Empirical experiments with XP.

Francisco Macias
University of Sheffield
Regent Court
Sheffield, S1 4DP, UK
+44 114 222 1800
F.Macias@dcs.shef.ac.uk

Mike Holcombe
University of Sheffield
Regent Court
Sheffield, S1 4DP, UK
+44 114 222 1802
M.Holcombe@dcs.shef.ac.uk

Marian Gheorghe
University of Sheffield
Regent Court
Sheffield, S1 4DP, UK
+44 114 222 1800
M.Gheorghe@dcs.shef.ac.uk

ABSTRACT
This paper describes the current state of an on-going study into an empirical assessment of eXtreme Programming. A pilot study has finished and a rigorous experiment is underway. A detailed description of the pilot study is given together with some details of the data collected so far and an initial analysis is presented. The focus of this report, however, is not the results but the knowledge obtained from the experience from this pilot study which will be utilised in the forthcoming experiments.

Keywords pilot study, extreme programming, empirical software engineering**.**

1. INTRODUCTION

In empirical research of any kind it is important that the experiments are carefully designed, the right sort of data is collected and the analysis of the results is oriented to some coherent purpose. In empirical research into software engineering processes and methodologies it is particularly difficult to do this because of the many parameters and variables that are relevant to the issues under investigation. Poorly defined and executed experiments can be easily challenged and their conclusions rejected. Thus it is vital that suitably designed pilot studies are carried out to articulate the problems that might be faced in a full trial.

A pilot study can help in tuning the method as well as discovering unseen problems and weaknesses in the method. With this study it is possible to predict the usefulness of a set of data or the lack of some data. In the current study the conditions of the experiment are repeated every year so it is possible to carry out the pilot study and one year later to carry out another experiment.

This report describes the current state of a study intended to assess eXtreme Programming. The experimental context is divided into two parts. The background for the hypothesis depicts the field for the further discussion, once the interpretation has been finished. The hypothesis states the core of the study. The formal definition of the experiment provides the required information to distinguish the experiment in accordance with Basili [2], [3]. Some initial observations help in gaining a better understanding of the problem as well as giving a preview of the results that we could expect. A short description of the general method chosen for this study follows. Finally the conclusions describe what was learnt through the pilot study.

2. EXPERIMENTAL CONTEXT

The study involves a comparative experiment involving two types of software development: eXtreme Programming and a design-led traditional approach where testing is not considered at the outset.

The object of the study translated into a formal hypothesis is presented as follows:

Null hypothesis

eXtreme Programming no advantages over traditional design-led approaches for a short-medium size project.

Alternative Hypothesis

eXtreme Programming provides a methodology that enables software developers to get better quality software with less effort than the traditional design-led approach.

3 DEFINITION OF THE EXPERIMENT

According to Basili *et al.* [1] most empirical research

study definitions contain six parts: motivation, object, purpose, perspective, domain, and scope. The **motivation** of this study is to assess eXtreme Programming (XP). The **object** is the complete process of software construction. The **purpose** consists of evaluating the philosophy compared with a traditional design-led approach from the **perspective** of a researcher - an external observer not engaged with the process. As there will be two separate experiments (Genesys and Software Hut) the **domain** of the study involves different teams of people, they differ in size, in skills and all the projects are different. Therefore, the **scope** of the study splits as follows: in the Genesys project several teams working with several projects will be observed. All the projects are different. Every team could be involved with several projects, but a project will involve just one team. This **scope** has been called "multi-project variation". In the Software Hut project several teams working with several projects will be observed. Every team will work on just one project, and one project will be simultaneously addressed by several teams. This **scope** is known as "blocked subject-project". Fenton and Pfleeger [5] provide a classification for projects. From their point of view Genesys is a case study because we have restricted control on the process, there is no replication and there is no control subject, while the Software Hut project is a proper experiment for the opposite reasons, it has replication and there are control subjects.

The formal definition of the experiment is fully detailed in a more extensive document available from the authors. This includes particular goals and detailed definition for the intended metrics. It was not included in this report because of space limitations which will focus on the comparative study in the Software Hut.

4. THE EXPERIMENTAL DESIGN

The experiment splits into two studies [7], [8].

The first one is the Software Hut project. This project is taken by 2nd year undergraduate students during the spring semester. They work for real clients and the client pay a nominal fee. The Software Hut pilot study ran with 80 people distributed in 15 teams and three clients were involved. The students were doing these projects alongside their other courses, they were each supposed to spend 6 hours per week on the project. These projects were tracked by three lecturers. Every client deals with five teams, two or three of them worked with XP and the rest with the alternative. Each client provided a short description of their software needs at the beginning of the semester and this was followed by regular weekly meetings with the teams in order for the teams to understand the clients' requirements. It was not possible to have an *on-site* customer but students were able to meet the client sufficiently often to identify the problem. This process lasted for 5 weeks and then the software was developed, prototypes were demonstrated to clients where appropriate and at the end of the semester every client received five different systems. Even though these systems were different, they carried out the same task. The client did not know that some systems were produced using eXtreme Programming and others were produced using a traditional approach. Finally the client was required to classify the systems from the best to the worst. The client was blind to the methodology followed.

The second study is the Genesys company [6]. This company was set up in 1997. Every autumn its personnel are renewed. Master students and 4th year undergraduate students produce software for real clients. The client pay near commercial rates for the software and can accept or reject it. The previous year's pilot study involved 25 students, and the current year comprises of 33 students. Each team is composed of 3, 4 or 5 students and has its own client. The results from this experiment will be discussed elsewhere. The main focus was on investigating how XP can be introduced into an existing software company.

The **design** of the Software Hut project that ran this spring semester included 15 experimental units distributed in a random balance between two treatments. There was simultaneous replication. The groups were allocated in two blocks, one for every treatment. The first block had 8 teams, and the second 7 teams. In this pilot study two factors were considered. The first one was the time spent on the project for every team and the second was the bulk of work. Special attention was made to the number of requirements.

5. METHOD FOLLOWED

The pilot study has offered the possibility to define, discover and redefine not only the experiment but the process as well, around the goal of the study. The factors chosen for the pilot study came from the challenge to carry out an approximation as close as possible to the actual experiment. But the experiment is driven, after its definition and design, by a set of metrics. These metrics were fixed through the *Goal-Question-Metric* template [2]. Once the goals and hypothesis of the entire project

have been stated, the first set of questions was formulated with the consequent metrics. The subsequent questions came from secondary goals or from the refinement and feedback process for previous questions.

6. THE PILOT STUDY

The pilot study ran between October 2000 and July 2001. The data relative to the Software Hut project was collected in several ways. Reports of meeting and the timesheets that every team had to fill in weekly these were automatically collected on Fridays at 4:00 p.m. Another source of data was the progress reports that the teams had to submit as well as the schedules and plans for their activities. The final report, manual, software, client's evaluation and lecturers' marks were considered as well. There were no verification mechanisms, and some inconsistencies were detected at analysis time. The lesson learned was that data collected must be verified. [1] suggests carrying this out through interviews.

This report on the pilot study includes only two factors: time spent in the activities (measured in hours), and quantity of work. This last one stands for only one subproduct: the requirements as it is one of the factors that we can compare directly, not only in quantity but in quality aspects like level of granularity and confidence. The quantity of time spent by the teams was collected through the timesheets filled-in weekly.

The total time spent by the teams has a standard deviation of 146.2 and mean 196.82. This distribution has 66% of the data at the distance of one standard deviation or closer from the mean, 84% of the data is at two standard deviations from the mean or closer and 90% of the data is at three standard deviations from the mean or closer.

The number of requirements was another factor in the pilot study. The standard deviation of the number of requirements was 12.8 and the mean was 32.9. 6 teams identified 25 requirements, 3 teams had 30 and 7 other teams had requirements ranging from 20 to 60.

This distribution has 72% of the data at the distance of one standard deviation from the mean or closer, 84% of the data is at two standard deviations from the mean or closer and 90% of the data is at three standard deviations from the mean or closer. The distribution of the data per client is closer to the normal distribution than this one.

Once having this data, a relationship between them was investigated, and between some other dependent variables. Looking for such a relationship the correlation coefficient between these two factors was calculated as 0.63. Many other scatter diagrams were produced, as well as the correlation coefficient between the compared data. Some dependent variables were considered, such as the assessment of the client to the manual and the software, and the assessment of the lecturer to the reports. Then three factors were mainly considered: the manual, the external quality factors and the internal quality factors. The client assessed numerically items like comprehensiveness, understandability etc. of the manual. The client assessed the software (the final product) as well. The targets of the clients' evaluation were the external quality characteristics of the product. The lecturers did not evaluate exactly the same aspects because the product documentation differs, but in both cases the internal quality items were evaluated. All these marks were treated as dependent variables.

The time spent and the number of requirements generated were contrasted through the scatter diagrams and correlation coefficient against the data produced by the lecturers and clients.

7. SOME INITIAL OBSERVATIONS

From the data the following it was noticed that the XP teams produced more useful information than the others. There is a weak relationship (the correlation coefficient obtained is 0.63) between the time spent on the project and the quantity of the material generated, this includes the size of the product. All the teams in the Software Hut project were considered. Maybe the set of data requires another kind of manipulation, because one of the factor sets presented a left skewed distribution. This possibility must be considered in the actual experiment. The final product results indicated better quality in the case of XP projects than the other projects.

From the point of view of the client, the manuals written by the XP teams were better than the other teams. Two of the three clients found that the best external quality factors were in the product delivered for XP teams, the third client found best quality in the product delivered by one of the other teams. Apart from this team, the better quality tendency was observed in XP teams. From the point of view of the lecturers the product produced by the XP teams has better internal quality characteristics than the other teams.

8. CONCLUSIONS

The pilot study highlighted faults in the experimental process intended for the current assessment. Some other practices already carried out through the pilot study were evaluated. Some characteristics of the study that were formerly considered harmful, like the non-homogeneous skills of the students, were considered in the pilot study. The pilot study can not be considered a risk analysis study, but it provides information about the viability of the research. Some of the practices required for an empirical assessment were noticed through the pilot study. The most relevant are:

- the need for verification of the data collected
- the need for tracking the progress of the process over short periods of time, e.g. weekly

The pilot study is no guarantee that the actual experiment will run without any problems but it is useful in order to identify in advance some of the likely problems we will face.

9. BIBLIOGRAPHY

1. V. R. Basili, R. W. Selby, D. H. Hutchens; Experimentation in software engineering; IEEE Transactions on software engineering, vol. SE-12, pp. 733-743; Jul. 1986.

2. V. R. Basili, H. D. Rombach; The TAME Project: Towards Improvement-Oriented Software Environments; IEEE Transactions on software engineering, 14(6):758-773; Jun. 1988.

3. V. R. Basili, F. Shull, F. Lanubile; Building knowledge through families of experiments; IEEE Transactions on Software Engineering, 25(4):456-473; Jul-Aug. 1999.

4. K. Beck; 1999; Extreme Programming Explained: Embrace Change; Addison-Wesley; U.S.A

5. N. E. Fenton, S. L. Pfleeger; 1996; Software Metrics: A Rigorous and Practical Approach; 2nd Ed.; International Thomson Computer Press; U.K.

6. Genesys Solutions Web site, On-line at <http://www.genesys.shef.ac.uk>

7. M. Holcombe, M. Gheorghe, F. Macias; Teaching XP for real: Some initial observations and plans; Proceedings XP2001; Sardinia, Italy, May 20-23, 2001; 14-17.

8. F. Macias, M. Holcombe; Experiments of an apprentice of scientist: An empirical assessment of eXtreme Programming; Proceedings ENC'01; Aguascalientes, Mexico, Sep. 15-19, 2001; 875-880.