Towards the Effective Distribution of Agile Practice

Paul Adams, Cornelia Boldyreff[⋆]

Department of Computing and Informatics
University of Lincoln

Abstract. The agile methods are quickly gaining notoriety amongst software engineers. Having been developed over the past decade, they now present a mature, lightweight alternative to the "classic" approaches to software engineering. Although agile methods have solved some of the problems of established software engineering practice, they have created some problems of their own. Most importantly, we can infer a, potentially problematic, requirement of collocation.
In this research I intend to develop a system that will allow the effective distribution of agile practice, with a particular focus on the eXtreme Programming method. This paper discusses the motivation for this research and outlines the proposed research method and evaluation.

1 Introduction

The usefulness of the agile methods is restricted by their requirements of collocation and small development teams. If these requirements can be loosened then it would be possible to apply agile methods to a larger arena of software development. This research intends to extend the usefulness of agile methods by defining a new paradigm for software engineering practice, the "Liberal" paradigm.

This paradigm shall be important as it will encompass all the important features of both agile and libre software engineering practice in order to facilitate the distribution of effective agile practice. Figure 1 shows how this "Liberal" paradigm may be generated.

This new, "Liberal", paradigm will be based on the principles of agility combined with the experience of libre software process distribution. It is intended that this paradigm will provide a new set of processes that allow the distribution of agile practice within the open source paradigm, but also potentially improve the process of collocated agile teams.

2 Research Method

My research is largely based on empirical process. After thoroughly researching the requirements of agile programmers and distributed software engineering I

[⋆] Project Supervisor

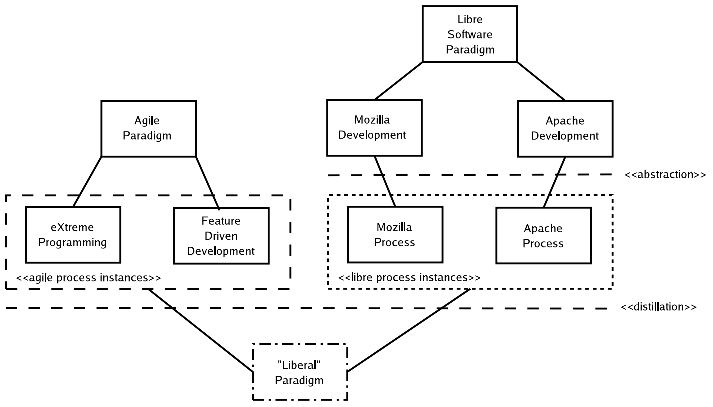

Fig. 1. The transition from agile and libre paradigms to the "Liberal" paradigm.

shall iteratively implement features for the distributed agile environment. Once the entire system is complete I shall evaluate the system through experimentation, improvement (where required) and then further experimentation etc.

This research is focused on a bottom-up approach, that is, the development of a tool for supporting distributed agile practice; the development of a process for this tool and ensuring this process fits within the "Liberal" paradigm are secondary. However, there is an element of top-down approach, in that it is possible to form some predictions on the "Liberal" paradigm and processes within it.

From the agile paradigm the "Liberal" paradigm will inherit some of the high-level principles of the Agile Manifesto[1]. However, these processes will have to be more adaptable than the current agile processes to ensure that the principles do not conflict. For example, it may not be desirable to allow customer collaboration to restrict code production. From the libre paradigm the "Liberal" paradigm will inherit the low-level freedoms that libre practitioners are afforded. It will also be important for the "Liberal" paradigm to inherit the ability to scale from the libre software paradigm; agile methods are designed with small or medium sized teams in mind. As figure 1 shows, the new "Liberal" paradigm will be formed as a distillation of the agile and libre features.

3 Tool Support for the "Liberal" Paradigm

Unlike some research where process has been the focus [JRS04], the principle contribution of this research shall be a tool set for supporting processes within the "Liberal" paradigm. This tool set must be rich enough to support distributed agile practice, but flexible enough to allow libre software practitioners the low-

[1] http://www.agilemanifesto.org

level freedoms they are used to. There are three types of tool that will be developed within this project: awareness tools, communication tools and task support tools. All of these shall be encapsulated within an Eclipse[2] environment. The tasks initially identified for support include distributed story cards, virtual meetings, component integration and most importantly, pair programming.

It is not necessary to develop an entirely new integrated development environment (IDE) to support the workflow of distributed agile practice. Instead, I propose to build support for distributed agile as a plug-in for the Eclipse IDE. My proposal is that a user of the Eclipse IDE can select to take part in a distributed agile project and that the interface shall be customised to suit. Eclipse is suitable for this purpose as it was designed to be easily extensible.

The first prototyped tool of this system offers support for pair programming. Unlike with previous work on supporting the distribution of XP [MDB+04,MM02], my work is focused on producing an environment built with pair programming in mind, as opposed to adapting the use of existing software to fit the needs of distributed agile practitioners.

This prototype tool allows for an agile programmer to code while another can view their work and comment on it in their own window. Feedback is marked on the code and the user can access it through the interface.

4 Proposed Case Studies

It is proposed that the tools developed as part of this research will be evaluated within two case studies: an study of the tool in industry and a study of the tool used within the libre software paradigm.

Other than to asses the effectiveness of the developed tool, the purpose of the industrial case study will be to assess the scalability of processes within the "Liberal" paradigm. The purpose of the libre software case study shall be to see how well the new process performs in different contexts.

4.1 Distributed Agile Practice in Industry

The main aim of this research is to improve on agile methods to enable them to be distributed and to improve upon their scalability. However, it may prove difficult to enlist the support of an industrial partner to assess this. Instead, it may be required that this case study is conducted using students.

This case study may be used to establish how easy it may be for industrial partners to contribute to a libre software project following the "Liberal" paradigm. More importantly, this case study will form the basis of the main body of evaluation for the developed system.

[2] http://www.eclipse.org

4.2 Distributed Agile Practice in the Libre Software Paradigm

As figure 1 shows, within the libre software paradigm a process of abstraction will have to take place in order to facilitate the generation of the "Liberal" paradigm. It has been shown that, as we might expect, there is a great difference between the libre-software processes [MFH02] and, therefore, some of these processes are more agile than others.

Within this particular case study, a set of libre software development features shall be abstracted from current projects. These libre practices shall form the second of the required inputs for the development of the "Liberal" paradigm. However, the scope of this part of the research will be limited to those libre processes that closely fir the agile paradigm. This should ensure that the "Liberal" paradigm has a greater opportunity for success.

5 Research Evaluation

The important aspect of evaluating this project shall be the development of a suitable set of metrics for assessing how effective the distribution of agile pracrtice has been. These metrics shall be applied within a goal-question-metric evaluation framework [vS99]. Suitable metrics may be project velocity, number of change requests or number of participants.

The obvious (and ideal) method for evaluating this project would be based on a groups of experienced agile practitioners using the final system. It would be possible to evaluate the performance of my system by comparing key metrics, most importantly project velocity, between collocated and distributed processes. The results of these evaluations could be used to refine and improve the system. It is hoped that much of the evaluation will take part within the Calibre[3] project, or similar.

An alternative approach to the evaluation shall be based on a iterative evaluation program. In the first iteration, as components of the system are developed (support for pair programming, distributed meetings etc.), they shall be evaluated individually. Again, this would be a metric-based analysis but performed by students working on group projects at the University of Lincoln. For the individual components I would be able to evaluate the performance of teams using the components against those not using the components. The evaluation of the system components, based on student activities, will allow for further development and refinement of the system.

In the second iteration, evaluation of the entire system will take place using experienced software engineers. However, the focus shall not be on improvement of the system but purely on assessing the performance with a view to establishing a basis for comparison between collocated and distributed performances. This analysis shall allow me to draw conclusions as to what a process within the "Liberal" paradigm may look like.

[3] http://www.calibre.ie

6 Conclusions

Communication is a key driver for successful collaboration and is therefore necessary within a software development process. We can infer from many agile methods that communication must take place in a collocated manner. This project aims to allow the distribution of effective agile practice by providing tool support within a hypothesised process and paradigm.

The “Liberal” paradigm, inheriting the important high-level features of agile practice and the low-level features of libre software practice, can be described in general terms before the development of tool support is complete. Processes within this paradigm can be hypothesised from the results of evaluating the tool in use within different contexts: industry and libre software development.

The main focus of this research is the creation of a tool set that aid the distribution of effective agile practice. This tool must not only support agile practice (pair programming, end-user collaboration etc.) but also offer support for the communication and awareness overheads that distribution cause. It is intended that the tool developed within this research, as part of the “Liberal” paradigm, will aid the distribution of effective agile practice in a manner that is appealing to both industrial practitioners and libre software practitioners and thus broaden the usefulness of the agile methods.

References

[JRS04] Ricardo Jota and António Rito-Silva. Supporting distributed extreme programming with adaptive workflow. In *Automated Software Engineering: Proceedings of the Workshop on Cooperative Support for Distributed Software Engineering Processes*, September 2004.

[MDB+04] F. Maurer, B. Dellen, F. Bendeck, S. Goldmann, H. Holz, B. Kötting, and M. Schaaf. Merging project planning and web-enabled dynamic workflow technologies. In *IEEE Internet Computing*, May 2004.

[MFH02] Audris Mockus, Roy T. Fielding, and James D. Herbsleb. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*, July 2002.

[MM02] Frank Maurer and Sebastien Martel. Process support for distributed extreme programming teams, 2002. Available at `http://sern.ucalgary.ca/ milos/papers/2002/MaurerMartel2002a.pdf`.

[vS99] Rini van Solingen. *The goal/question/metric method: a practical guide for quality improvement of software development.* McGraw-Hill, 1999.