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

Hugo Corbucci[1] and Alfredo Goldman[1]

Instituto de Matemática e Estatística (IME)
Universidade de São Paulo (USP) - Brazil
corbucci@ime.usp.br and gold@ime.usp.br

Abstract. Agile methods and open source software communities have different approaches to produce high quality and successful software. However agile methodologies are not very diffused in open source communities nor the members of those communities follow many agile practices.

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

1 Introduction

Typical Open Source (OS) projects (the scope of 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] on many open source software projects. Being ready for changes, working with continuous feedback, delivering real features, respecting collaborators and users and facing challenges are expected attitudes from agile developers naturally found in the Free and Open Source Software (FOSS) communities[5].

During a workshop [3] held at OOPSLA 2007 celebrating 20 years from the publication of "No Silver Bullets" [4], agile methods and OS software development were mentioned as two failed silver bullets having both brought great benefit to the software community. During the same workshop the question was raised whether the use of several failed silver bullets simultaneously could not raise production levels by an order of magnitude. This work is an attempt to suggest one of those merges to partially tackle software development problems.

2 Scopes

The FOSS environment as well as the agile 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.

Throughout this work, any software engineering method that follows the principles of the agile manifesto [2] will be considered and treated as an agile method. Focus will be given on the most known methods, such as eXtreme Programming (XP) [6], Scrum [7] and the Crystal family [8]. Closely related ideas will also be mentioned from the wider Lean philosophy [9] and its application to software development [10].

The terms "open source software" and "free software" will be considered the same in this work although they have some differences in their specific contexts [12, Ch. 1, Free Versus Open source]. Projects will be called 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 to 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 [13] published a report about a survey they conducted regarding FOSS contributors. Their collected data [14] 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 [15] 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 FOSS community, the next section presents what is 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" [11], he included open source software development as part of the new methodology of software development along with now well known Agile methods. He decided to remove it from the final article because the FOSS environment is so big and spread that

any attempt to characterize the development process would undoubtedly fail on a given environment.

Afterwards, in 2002, Warsta [19] published a review about agile software development methods including and discussing OSS as an agile method. However, he stresses that FOSS development is not a precise method and evolves differently for each project. Indeed, Eric Raymond's description of the development process from "The Cathedral and the Bazaar" [16] 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 FOSS projects.

FOSS communities are, by definition, a group of people gathered around a FOSS project. A working and useful software project attracts individuals to collaborate to evolve the software[17]. It is the people's interactions that will define the development process, tools and even the goals of the software. FOSS projects that do not evolve to fit the needs of its community are abandoned in a highly competitive environment where the best (in certain criteria) gains adopters in a thin line between success and oblivion.

From such aspects, FOSS project share a lot in common with the values presented by agile methods. The practices are also quite similar. Eric Raymond quotes Linus Torvalds about two specific development policy for the Linux Kernel.

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 will be obvious to someone.

The first policy hints for 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 core principles for agile methods and largely applied to several successful FOSS projects.

We can, therefore, state that the FOSS community have a culture that is similar to the one of the agile community. Given that there is such proximity between agile methods and FOSS development, it must not be forgotten 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" [18], i.e. a product in which the own developer can be the user. Such product evolve across the Internet gathering volunteers interested in such development by releasing versions of the software frequently.

Agile methods come from industry consultants with 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 gathered with freedom to improve their environment and process to enable what Cockburn calls osmotic communication [20].

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 unite 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. The next section describes how the surveys were elaborated and what information were expected to be collected.

4 Surveys

Since the motivation to write was to understand if the agile community and the FOSS community could share more principles and practices, one survey was directed to agile practitioners while the other was aimed to FOSS contributors. Each survey intended to characterize the answering public, was available on the Internet and was spread in channels common to the target communities.

The next subsection presents the survey created for the FLOSS community while the following one shows the one aimed to the agile community.

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 representatives they were for their community. Therefore, the first set of questions were pretty similar between both surveys and inquired the 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 to the amount of project with which the participant had contributed with as well the first year of contribution.

This last question as well as 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 as well as their role in the project.

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

At last, the survey inquired the participants about which of 8 tools did the project already used and how they would classify the usefulness of those tools to mitigate their problem with the project's 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 mailing lists for the Python community as well other international mailing lists.

4.2 To the Agile community

The beginning of the survey directed to the Agile community was very similar to the one to the FLOSS community since its goal was to collect data to characterize the participants. After the first questions regarding the residence country and the year of bird, the agile survey inquired the participants about their agile experience asking on how many agile projects they participated in and when was their first agile project.

Similarly to the FLOSS survey, this last question as well as the following were only displayed to participants that had been in at least one of 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 to communicate with the client of that project as well as the quality of communication on that channel.

Divergence from the FLOSS survey followed with a question inquiring if the participant 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 3 most critical problems encountered in the agile environment in which they participated. Afterwards the participant should sort the top 3 tools (out of 8) 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 would they sort the problems and tools in that FLOSS environment.

The paper version of this version can be found in Appendix B[4] 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.

[1] The online address of the survey was omitted according to the submission rules
[2] http://twitter.com
[3] http://github.com
[4] The online address of the survey was omitted according to the submission rules

5 Survey results

The surveys were ellaborated to be answered by the participants through the Internet and used some dynamic contents to minimize the amount of answers each participants 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 an extra information regarding the browser used by each community.

The individual results of each survey provide interesting data by themselves and are described in subsections 5.1 and 5.2 for the FLOSS survey and the agile one respectively.

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 answers (caused by Javascript errors). This unexpected data shows that approximately 1% of people who are connected with FLOSS communities use a browser not compatible with the standards.

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

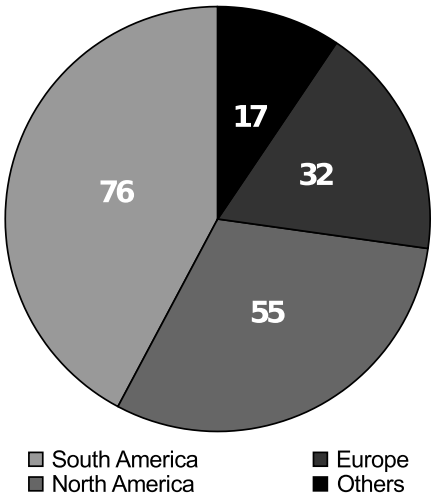

Fig. 1. FLOSS answers in the world

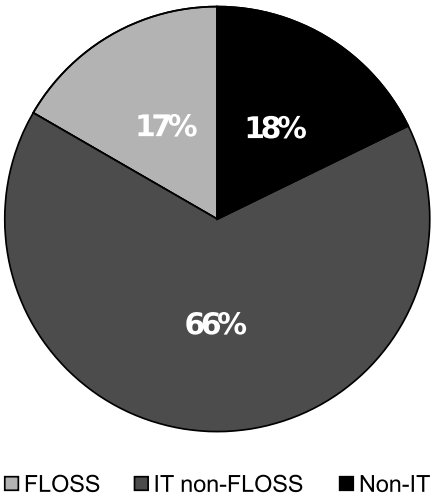

Fig. 2. Main income origin for FLOSS participants

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

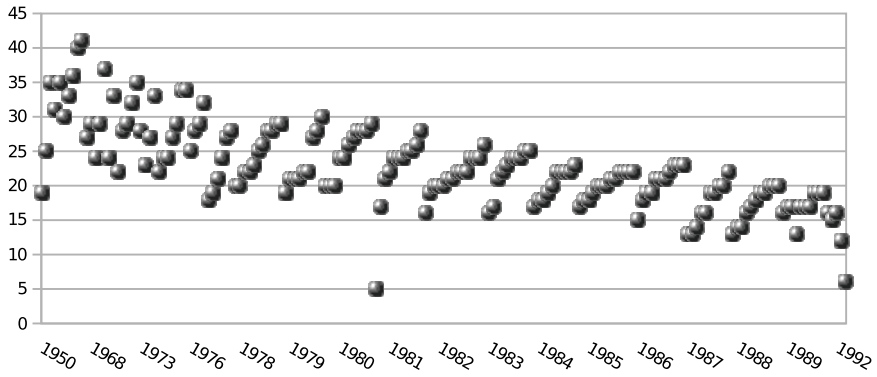

Fig. 3. Age of first FLOSS contribution by year of birth

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 5 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 [15] obtained in 2003.

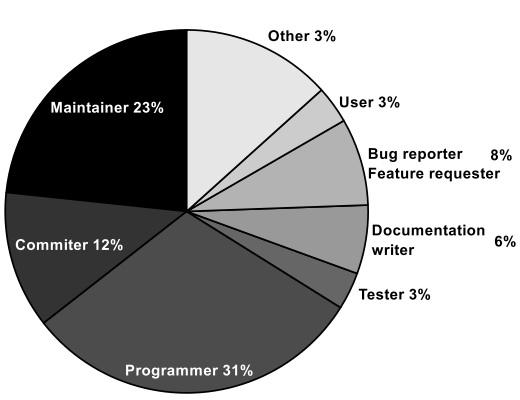

Fig. 4. Distribution of participant's roles

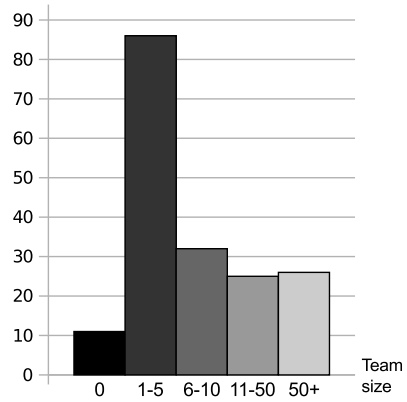

Fig. 5. FLOSS projects reported team sizes

Regarding the main communication channels, it seems to have changed a little since the FLOSS world research or Reis' one. 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 quality of the 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.

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

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

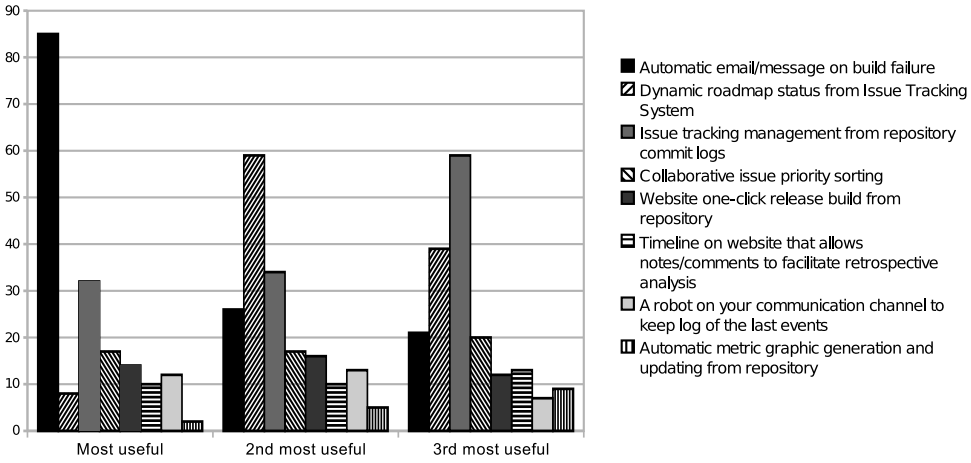

Fig. 6. FLOSS answers regarding tools usefulness

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

Figure 7 shows that a reasonable amount of project already have automatic email/message on build failure and issue tracking management from repository commit logs.

However, we should be aware that Github features an issue tracking management from repository commits while many other forges don't. And since Github officially announced the survey, it is probable that many of their users answered the survey.

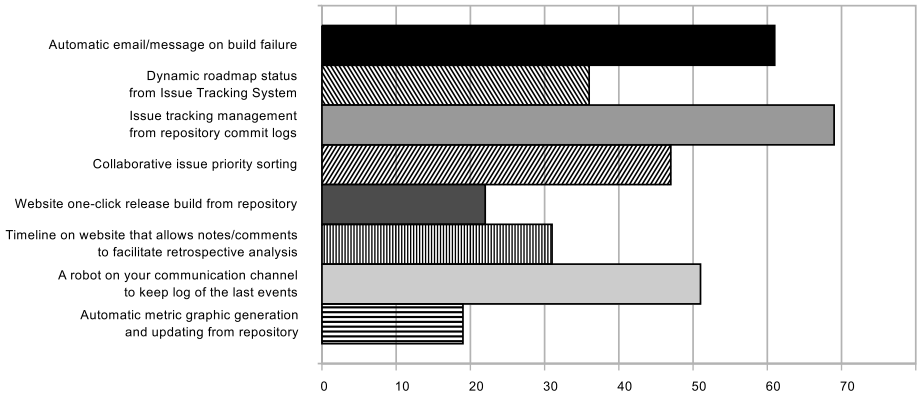

Fig. 7. Tool participants already use in their FLOSS project

5.2 Individual results from the Agile community

The results for the survey directed to 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 participants still use browsers incompatible with Javascript standards. Sensibly more people than in the FLOSS community.

Out of those 161 valid answers, only 28 were from people that 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.

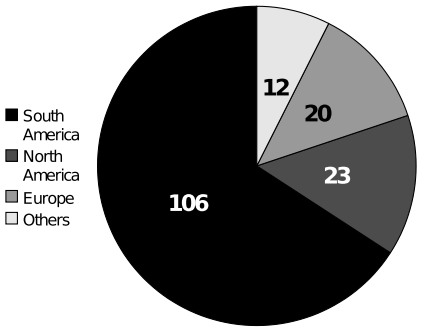

Fig. 8. Answers by region of the world

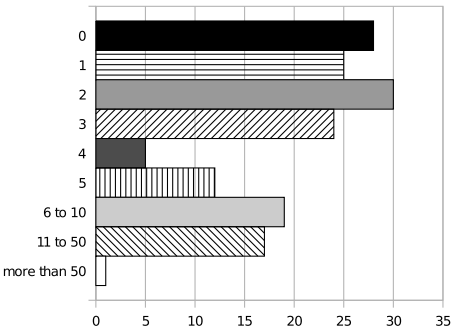

Fig. 9. Number of agile projects experience of the participants

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.

For those who had some experience, most still 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.

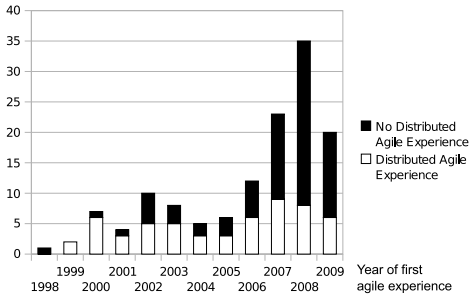

Fig. 10. Distributed agile experience according to the year of the first agile experience

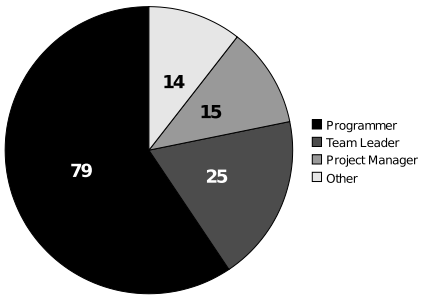

Fig. 11. Roles description in the agile community

Figure 11 shows that most participants considered themselves programmers which contrasts with the variety of roles accumulated in the FLOSS community. Such indication can be justified by the low hierarchy suggested by some agile books.

When it comes to team sizes, smaller teams are obviously preferred. Figure 12 shows the distribution of average team sizes reported by the participants. From such results, it is clear that a large majority of agile projects involve small teams.

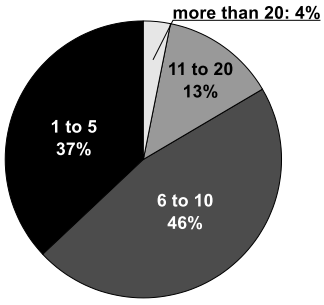

Fig. 12. Average agile team sizes

About 70% of those teams have face to face communication with their clients and evaluate the quality of such communication around 67%. Emails, issue tracking systems and telephones accumulate another 19% of the teams' communica-

tion with their clients with only 54%, 50% and 35% effectiveness respectively. The rest of the channels are not used enough to provide trustful 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. Emails share a reasonable part of the experiences but are rated around 31% effective to communicate between distributed teams.

The ineffectiveness of those communication channel can explain why 56% of the participants stated that "discovering what the users/clients need/want" is the biggest problem they face on 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".

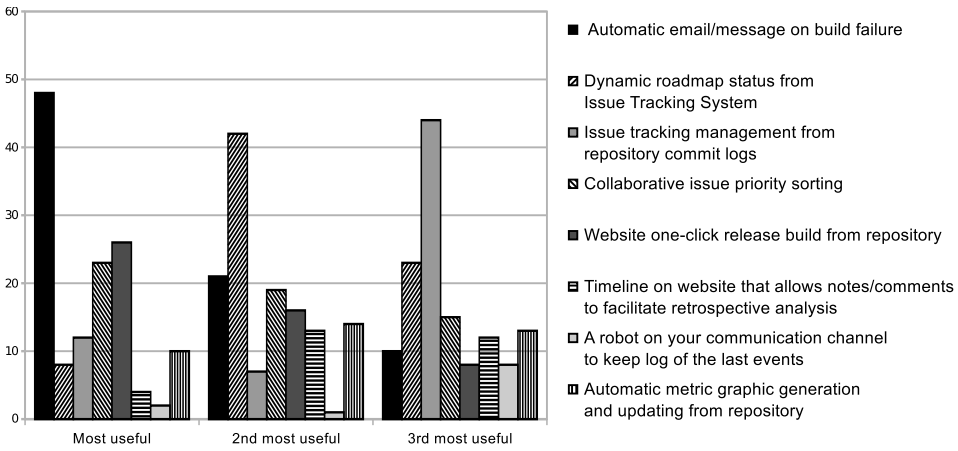

Fig. 13. Most useful tools to agile practitioners

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

Not surprisingly, for the 35% of the participants that contribute with FLOSS projects, the problems and tools are the same in that FLOSS environment. However, they considered their FLOSS projects to be only "half" agile. This might indicate that agility and FLOSS development have the same problems that still need to be solved.

6 Conclusion

The results of the survey indicate that the communities are not very closely related but share common problems and views to solve those problems. It seems, however, that the most severe issues are not directly addressed with the suggested tools.

Acknowledgments. This work was supported by the QualiPSo project [21].

References

1. Bert J Dempsey and Debra Weiss and Paul Jones and Jane Greenberg: A quantitative profile of a community of open source Linux developers (1999)
2. Kent Beck and Alistair Cockburn and Ward Cunningham and Martin Fowler and Ken Schwaber and al.: Manifesto for Agile Software Development, http://agilemanifesto.org (2001)
3. Dennis Mancl and Steven Fraser and William Opdyke: No silver bullet: a retrospective on the essence and accidents of software engineering (2007)
4. Frederick P. Brooks, Jr.: No Silver Bullet: Essence and Accidents of Software (1987)
5. Ron Goldman and Richard P. Gabriel: Innovation Happens Elsewhere: Open Source as Business Strategy (2005)
6. Kent Beck and Cynthia Andres: Extreme Programming Explained: Embrace Change, 2nd Edition (2004)
7. Ken Schwaber: Agile Project Management with Scrum (2004)
8. Alistair Cockburn: Agile Software Development (2002)
9. Taiichi Ohno: Toyota Production System: Beyond Large-Scale Production (1998)
10. Mary Poppendieck and Tom Poppendieck: Introduction to Lean Software Development (2005)
11. Martin Fowler: The New Methodology, http://martinfowler.com/articles/newMethodologyOriginal.html
12. Karl Fogel: Producing Open Source Software (2005)
13. International Institute of Infonomics - University of Maastricht: Free/Libre/Open Source Software: Survey and Study - Report, http://www.flossproject.org/report/
14. International Institute of Infonomics - University of Maastricht: Free/Libre/Open Source Software: Survey and Study - Report, http://www.flossproject.org/floss1/stats.html
15. Christian Robottom Reis: Caracterização de um Processo de Software para Projetos de Software Livre (2003)
16. Eric S. Raymond: The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary (1999)
17. Kevin Crowston and Barbara Scozzi: Open source software projects as virtual organisations: competency rallying for software development (2002)
18. Joseph Feller and Brian Fitzgerald: A framework analysis of the open source software development paradigm (2000)
19. Pekka Abrahamsson and Outi Salo and Jussi Ronkainen and Juhani Warsta: Agile software development methods (2002)
20. Alistair Cockburn: Crystal Clear: A Human-Powered Methodology for Small Teams (2004)
21. Qualipso — Trust and Quality in Open Source systems, http://www.qualipso.org/

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 name of the FLOSS project you mostly contribute (or contributed) with? _____
5. In which year was your first contribution to that project? _____
6. What is (or 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 do (did) not receive any income from your contributions or if those contributions were not your main income), 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 (or was) your main communication channel with your team of that FLOSS project?
    ( )Face to face ( )Instant Message (Jabber, ICQ, etc.)
    ( )Website ( )Email
    ( )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 (or was) your main communication channel with the users of that FLOSS project?

( )Face to face
( )Website
( )Mailing list
( )Issue tracker (Trac, Bugzilla, etc.)
( )Internet Relay Chat (IRC)
( )Instant Message (Jabber, ICQ, etc.)
( )Email
( )VoIP (Skype, Ekiga, iChat, etc.)
( )None
( )Other: _____

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

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

16. Which of the following tools does (or did) your project already use?
[ ] Automatic email/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 notes/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 email/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 notes/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 (or was) your main role in that project?
   ( )Project manager ( )Tester
   ( )Team leader ( )Tracker
   ( )Programmer ( )Documenter
   ( )Quality Analyst ( )Other: _____

6. What is (or 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 (or was) your main communication channel with the clients of that project?
   ( )Face to face ( )Instant Message (Jabber, ICQ, etc.)
   ( )Website ( )Email
   ( )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. If yes, what is (or was) your main communication channel within that distributed team?
    ( )Face to face ( )Instant Message (Jabber, ICQ, etc.)
    ( )Website ( )Email
    ( )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 email/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 notes/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 email/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 notes/comments to facilitate retrospective analysis
    [ ] A robot on your communication channel to keep log of the last events