

# Ego & eXtreme Programming in the Libre Software Context

Paul James Adams, University of Lincoln

padams@hemswell.lincoln.ac.uk

## Problem Background

In his essay of 1998, “Homesteading the Noosphere” [3], Eric Raymond shows how within the open source community there are various subcultures that are formed on the basis of zealotry and (anti)commercialism. Almost any culture imaginable is going to be divided into sub-cultures according to varying levels of belief in certain key principles. At the lowest levels, it is these beliefs that make us individuals. Although Eric’s observation, by itself, may appear to be trivial, in the context of XP[2] in libre<sup>1</sup> software it is vital. As he points out, “It is worthwhile to point out the distinctions because they imply different agendas, and different adaptive and cooperative behaviors.”

Ego plays an unusual role within the libre process. In the first place, gratuitous ego-gratification (as with most communities) is generally regarded as a bad thing. Secondly, many open source practitioners are taking part because of the recognition they hope to obtain through making their contribution. Because of these contradicting scenarios within the same community, ego-gratification can only successfully be obtained through continued contribution. Again, parallels can be drawn between libre software and almost all other known communities, eg. politics, military and education. However, it is not too often we hear conversations such as, “I fixed a bug in Linus Torvald’s code”. When a project gains notoriety, the project and the lead programmer’s contribution are often mentally inseparable. Is it possible to discuss Linux without mentioning Linus?

When a project becomes this influential on the libre community, major contributors and project leaders can obtain ego satisfaction through knowing that other people use their software, enjoy it and want to contribute to it. Lesser contributors to projects have to claim ownership rights over their

---

<sup>1</sup>The term “libre” is used to avoid confusion over which of the open source of free software communities I may be referring to. In saying “libre” I am referring to both of these communities.

contributions without going too far and upsetting the community; a fine line to tread.

Libre software is formed in a reasonably agile manner. Its high level process is generally based on short releases each producing useful code. The biggest strength of the libre software process is contained in its low-level activities, the work of the individuals. Contributors within a libre project do have to conform to certain processes within the project. For example the project leaders will determine dates for releases and make policies on which software tools to use etc. However, the libre process also affords the contributors certain freedoms: individual contributors can choose their work environment, they can (in most cases) use what software they wish, take breaks when they wish etc.

Given the role that the ego plays within libre software development and the freedoms given to the contributors, the main question this research proposes is “*Can libre software development succeed when the contributors’ low-level process is dictated?*”. In particular this project will look at applying eXtreme Programming in the libre software content. XP has been chosen as it is an agile method (and therefore more likely to be suited to libre development) and because of its unique activities which normally require collocation: pair programming, story card development and continuous integration. Pair programming is of particular interest as it forces two contributors to adhere to the same work practises as each other, where usually they would work in a manner that suited themselves.

## Research Hypothesis

The agile manifesto[1] provides a noble approach to software engineering. Unfortunately, there is an irony inherent in agile methods; the very fact that they are methods at all. Within the agile manifesto, practitioners of agile methods stated their core beliefs as software engineers. These included beliefs that producing software was more important than producing documentation to support the engineering process and that individuals involved in the project are more important than the tools they use. Unfortunately there are occasions when the importance of tool support for the process cannot be denied.

Libre software is produced as part of a mostly agile process. In some respects the libre method is the most agile of the recognised agile methods as it allows the individuals within the project, in most respects, to work under the conditions they choose, where and when they want and how they want. Plus, of course, libre software development follows a high-level process of short releases each producing useful software.

The main hypothesis of this research is that it is possible to develop an environment to support distributed eXtreme Programming in a manner

that allows a distributed team to to undertake collaborative activities without collocation or loss of productivity; using project velocity as a metric for productivity. This, in itself, provides the problem of having to find a method for measuring productivity with the Libre software context. However, this hypothesis also leads to a further question. Is it possible to get libre software developers to adopt the XP process without suffering loss of freedom and ownership. More interestingly it leads this research to a further hypothesis. In some cases, XP within the Libre context will result in a loss of productivity because of the role ego will play in pair programming.

It is obvious that XP within libre may fail, in some cases, because of geographical reasons. Asking a contributor in Spain to pair program with a collaborator in the USA is asking too much because of the time difference. This research is interested in the role of ego in libre pair programming. Libre software development is, to a reasonable extent, ego driven. Although XP practitioners would be making a contribution to a Libre project by reviewing someone's code while they type, it may not carry the same respect, in their mind, as the coding itself.

## Proposed Work

This research is in its initial stages and is currently concerned with developing a system to support the distribution of eXtreme Programming. It is planned that in the future this system will be used within two different contexts in order to evaluate its effectiveness; within industry and within the libre software community.

## References

- [1] The agile manifesto. <http://www.agilemanifesto.org>.
- [2] Kent Beck. *eXtreme Programming Explained: Embrace Change*. Addison-Wesley, 2000.
- [3] Eric Raymond. *The Cathedral and the Bazaar*, chapter Homesteading the Noosphere. O'Reilly, 2001.