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 difused 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 2.2) 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 cellebrating 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. Agile methods in the scope of this work are described in Section 2.1 while the OS projects discussed are described in Section 2.2.

2.1 Agile methods scope

Thoughout 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].

2.2 Open source scope

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 thise 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. Some methods will work better to attract external contributions but the company still controls its own team and can maintain the software without external collaboration.

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 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 undoutebly fail on a given environment. Aftewards, in 2002, Warsta [**?**] 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 bse, almost every problem will be characterized quickly and the fix 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 most sucessful 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 environment in which each solution outstand are quite different.

Open source software is stronger when it comes to products that "scratch a developer's itch" [**?**], 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 [**?**].

The most critical discrepency is the one regarding communication. Agile methods stress out that many software development problems come from miss communication and the best way to mitigate it is to keep people close and in constant conversations. Open source software are inherently distributed and frequently run by people that only met through the Internet. Could there be a way to unite the strenght 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 be united, one survey was directed to agile practionners while the other was aimed to FOSS contributors. Each survey intended to characterize the answering public and was spread in channels common to the desired community.

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

The survey directed to FLOSS contributors was announced via twitter by the authors and received the support of GitHub[1] who asked their users to fill in the survey.

4.2 To the Agile community

5 Survey results

5.1 Individual results from the FLOSS community

5.2 Individual results from the Agile community

5.3 Crossed results

6 Conclusion

Acknowledgments. This work was supported by the QualiPSo project [31].

[1] http://github.com

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 developement (2002)
18. Joseph Feller and Brian Fitzgerald: A framework analysis of the open source software development paradigm (2000)
19. Alistair Cockburn: Crystal Clear: A Human-Powered Methodology for Small Teams (2004)
20. Andy Oram: Why Do People Write Free Documentation? Results of a Survey (2007)
21. Dirk Riehle: The Economic Motivation of Open Source Software: Stakeholder Perspectives (2007)
22. Jeff Sutherland and Anton Viktorov and Jack Blount and Nikolai Puntikov: Distributed Scrum: Agile Project Management with Outsourced Development Teams (2007)
23. Frank Maurer: Supporting Distributed Extreme Programming (2002)
24. Kent Beck: Tools for Agility, http://www.microsoft.com/downloads/details.aspx?FamilyID=ae7e07e8-0872-47c4-b1e7-2c1de7facf96 (2008)
25. Nachiappan Nagappan and Prashant Baheti and Laurie Williams and Edward Gehringer and David Stotts: Virtual Collaboration through Distributed Pair Programming (2003)
26. Dan North: Behaviour Driven Development, http://dannorth.net/introducing-bdd

27. Danilo Sato and Alfredo Goldman and Fabio Kon: Tracking the Evolution of Object-Oriented Quality Metrics on Agile Projects (2007)
28. J. Surowiecki: The Wisdom of Crowds: Why the many are smarter than the few and how collective wisdom shapes business, economies, societies, and nations (2004)
29. Don Tapscott and Anthony D. Williams: Wikinomics: How Mass Collaboration Changes Everything (2006)
30. Yochai Benkler: The Wealth of Networks: How Social Production Transforms Markets and Freedom (2006)
31. Qualipso — Trust and Quality in Open Source systems, http://www.qualipso.org/