Open Source and Agile Methods:
Two worlds closer than it seems

Abstract. Agile methods and open source software communities have different approaches to produce high quality and successful software. However each approach excels in different scenarios and is not easily ported to a different environment.
This work presents an analysis of the proximity of the agile community and the open source community as basis for the elaboration of two surveys. Each survey was directed to one of the communities and aimed to identify communication issues encountered in each environment. The results of those surveys provide an insight to the greatest problems faced in each environment and the approaches that could be used to solve them.

Key words: agile software development, open source software, distributed agile, free software, software libre

1 Introduction

Typical Open Source (OS) projects (the scope of OS project will be narrowed according to Section 2) usually receive the collaboration of many geographically distant people [1]. At first glance, this argument could indicate that such projects are not candidates for the use of agile methods since some basic values seem to be missing. In this case, the distance and diversity separating developers deteriorates communication, a very important value within agile methods. However, it is common to identify some principles presented by the agile manifesto [2] in many OS software projects. Parts of the manifest such as being ready for changes, working with continuous feedback, respecting collaborators and users, delivering working features and facing challenges can be naturally found in the Free, Libre, Open Source Software (FLOSS) communities [5].

During a workshop [3] held at OOPSLA 2007 celebrating 20 years of the publication of "No Silver Bullets"[4], agile methods and OS software development were mentioned as two failed silver bullets having both brought great benefits to the software community. During that workshop, someone asked if the use of several failed silver bullets simultaneously could not raise production levels by an order of magnitude. The latest work by Russo, Scotto, Sillitti and Succi [6] is the recent leader of this sort of work, to which this paper contributes. We provide an analysis of parts of the agile and the FLOSS environments to observe if a merging between both could have benefical effects to software development.

This work is structured as follows. Section 2 defines what is meant by FLOSS and agile methods in this paper. Section 3 shows where agile methods and FLOSS development are close to each other and where they diverge. Section 4 presents two surveys elaborated to try to identify communication issues and possible tools

to minimize the problems of each community. Section 5 presents the results of the survey while Section 6 concludes and provides suggestions for future works.

2 Scope

The FLOSS environment as well as the agile methodologies both comprehend such a wide variety of projects, people and contexts that it is very hard to cover all of them. Therefore it is necessary to first define which part of each community will be analysed in this work.

In this work, we will treat any software engineering method that follows the principles of the agile manifesto [2] as an agile method. However the text will be guided by the most known methods such as eXtreme Programming [7], Scrum [8] and the Crystal family [9]. Closely related ideas from the wider Lean philosophy [10] and its application to software development [11] will also be mentioned.

The terms "open source software" and "free software" will be considered the same in this work although there are some differences in their specific contexts [13]. Projects will be said to be open source (or free) if their source code is available and modifiable by anyone with the required technical knowledge, without prior consent from the original author and without any charge.

OS projects essentially controlled by a single company fall out of the scope of this work. The reason for such reduction of scope is that projects controlled by companies, whether they have a public source code and accept external collaboration or not, can be run with any software engineering method established in the company since it can be enforced on the employees of this company.

Considering this scope, it is important to characterize the people involved in such kind of projects. In 2002, the FLOSS Project [14] published a report about a survey they conducted regarding FLOSS contributors. Their collected data [15] shows that 78.77% of the contributors are employed or self-employed (question 42) and that only 50.82% of the OS community are software developers while 24.76% do not earn their main income with software development (question 10). In addition to those results, the survey presents the fact that 78.78% of the collaborators consider their OS tasks more joyful (question 22.2) than their regular activities and 42.3% also consider them better organized (question 22.4). As an outcome of those results, we could say that OS contributors perceive their activities both pleasurable and effective.

Another survey [16] points out that 74% of open source projects have teams with up to 5 people and 62% of the contributors work with each other over the Internet and have never met physically.

Considering such characterization of the FLOSS community, the next section presents the relation between such development environment with the ideas and guidelines of agile methodologies.

3 How closely related are Open source and Agile?

In Martin Fowler's first version of "The New Methodology" [12], OSS development as part of the new methodology of software development along with now well known agile methods. Fowler decided to remove it from the final article because the FLOSS environment is so big and spread that any attempt to characterize the development process would undoubtedly fail in a given environment.

Afterwards, in 2002, Warsta [20] published a review about agile software development methods including and discussing FLOSS as an agile method. However, he stresses that FLOSS development is not a precise method and evolves differently for each project. Indeed, Eric Raymond's description of the development process in "The Cathedral and the Bazaar" [17] is closer to an experience report than to the description of a process with guidelines and practices. Nevertheless, Raymond's text presents several actions that could be related to the agile manifesto [2] and are common in other FLOSS projects.

FLOSS communities are, by definition, a group of people gathered around a FLOSS project. A working and useful software project attracts individuals to collaborate and develop the software [18]. It is the people's interactions that will define the development process, tools and even the goals of the software. FLOSS projects that do not evolve to fit the needs of its community are abandoned in a highly competitive environment where the best (according to certain criteria) gains adopters in a thin line between success and oblivion.

From such aspects, FLOSS projects share a lot with the values presented by agile methods. The practices are also quite similar. Eric Raymond quotes Linus Torvalds about two specific development policies for the Linux Kernel [17].

7. Release early. Release often. And listen to your customers.
8. Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone.

The first policy hints at an iterative and short development process with frequent feedback input. The second one points to a large amount of test done frequently. Those ideas are the core principles for agile methods and applied to several successful FLOSS projects.

We can, therefore, state that the FLOSS community have a culture that is similar to the one of the agile community. Given that there is such proximity between agile methods and FLOSS development, it must not be forgotten, nevertheless, that the environments in which each solution out stands are quite different.

Open source software is stronger when it comes to products that "scratch a developer's itch" [19], i.e. a product in which the developer can be the user. Such product evolves across the Internet gathering volunteers interested in such development by releasing versions of the software frequently.

Agile methods come from industry consultants with a focus on business problems and customer satisfaction through frequent high quality releases. To achieve such goal, an ideal agile team needs motivated competent people closely gath-

ered with freedom to improve their environment and process to enable what Cockburn calls osmotic communication [21].

The most critical discrepancy is the one regarding communication. Agile methods stress out that many software development problems come from the lack of high quality communication and the best way to mitigate it is to keep people close and in constant contact. Open source software are inherently distributed and frequently run by people that only meet through the Internet. Could there be a way to bring together the strength of both communities and improve development in distributed environments with changing requirements?

In order to evaluate this possibility, we decided to elaborate two surveys that will be described in the next section.

4 Surveys

Since the motivation to build those surveys was to understand if the agile and the FLOSS community could share more principles and practices, one survey was directed at each community. Each survey intended to characterize the answering public, was available on the Internet and spread in common channels to the target communities. The following subsections present both surveys.

4.1 To the FLOSS community

For both surveys, part of the goal was to identify which part of the community answered the question and how representative they were for their community. Therefore, the first set of questions were pretty similar in both surveys and asking participants about their year of birth and country of residence. Both surveys also tried to evaluate the experience of the participants in their community. For the FLOSS community that translated into how many projects the participant had contributed with and when did the first contribution happened.

This last question and the following ones were only displayed if the participant had contributed to at least one FLOSS project since most questions would be meaningless otherwise. In order to narrow the environment that the participants would have to evaluate as well as their experience in such project, they were asked the name of the main project and their role in it.

To understand how the project ensures communication between its collaborators, the participants were asked how big the project team was and, in case there was a team (not a single person team) what the communication channel used to communicate among them was. The survey also asked the participant to evaluate the quality of communication through that channel and included a similar question regarding the communication channel with the users of the project and its evaluated quality.

At last, the survey inquired which of the eight tools the project had already used and how they would classify the usefulness of those tools to mitigate their problem with the project development.

A paper version of the survey can be found in Appendix A. [1]

The survey was announced via twitter[2] by the authors and received the support of GitHub[3] who asked their users to fill in the survey. The survey was also sent to other FLOSS project hosting systems such as SourceForge.net, LaunchPad.net, CodeHaus, Google Code but none answered the request or provided any sort of answers. It was also published in a few blogs related to the FLOSS community and some international mailing lists.

4.2 To the Agile community

The beginning of the survey directed at the agile community was very similar to the one sent to the FLOSS community since its goal was to characterize the participants. After the first questions regarding the country of residence and the year of birth, the agile survey inquired the participants about how many agile projects they had participated in and when was their first agile project was.

Similarly to the FLOSS survey, this last question as well as the following were only displayed to participants who had been in at least one agile project. The survey kept on asking about the participant's main role in the project and the size of the team involved. The participant was then asked to inform the main communication channel used with the client of that project as well as the quality of communication on that channel.

Unlike the FLOSS survey, the next question inquired whether the participant had had any previous experience with applying agile methods in a distributed environment. If so, the participant was asked to describe the communication channel used in this distributed project and its quality.

The participants were then asked to sort the three most critical problems encountered in the agile environment in which they had participated. Afterwards the participant had to sort the top three tools (out of eight) that would help in a distributed agile environment.

Finally, the participants were asked if they were FLOSS contributors and, if so, how agile they considered their FLOSS project and how they would sort the problems and tools in that FLOSS environment.

The paper version of this survey can be found in Appendix B. [1]

This survey was announced in mailing lists related to the subject as well as blogs and twitter by various supporters. The authors also sought the support from the Agile Alliance but no answer was provided.

5 Survey results

The surveys were elaborated to be answered online by the participants and used some dynamic contents to minimize the amount of answers each participants

---

[1] The online address of the survey was omitted according to the submission rules

[2] http://twitter.com

[3] http://github.com

should provide. Such work was performed through Javascript but was not validated against older browsers such as Internet Explorer 6 and 7. Trying to answer the surveys using those browsers resulted in an invalid answer. This unexpected error provided extra information regarding the browser used by each community. The following subsections analyse the data of each survey.

5.1 Individual results from the FLOSS community

The results for the FLOSS survey were collected between 2009/07/28 and 2009/11/01. The survey received 309 entries from which 3 were duplicated data (same IP address and time entry) while 4 others were invalid (caused by Javascript errors). This data shows that about 1% of people connected with FLOSS communities use browsers which are incompatible with the current standards.

Out of those 302 valid entries, 122 were answers in which the participant never contributed to a FLOSS project but felt as part of the FLOSS community. Such attitude shows that only about 60% of the FLOSS community actually contributes with projects.
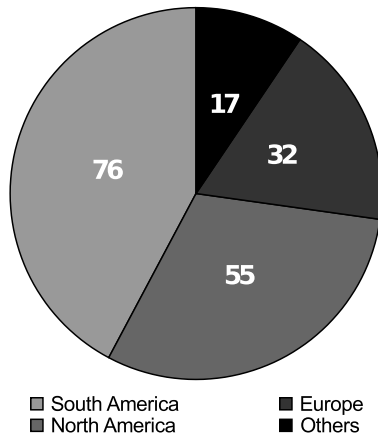

Fig. 1. FLOSS answers in the world
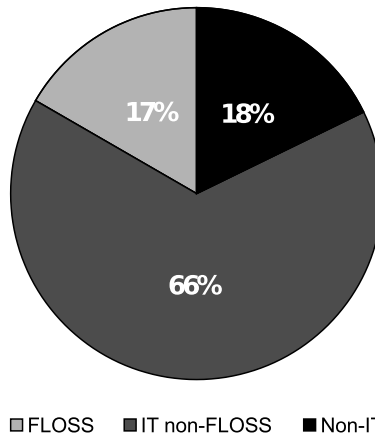

Fig. 2. Main income origin for FLOSS participants

The analysis was performed over the 180 answers left since they provided more interesting data. Figure 1 represents the distribution of the answers around the globe. Figure 2 shows the main income origin of the participants. It is interesting to notice that those data do not divert that much from the results collected from the FLOSS project mentioned previously in Section 2.

The average age of the participants was 28 years old and the average year of first FLOSS contribution was 2003. Figure 3 shows that younger participants started contributing earlier in their life than older ones which can be explained by the increasing ease to get in touch with a computer in the last years.
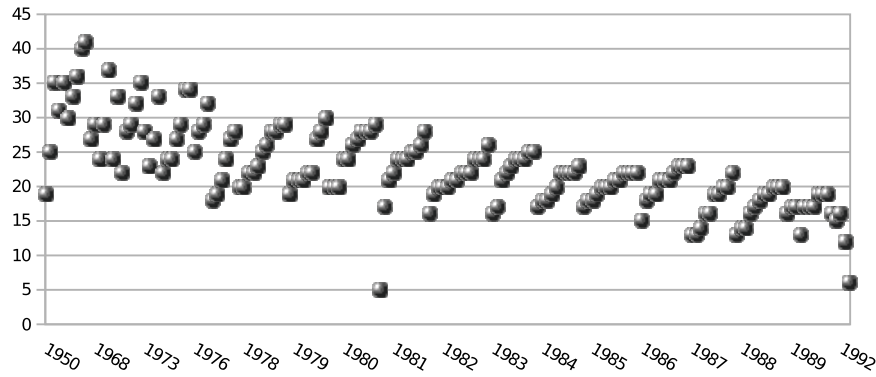
Fig. 3. Age of first FLOSS contribution by year of birth

About two thirds of the participants were project maintainers, commiters or programmers. The last third was partitioned between other roles as shows Figure 4. Team sizes were also fairly representative since only 6% of the project were single person team while 48% were up to 6 team members. Figure 5 shows those results and the reported team sizes. It is interesting to notice that such profile is similar to the one described by Reis [16] obtained in 2003.
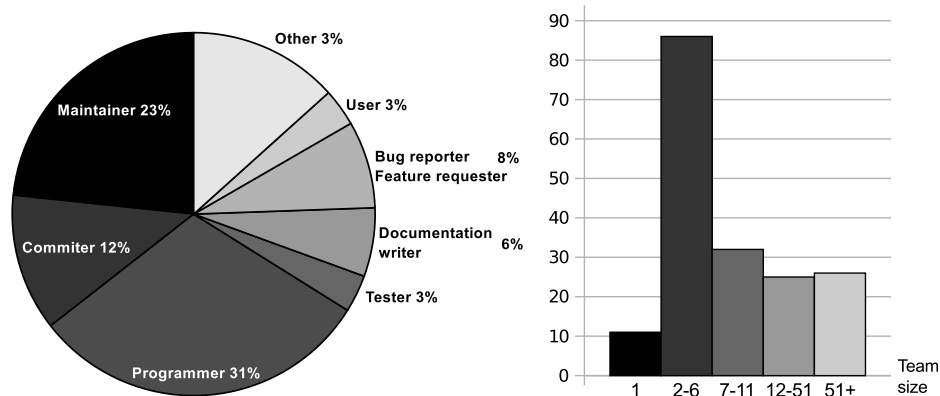
**Fig. 4.** Distribution of participant's roles **Fig. 5.** FLOSS projects reported team sizes

Regarding the main communication channels, it seems they have changed little since the FLOSS world or Reis' research. The main communication channels with the rest of the team are still mailing lists (27%) and Internet Relay Chat (IRC - 23%). However the amount of people using face to face communication within the team shows some increase (now at 15%).

The evaluated quality of communication in those channels was fairly similar. Mailing lists were evaluated to be 44% effective against 52% for IRC channels and 49% for face to face. It seems that with the growing adoption of fast feedback

channels over the Internet, mailing lists are showing signs of weakness comparing to higher bandwidth channels.

When it comes to communication with the users, mailing lists were the most used (32%) followed by websites (18%) and IRC channels, e-mails and issue trackers (11% each). When it comes to quality of communication in such channels, IRC channels scores once again with 49% effectiveness against 44% for mailing lists, 37% for websites, 33% for issue tracker and only 23% for e-mails.

For both environments, other communication channels were omitted since there were too few answers to show any significant data.
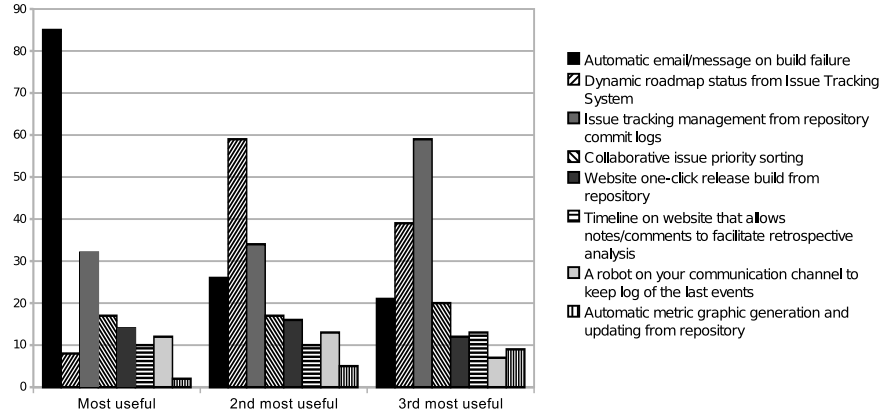

Fig. 6. FLOSS answers regarding tools usefulness

Figure 6 shows the top three choices of most useful tools within a FLOSS project. Automatic e-mail/message on build failure was considered the most useful tool by large followed by a dynamic roadmap status from the issue tracking system and issue tracking management from repository commit logs.

Figure 7 shows that a reasonable number of projects already have automatic e-mail/message on build failure and issue track management from repository commit logs. However, we should be aware that Github offers the latter feature while many other forges don't. And since Github officially announced the survey, it is probable that many of their users answered the survey. Therefore the sample could be biased in such sense.

## 5.2 Individual results from the Agile community

The results for the survey directed at the agile community were collected between 2009/10/01 and 2009/12/01. It received 204 answers from which 9 were duplicate entries and 34 were invalid due to the use of incompatible browsers. Such data shows us that about 18% of the agile community still use browsers incompatible with Javascript standards. Sensibly more people than in the FLOSS community.
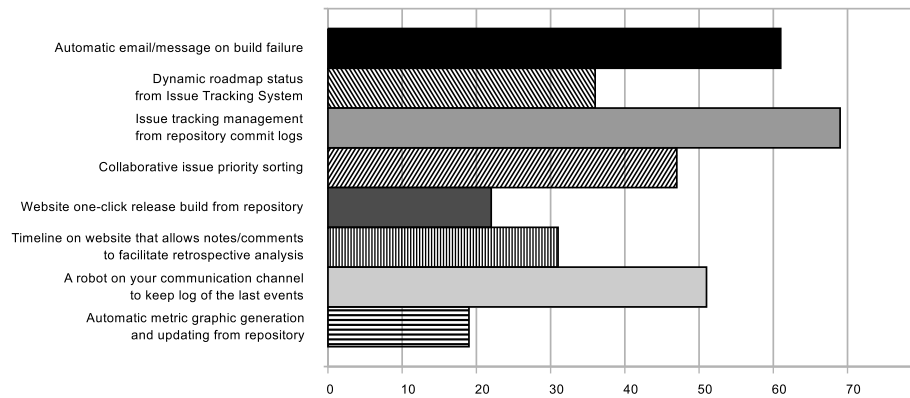
Fig. 7. Tool participants already use in their FLOSS project

Out of those 161 valid answers, only 28 were from people who had never actually participated in an agile project but considered themselves as agile practitioners. Another fairly different result from the FLOSS community. The agile community seems to value "hands on" experience much more than the FLOSS community.
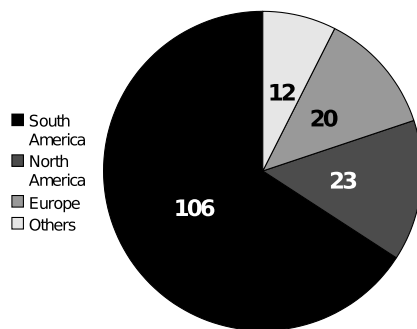

Fig. 8. Answers to the agile survey by region of the world
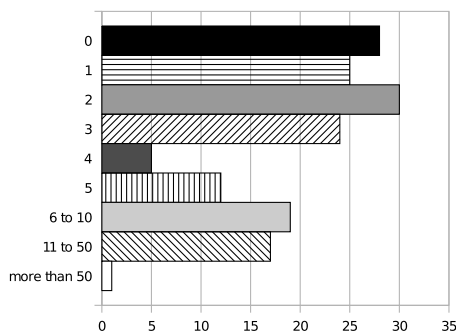
Fig. 9. Number of agile projects experience of the participants

However it is not a very deep experience since 51% of the participants were involved in at most 2 agile projects and only 23% had more than 5 agile projects experience. For the rest of the analysis, participants without any agile experience will not be counted since they provide little useful data.

Most participants with some experience only had a very recent contact with agile projects. Figure 10 shows that the first experience with agile for most participants only happened after 2006. We can also see that there is a fairly regular amount of people with distributed agile experience regardless of the year of first agile experience which suggests there has not been a major increase on distributed agile projects.
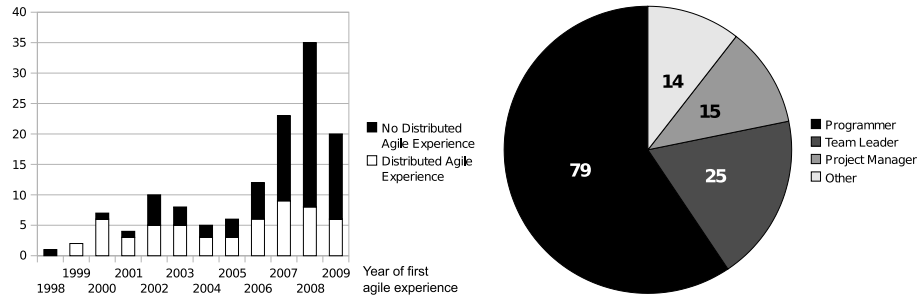
Fig. 10. Distributed agile experience according to the year of the first agile experience

Fig. 11. Roles description in the agile community

Figure 11 shows that most participants considered themselves programmers which contrasts with the variety of roles accumulated in the FLOSS survey. Such indication can be a consequence of the "Whole team" practice advocated by Kent Beck [7] in which he states that a good XP team does not have fixed roles but rather adapt to extract the best of each member so that anyone can perform any task if they are the best ones to do it. With such practice, members of an agile team are just developers that contribute to the project which can justify the smaller number of roles described.

When it comes to team sizes, smaller teams are obviously preferred. 37% of the participants reported to work with an average team size between 1 and 5 people. 46% reported to have teams from 6 to 10 people, 13% worked with 11 to 20 people and only 4% reported to work with more than 20 people. This shows that agile teams are still small teams following the original suggestions.

About 70% of those teams have face to face communication with their clients and evaluate the quality of such communication around 67%. E-mails, issue tracking systems and telephones accumulate another 19% of the teams' communication with their clients with only 54%, 50% and 35% effectiveness respectively. The rest of the channels are not used enough to provide trusted data.

In distributed environments, the results show that there is no clear consensus regarding the best communication channel within the team. There is no clearly most used communication channel nor a clearly most effective one. However, there is a clearly less effective one. E-mails share a reasonable part of the experiences but are rated around 31% effective to communicate between distributed teams which ranks way below most reported channels.

The ineffectiveness of this communication channel can explain why 56% of the participants stated that "discovering what the users/clients need/want" is the biggest problem they face in their agile projects. The second greatest problem is to "synchronize with other collaborators to achieve a common goal" and to "discover what is the next task to be done is".

Regarding the useful tools to help agile practitioners, the results are very similar to the ones listed by FLOSS contributors. The most useful tools for agile practitioners are exactly the same as for FLOSS contributors. Messages
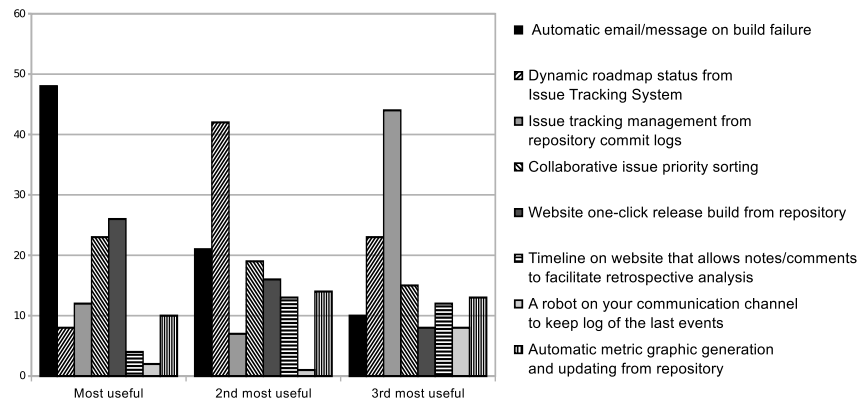
Fig. 12. Most useful tools to agile practitioners

warning of build failures lead the ranking followed by dynamic roadmaps and issue tracking management from commit logs.

It is no surprise that for the 35% of agilists who contribute to FLOSS projects, the problems encountered in their FLOSS environments are the same as in their agile environment. The tools to mitigate their FLOSS problems are also the exact same ones from the agile enviroment. However, such similarity does not come from the fact that the projects in which agilists contribute are agile. In average, the participants graded their FLOSS project to be only 56% agile. Such non agility with the similar results might indicate that agility and FLOSS development have the same unsolved problems.

6 Conclusion

Although a thoughtful analysis pictures agile software development closely related to open source software development, the results of the surveys indicate that the communities themselves are not that close to each other. However the mindset is quite similar since members of both communities identify the same issues and evaluate tools in a similar way. Such results indicate that there is a strong common root to both movements but there still is a considerable effort to be made to bring the movements closer.

When it comes to the tools still missing, both communities share the same issues and hope to facilitate integration between developers and increase the frequency of feedback input. Both FLOSS and agile projects should indicate better results by using fast feedback communication channels such as face to face encounters or IRC channels instead of slower channels such as simple e-mails to communicate within the development team.

Finally, this study showed that members of the FLOSS community as well as the agile community face the same issues and share a common view of possible solutions. This indicates that although bringing these two movements together is far from easy, there is no major cultural impact in this path.

Future work could study how to apply agile techniques, such as planning games and story writing, in FLOSS environments. Another idea could be to evaluate FLOSS projects according to the agility of the tools used. This evaluation could help discover the impact of tools that facilitate agile practices on the quality and success of the projects.

References

1. Dempsey, B. and Weiss, D. and Jones, P. and Greenberg, J.: A quantitative profile of a community of open source Linux developers (1999)
2. Beck, K. and Cockburn, A. and Cunningham, W. and Fowler, M. and Schwaber, K. and al.: Manifesto for Agile Software Development, http://agilemanifesto.org (2001)
3. Mancl, D. and Fraser, S. and Opdyke, W.: No silver bullet: a retrospective on the essence and accidents of software engineering (2007)
4. Brooks, F.: No Silver Bullet: Essence and Accidents of Software (1987)
5. Goldman, R. and Gabriel, R.: Innovation Happens Elsewhere: Open Source as Business Strategy (2005)
6. Russo, B. and Scotto, M. and Sillitti, A. and Succi, G.: Agile Technologies in Open Source Development (2009)
7. Beck, K. and Andres, C.: Extreme Programming Explained: Embrace Change (2004)
8. Schwaber, K.: Agile Project Management with Scrum (2004)
9. Cockburn, A.: Agile Software Development (2002)
10. Ohno, T.: Toyota Production System: Beyond Large-Scale Production (1998)
11. Poppendieck, T. and Poppendieck, M.: Introduction to Lean Software Development (2005)
12. Fowler, M.: The New Methodology, http://martinfowler.com/articles/newMethodologyOriginal.html
13. Fogel, K.: Producing Open Source Software (2005)
14. International Institute of Infonomics - University of Maastricht: Free/Libre/Open Source Software: Survey and Study - Report, http://www.flossproject.org/report/
15. International Institute of Infonomics - University of Maastricht: Free/Libre/Open Source Software: Survey and Study - Report, http://www.flossproject.org/floss1/stats.html
16. Reis, C.: Caracterização de um Processo de Software para Projetos de Software Livre (2003)
17. Raymond, E.: The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary (1999)
18. Crowston, K. and Scozzi, B.: Open source software projects as virtual organisations: competency rallying for software development (2002)
19. Feller, J. and Fitzgerald, B: A framework analysis of the open source software development paradigm (2000)
20. Abrahamsson, P. and Salo, O. and Ronkainen, J. and Warsta, J.: Agile software development methods (2002)
21. Cockburn, A.: Crystal Clear: A Human-Powered Methodology for Small Teams (2004)

A Paper version of the survey to the FLOSS community

1. What country do you live in? _____

2. What year where you born? _____

3. How many FLOSS projects have you already contributed with?
   ( ) 0 ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5-10 ( ) 11-50 ( ) 51+

4. What is the FLOSS project you mostly contribute/contributed with?   _____

5. In which year was your first contribution to that project? _____

6. What is/was your main role in that project?
   ( )Maintainer   ( )Documentation writer
   ( )Commiter   ( )Bug reporter/Feature requester
   ( )Programmer   ( )User
   ( )Tester   ( )Other: _____

7. Do/Did you receive any income from your FLOSS contributions? ( ) Yes ( ) No

8. If you do/did, is/was this your main income? ( ) Yes ( ) No

9. If you answered no to the previous question or the one before, is/was your main income related to IT? ( ) Yes ( ) No

10. How many people work (or worked) with you on your main FLOSS project?
    ( ) 0 ( ) 1-5 ( ) 6-10 ( ) 11-50 ( ) 51+

11. What is/was the main communication channel with the team of that project?
    ( )Face to face   ( )Instant Message (Jabber, ICQ, etc.)
    ( )Website   ( )E-mail
    ( )Mailing list   ( )VoIP (Skype, Ekiga, iChat, etc.)
    ( )Issue tracker (Trac, Bugzilla, etc.)   ( )None
    ( )Internet Relay Chat (IRC)   ( )Other: _____

12. How would you evaluate the quality of your communication with that team?
    Extremely poor ------------------------------------- Perfect

13. What is/was the main communication channel with the users of that project?
    ( )Face to face   ( )Instant Message (Jabber, ICQ, etc.)
    ( )Website   ( )E-mail
    ( )Mailing list   ( )VoIP (Skype, Ekiga, iChat, etc.)
    ( )Issue tracker (Trac, Bugzilla, etc.)   ( )None
    ( )Internet Relay Chat (IRC)   ( )Other: _____

14. How would you evaluate the quality of your communication with the users?
    Extremely poor ------------------------------------- Perfect

15. How much effort do/did you spend to keep project's information updated?
    Very little ------------------------------------- Huge

16. Which of the following tools does/did your project already use?
    [ ] Automatic e-mail/message on build failure
    [ ] Dynamic roadmap status from Issue Tracking System
    [ ] Issue tracking management from repository commit logs
    [ ] Website one-click release build from repository
    [ ] Automatic metric graphic generation and updating from repository

    [ ] Collaborative issue priority sorting
    [ ] Timeline on website that allows comments to facilitate retrospective analysis
    [ ] A robot on your communication channel to keep log of the last events

17. Sort the tools from the most useful (first) to the less useful (last).
    ( )Automatic e-mail/message on build failure
    ( )Dynamic roadmap status from Issue Tracking System
    ( )Issue tracking management from repository commit logs
    ( )Website one-click release build from repository
    ( )Automatic metric graphic generation and updating from repository
    ( )Collaborative issue priority sorting
    ( )Timeline on website that allows comments to facilitate retrospective analysis
    ( )A robot on your communication channel to keep log of the last events

B Paper version of the survey to the agile community

1. What country do you live in? _____

2. What year where you born? _____

3. How many projects you would consider agile have you been on?
   ( ) 0 ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5-10 ( ) 11-50 ( ) 51+

4. In which year was the first Agile project you participated? _____

5. What is/was your main role in that project?
   ( )Project manager   ( )Tester
   ( )Team leader   ( )Tracker
   ( )Programmer   ( )Documenter
   ( )Quality Analyst   ( )Other: _____

6. What is/was the average number of people in the agile projects you worked?
   ( ) 1-5 ( ) 6-10 ( ) 11-20 ( ) 21-50 ( ) 51-100 ( )100+

7. What is/was your main communication channel with the clients of that project?
   ( )Face to face   ( )Instant Message (Jabber, ICQ, etc.)
   ( )Website   ( )E-mail
   ( )Mailing list   ( )Telephone
   ( )Issue tracker (Trac, Bugzilla, etc.) ( )VoIP (Skype, Ekiga, iChat, etc.)
   ( )Internet Relay Chat (IRC)   ( )None
   ( )Other: _____

8. How would you evaluate the quality of your communication with the clients?
   Extremely poor ------------------------------------ Perfect

9. Have you ever been on a distributed agile project? ( ) Yes ( ) No

10. What is/was the main communication channel within that distributed team?
   ( )Face to face   ( )Instant Message (Jabber, ICQ, etc.)
   ( )Website   ( )E-mail
   ( )Mailing list   ( )Telephone
   ( )Issue tracker (Trac, Bugzilla, etc.) ( )VoIP (Skype, Ekiga, iChat, etc.)
   ( )Internet Relay Chat (IRC)   ( )None
   ( )Other: _____

11. If you had distributed agile experience, how would you evaluate the quality of your communication with that team?
    Extremely poor ------------------------------------ Perfect

12. For some of the common problems encountered by agile teams listed below, sort the 3 most critical problems in the agile environments you worked on.
    [ ] Discover what the users/clients need/want
    [ ] Discover what is the next task to be done
    [ ] Understand how the project works from a technical point of view
    [ ] Discover the current project status
    [ ] Integrate the source code to the main repository
    [ ] Keep the information about the project updated in its main communication channel
    [ ] Evaluate the work done to identify improvement points
    [ ] Synchronize with other collaborators to achieve a common goal

13. Sort the 3 tools that would most help you in a distributed agile environment.
    [ ] Automatic e-mail/message on build failure
    [ ] Dynamic roadmap status from Issue Tracking System
    [ ] Issue tracking management from repository commit logs
    [ ] Website one-click release build from repository
    [ ] Automatic metric graphic generation and updating from repository
    [ ] Collaborative issue priority sorting
    [ ] Timeline on website that allows comments to facilitate retrospective analysis
    [ ] A robot on your communication channel to keep log of the last events

14. Have you ever contributed to Free, Libre, Open Source Software (FLOSS)?
    ( ) Yes ( ) No

15. How would you evaluate the agile level of your FLOSS project?
    Anti agile ----------------------------------- Very agile

16. For some of the common problems encountered by agile teams listed below, sort the 3 most critical problems in the FLOSS projects you worked on.
    [ ] Discover what the users/clients need/want
    [ ] Discover what is the next task to be done
    [ ] Understand how the project works from a technical point of view
    [ ] Discover the current project status
    [ ] Integrate the source code to the main repository
    [ ] Keep the information about the project updated in its main communication channel
    [ ] Evaluate the work done to identify improvement points
    [ ] Synchronize with other collaborators to achieve a common goal

17. Sort the 3 tools that would most help you in that FLOSS environment.
    [ ] Automatic e-mail/message on build failure
    [ ] Dynamic roadmap status from Issue Tracking System
    [ ] Issue tracking management from repository commit logs
    [ ] Website one-click release build from repository
    [ ] Automatic metric graphic generation and updating from repository
    [ ] Collaborative issue priority sorting
    [ ] Timeline on website that allows comments to facilitate retrospective analysis
    [ ] A robot on your communication channel to keep log of the last events