projects, the average proportion of issues reported by external reporters per month is never larger than 43%, with the lowest fraction of 10% in the fourth epoch of JBossAS. Notice that the second epoch of JBossAS (JBoss Group LLC) is an exception, where the external participants reported almost 70% of issues. At that time, Marc Fleury (creator of JBossAS) was still leading the community of volunteers, and was only in the process of starting to provide expert technical support services. In other words, this

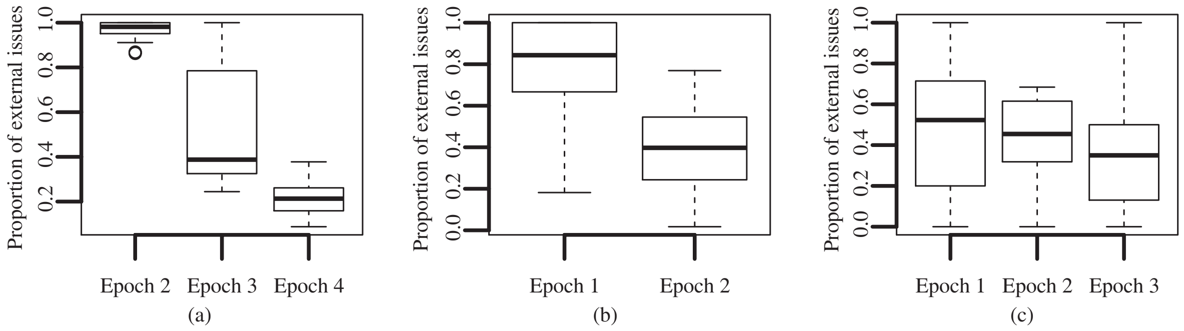

Figure 3 Proportion of external reports in (a) JBossAS, (b) Geronimo, and (c) JOnAS for each epoch.

epoch is likely to have more OSS than commercial features. Moreover, the percentage of issues reported by external reporters dropped in the last two epochs of JBossAS and in the second epoch of Geronimo. In contrast, in JOnAS, external reporters appear to be responsible for around 40% of issues, which suggests that the sharp drop in this percentage for JBossAS and Geronimo is probably due to changes in commercial involvement.

The proportion of external reports suggests that with commercial involvement, internal groups reported most of the issues, which implies that the original situation of OSS issue-reporting practice has changed. First, these hybrid projects do not seem to be so reliant on user-reports as the early OSS systems. The internal participants, mostly employees, will have more responsibility for arranging the development tasks to ensure the quality of software. One of the core developers from JBossAS also confirmed that they often record issues found in development. Second, because of the features of AS software and the business services provided by the companies, a number of issue reports will come from consultations with clients who are using the software. Although these issues are not found by developers themselves, the internal developers will still have the obligation to record and track them. Finally, because most issues are reported by internal developers, the issue-tracking process will become more formal and professional. This might have an impact on user-participation in issue reporting, e.g., deterring some casual and non-professional external reports. To determine the relationship between the increased internal participation and external participation in issue reporting, based on the results in Figures 1 and 2, we simply fit a Generalized Linear Model (GLM) for the over-dispersed Poisson distribution (which is suitable for the relatively low monthly counts of reporters).

$$\mathrm{Num}_{\mathrm{reporter}_{\mathrm{EX}}} \sim \sum_i \mathrm{epoch}_i, \tag{1}$$

where $\mathrm{Num}_{\mathrm{reporter}_{\mathrm{EX}}}$ is the number of external reporters, and epoch_i is an indicator function for each epoch. We also fit a similar model to the number of external issues $\mathrm{Num}_{\mathrm{issues}_{\mathrm{EX}}}$. From the estimated coefficients and p-values, we found that the trend of external issue reporting behaved differently in JBossAS than in the other two projects. In JBossAS, as the number of internal reporters increased, the number of external reporters decreased. However, rather than dropping as a consequence, the number of external reports remained relatively stable. This implies that the external reporters each reported more issues than before. This reflects the reality: in Epoch 2 and Epoch 3 of JBossAS, the companies tried to resolve important user issues, and paid a lot of attention to maintaining both an active forum and an issue tracking system to differentiate external issues. Note that the other two projects had no such mechanism to shunt external issues, and had fewer issues than JBossAS, so it seems that the numbers of internal and external reporters/issues changed with a similar trend.

In summary:

Observation 1. With commercial involvement, the major source of reported issues has shifted from users to developers. This suggests that hybrid development does not rely as extensively on user reports as the early OSS systems did.

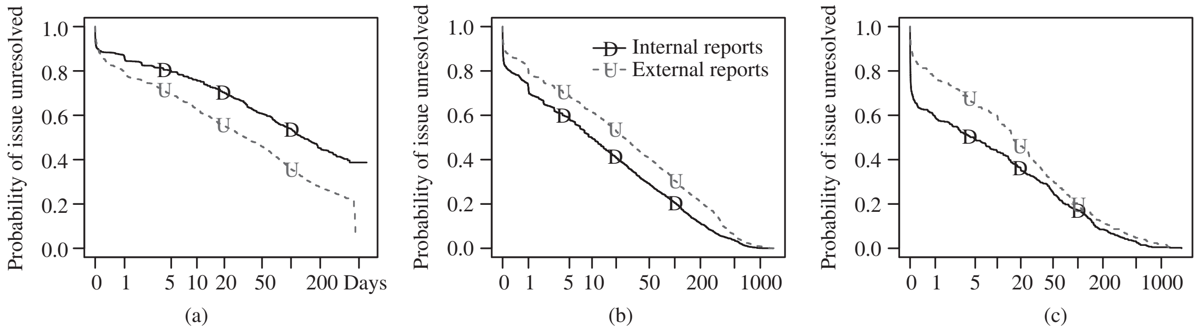

Figure 4 Issue resolution speed for (a) JBossAS (Epoch 3), (b) Geronimo (Epoch 2), and (c) JOnAS (Epoch 2).

4.2 Resolution speed of issue reports

Without the strict process of issue resolution and patch publishing in commercial projects, the response pace to issues in open source projects is much faster, as investigated in [2,25]. Quick issue resolution often implies a high degree of active project development, and also provides a good user-experience, which allows for an active and prosperous OSS community. In our previous study [9], the statistical results showed that all three projects responded to issues slower than Apache and quicker than commercial projects. In this section we further investigate the resolution speed of user issues and developer issues to identify the influence of commercial involvement. Note that, because of the openness of the software projects and lack of obvious identification of the reporters, it is hard to distinguish real software users from the records in the issue tracking system. To reflect the cooperation between internal and external participants, and to investigate the commercial impact, we assumed external reporters/reports to be the same as user reporters/reports, thus distinguishing their participation from that of internal developers.

We first investigated whether more intensified commercial involvement among epochs improved the speed of issue resolution, but found no obvious trend. Next, we compared the resolution speed of user issues with that of developer issues in each epoch, and obtained some interesting results.

We have drawn the survival curves of external and internal issues in Figure 4. These curves show the proportion of issues (y-axis) that were still open over a certain interval (x-axis). If a curve drops rapidly, it means that issues are being resolved quickly. For example, in Figure 4, more than 40% of external issues (i.e., user issues) for JBossAS were resolved inside 10 days, compared with nearly 30% for internal issues (i.e., developer issues). That is, the lower the curve is, the quicker the issues are resolved.

Because all the epochs in each project suggest the same trend for user and developer issue resolution speed, we illustrate only one epoch for each project in Figure 4. Note that the epoch selected for each project spanned a similar time period (e.g., 2004–2006 for JBossAS and JOnAS) or technique stage (i.e., "JavaEE 4 Specification") to make the projects as comparable as possible.

From Figure 4, we can see that user issues were resolved more quickly than developer issues in JBossAS. In the other two projects, however, developer issues were resolved quicker than user issues. In fact, to some extent, the difference in resolution speed between user issues and developer issues can reflect companies' user experience policies. Considering most of the issues were reported by developers, they were more likely to contain sufficient information to be resolved conveniently. This is the usual case in many projects, and is the case here for Geronimo and JOnAS. Otherwise, if the user issues are resolved quicker than developer issues, there should be some policy related to the business strategy to ensure a higher priority for user issues. As discussed in Subsection 3.1.3, the commercial strategy of JBossAS was different from that of the other two projects. The companies involved with JBossAS hosted the project and directly profited directly from it. According to the "JBoss Sales Machine" presented by the leaders of the JBoss project[18], the revenue stream of JBoss originated from exploring potential customer resources on the web, known as "raw leads". Naturally, the user experience on the websites like JIRA would be carefully

18) http://www.forentrepreneurs.com/lessons-from-leaders/jboss-example/.

nurtured. We interviewed one core developer on JBoss community day$^{19)}$, who confirmed that the usual way for JBoss developers to handle user issues in the first place. Moreover, shunting external issues to the forum and keeping the issue tracking system more professional in JBossAS helped with the rapid resolution of user issues. Note that, in turn, rapid resolutions also help make the project more popular.

Therefore, we have found that:

Observation 2. In hybrid projects, the resolution speed of user reports is influenced by commercial strategies. In some hybrid projects, e.g., projects from which companies make profits directly, user reports are resolved more quickly than developer reports.

5 Threats to validity

The primary limitations of our analysis concern the data, the measures, and the approaches used to interpret the evidence.

• **Raw data.** There are some limitations to our data collection due to non-artificial reasons, such as data lost when transferring the old issue system in JBossAS and JOnAS. We retrieved all records in both the old and new systems, and then removed duplicated records so as to capture as complete a collection of existing issue reports as possible.

Note that there are other places for users to report problems, such as mailing-lists and forums. For example, in Geronimo and JOnAS, there are some paying customers who have access to telephone and/or door-to-door support. These issues will not be recorded in public issue tracking systems such as JIRA. However, because our study focuses on user participation in the open Internetware environment, we conducted our investigation based solely on the records in the public issue tracking system.

• **Quantitative measures.** To quantify the participation of users and developers in issue reporting, we investigated the participation of external and internal reporters whose identities were derived from email domain names. To reduce the risk of misidentification, we manually checked the identities of reporters by searching for other email or related information online (e.g., using Google). Nevertheless, it should be noted that there were some cases in which it was difficult to determine whether an issue was from an external or internal source. For example, a user could report a problem over a mailing-list, and a developer may then enter this in the issue tracking system as necessary. To some extent, this kind of threat was due to the data, and our study could not avoid this problem.

When quantifying the resolution speed, we included all the issues opened in that epoch. This could cause errors to be introduced for two kinds of issues: those opened near the end of an epoch, when there was not enough time to find a resolution, and those issues opened in a former epoch but resolved at the beginning of the next epoch. Hence, we tried to include only the issues that were both reported and resolved in a single epoch, and obtained similar results. That is, whether the issues are resolved or not, the development principle of resolving user issues with high or low priority should be consistent. This consistency will be reflected in the data analysis. In fact, for both approaches to select issues, we found the same results: in JBossAS, user issues were resolved more quickly than developer issues. In addition, if we selected reported issues regardless of whether they had been resolved at the end of the epoch, there would be less of a limitation on the division of epochs, making our method more apportionable.

Comparing the money spent on OSS projects is a direct and effective way to measure the involvement of companies. However, there are some practical difficulties in this approach, e.g., obtaining such knowledge, confidentiality issues, and the duration of the projects. In fact, the money spent on each project is related to companies' business goals and strategies. Comparing the money spent would help us understand the development processes adopted in the projects. Although we were not able to clarify how much money was involved in the three projects, we can still investigate the business strategies and development processes, as in Subsection 3.1.3. This enables an understanding of the commercial involvement in the three projects.

• **Investigation method.** The particular approach of dividing the project time-line into epochs is debatable. For example, we could have divided the whole project lifespan into epochs of equal duration,

19) http://www.jboss.org/events/JUDCon/2012/china.

or separated all three projects into the same epochs. The approach we have presented is motivated by actual events within the companies or in the external environment. To exclude noise added by the transition between epochs, e.g., work handover, we excluded short periods around the epoch boundaries.

The three projects in our investigation are J2EE ASs. This may limit our findings to this domain. We made this choice to allow a comparison that would exclude the influence of the software domain and, thus, focus on our research questions.

6 Conclusion

In an earlier era, OSS and commercial software development implied two different philosophies, just like the bazaar and the cathedral illustrated in [3]. Some of the principles would be in conflict, such as non-profit vs. for-profit goals, loose vs. rigorous organization, non-paid volunteers vs. paid employees. Over the past decade, however, OSS projects have evolved: commercial-friendly OSS licenses have been invented (e.g., Apache license, MIT license), more companies have built business models to profit from OSS projects [1], many OSS projects are completely developed by companies [6], and OSS volunteers are being paid to develop OSS projects [13]. That is, current OSS projects have been deeply affected by commercial elements, and more are emerging. Our work provides elementary research that empirically shows how commercial software development and OSS development are combined.

Specifically, we investigated the impact of commercial involvement on OSS projects. Case studies were conducted for three OSS-commercial hybrid projects from the J2EE AS domain: JBossAS, Apache Geronimo, and JOnAS. We quantified the commercial influence through changes in contributors' participation in issue tracking practice, studying the background of the issue reporters and the resolution speed. The results demonstrate that a hybrid development approach will change the task distribution and improve the task performance. The main source of reported issues in an open issue tracking system has shifted from external to internal participants, with user reports likely to be resolved more quickly than developer reports.

Our findings provide practical insights for improving hybrid, OSS, and commercial software development. In particular, this study enables an understanding of development under the open, dynamic, and ever-changing Internet environment by utilizing various resources such as commercial involvement and user innovation. This provides a more in-depth understanding of what should be expected of a good paradigm for Internetware. However, the construction of more efficient collaboration platforms and tools (e.g., how to improve issue tracking practice), sufficient use of various Internet resources, and combination of OSS and commercial software development approaches still require further study.

Acknowledgements

This work was supported by National Basic Research Program of China (973 Program) (Grant No. 2009CB32-0703) and Nature Science Foundation of China (Grant Nos. 61073016, 61121063).

References

1 Bonaccorsi A, Giannangeli S, Rossi C. Entry strategies under competing standards: hybrid business models in the open source software industry. Manag Sci, 2006, 52: 1085–1098

2 Mockus A, Fielding R T, Herbsleb J. Two case studies of open source software development: Apache and Mozilla. ACM Trans Softw Eng Method, 2002, 11: 1–38

3 Raymond E S. The Cathedral and the Bazaar. O'Reilly Media, 1999

4 Hippel E V, von Krogh G. Open source software and the 'private-collective' innovation model: issues for organization science. Organ Sci, 2003, 14: 209–223

5 Von Hippel E. Democratizing innovation: the evolving phenomenon of user innovation. Int J Innov Sci, 2009, 1: 29–40

6 Riehle D. The single-vendor commercial open course business model. Inf Syst e-Bus Manag, 2012, 10: 5–17

7 Yang F Q, Lu J, Mei H. Technical framework for internetware: an architecture centric approach. Sci China Ser F-Inf Sci, 2008, 51: 610–622

8 Mei H, Huang G, Xie T. Internetware: a software paradigm for Internet computing. IEEE Comput, 2012, 45: 26–31
9 Ma X J, Zhou M H, Mei H. A case study of Internetware development. In: Proceedings of the 2nd Asia-Pacific Symposium on Internetware. New York: ACM, 2010. 9:1–9:13
10 Mockus A, Fielding R T, Herbsleb J. A case study of open source development: the Apache server. In: 22nd International Conference on Software Engineering, Limerick, 2000. 263–272
11 Yamauchi Y, Yokozawa M, Shinohara T, et al. Collaboration with lean media: how open-source software succeeds. In: Proceedings of the 2000 ACM conference on Computer Supported Cooperative Work, Philadelphia, 2000. 329–338
12 Ye Y W, Kishida K. Toward an understanding of the motivation open source software developers. In: 25nd International Conference on Software Engineering, Portland, 2003. 419–429
13 Roberts J A, Hann I, Slaughter S A. Understanding the motivations, participation, and performance of open source software developers: a longitudinal study of the apache projects. Manag Sci, 2006, 52: 984–999
14 Von Krogh G, Spaeth S, Lakhani K R. Community, joining, and specialization in open source software innovation: a case study. Res Policy, 2003, 32: 1217–1241
15 Zhou M H, Mockus A. Does the initial environment impact the future of developers? In: ICSE 2011, Honolulu, 2011. 271–280
16 Mockus A, Herbsleb J. Why not improve coordination in distributed software development by stealing good ideas from open source. In: ICSE'02 Workshop on Open Source Software Engineering, Orlando, 2002. 35–37
17 Riehle D. The economic motivation of open source software: stakeholder perspectives. Computer, 2007, 40: 25–32
18 Dinkelacker J, Garg P K, Miller R, et al. Progressive open source. In: ICSE2002 Proceedings of the 24th International Conference on Software Engineering, Orlando, 2002. 177–184
19 Munga N, Fogwill T, Williams Q. The adoption of open source software in business models: a Red Hat and IBM case study. In: the 2009 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists, Vanderbijlpark, Emfuleni, 2009. 112–121
20 Gurbani V K, Garvert A, Herbsleb J D. A case study of a corporate open source development model. In: Proceedings of the 28th International Conference on Software Engineering, Shanghai, 2006. 472–481
21 Wagstrom P, Herbsleb J, Kraut R, et al. The impact of commercial organizations on volunteer participation in an online community. Presentation at the OCIS Division, Academy of Management Conference, 2010. http://program.aomonline.org/2010/submission.asp?mode=ShowSession&SessionID=2095
22 Hendry D G. Public participation in proprietary software development through user roles and discourse. Int J Hum-Comput Studies, 2008, 66: 545–557
23 Ko A J, Chilana P K. How power users help and hinder open bug reporting. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York: ACM, 2010. 1665–1674
24 You C, Zhou M H, Xiao Z, et al. Towards a well structured and dynamic application server. In: 33rd Annual IEEE International Computer Software and Applications Conference, Seattle, 2009. 427–434
25 Bonaccorsi A, Rossi C. Why open source software can succeed. Res Policy, 2003, 32: 1243–1258