AN EMPIRICAL EVALUATION OF EVOLUTIONARY DESIGN APPROACH

Design, Results and Discussion of Experiments on Extreme Programming

René Noël, Marcello Visconti, Gonzalo Valdés and Hernán Astudillo
Departamento de Informática, Universidad Técnica Federico Santa María, Av. España 1258, Valparaíso, Chile

Keywords: Extreme Programming, Evolutionary Design, Experimental Design.

Abstract: Evolutionary Design is Extreme Programming's approach to organize software structure and its relationships, encouraging refactoring, test driven development and the simplest solution for the requirements of a single iteration, thus avoiding a big up-front design activity at the beginning of the project that can cause carrying on a huge structural complexity throughout the whole project. In order to contrast this approach with a planned or traditional design approach, an empirical evaluation of impact on software design quality and process productivity has been designed and conducted in an academic environment with toy size problems. Experimental studies planning details are presented, and two replications with different experimental designs are described. Results suggest that there are no differences in quality between both approaches, and that productivity is better when a planned design is adopted.

1 INTRODUCTION

Extreme Programming (XP) (Beck, 1999) proposes an iterative process and an evolutionary design (Fowler, 2004) approach that encourages the design of the simplest solution for the requirements considered in each iteration, without worrying about next iteration requirements and its design complexity. This approach allows embracing change by avoiding big up-front design activity and carrying design complexity throughout the whole project, as traditional software processes proposes.

There are documented cases of development teams that resist to this approach (Harrison, 2003; Keefe and Dick, 2004; Müller and Tichy, 2001; Rasmusson, 2003) questioning XP design practices and their naturalness; besides, a big up-front design is an instance for identifying reusable structures such as architectural or design patterns (Gamma et al, 1995), and may help to improve productivity and product internal quality. Theoretical fundaments can be argued in favour of both design approaches, but Experimental Software Engineering offers a chance to gather empirical evidence on its impact over product quality and process productivity. This paper presents two experimental designs and their results on evaluating the impact of different design approaches over product quality and process productivity. The design topics covered in this article are limited to the detailed design activity (class models, method's algorithms) and not architectural design issues.

Section 2 presents the two experimental designs, and its results. Section 3 discusses the main findings of the study considering both experimental studies, its interpretation on Evolutionary Design context and future work.

2 EXPERIMENTAL DESIGN

In order to compare XP approach with a traditional or planned approach, an empirical evaluation has been designed. The goal of the study is to detect the existing differences on product quality and process productivity when developing with an XP design approach versus a planned design approach.

2.1 Activity Description

The software development activity (which subjects must tackle) was designed considering the practices that must be implemented, and time constraints to execute experiment trials. Design, coding and testing of software features can't take more than two

Noël R., Visconti M., Valdés G. and Astudillo H. (2007).
AN EMPIRICAL EVALUATION OF EVOLUTIONARY DESIGN APPROACH - Design, Results and Discussion of Experiments on Extreme Programming.
In *Proceedings of the Second International Conference on Evaluation of Novel Approaches to Software Engineering* , pages 119-122
DOI: 10.5220/0002586201190122

academic time slots, so we planned a 4-hour activity and design problems that could be solved in that time box. A method based on XP original definition (Beck, 1999; Wake, 2001; Jeffries et al, 2000) was defined. In addition, a variation of the original XP method was defined, in order to incorporate a planned design session at the start of each iteration. Each trial of the experimental study consists of training the subjects on a development method, and then making them apply the method on a given problem.

The application of the methods consists basically of developing a simple software solution, whose requirements are separated in two sets. The whole context of the problem is presented on a global system description that explains the whole functionality and all the domain elements and its relationships. The first set specifies a functionality that adds value to the client, but is not the whole system functionality. The second set complements the first one, and covers all the desired features of the system. By giving a whole system description at the start, planned design subjects can anticipate system design complexity, probably improving its final design quality, and also improving its productivity when identifying already solved problems. If too much time is spent on making this up-front design, productivity could be negatively affected. XP approach can presumably be more productive without a first up-front design, but separating requirements in iterations guarantees that design must be at least reviewed to implement the second set of requirements, and could probably require refactoring the first iteration code.

2.2 Hypotheses and Variables

High level null hypotheses are formulated in order to evaluate the existence of differences on product quality and process productivity using XP approach and planned design approach:

H0': The use of planned design approach produces software with an equal design quality than evolutionary design approach

H0'': The use of an evolutionary software design approach is as productive as a planned approach on an XP project.

In order to test these hypotheses, quality and productivity are measured by using standard metrics. For each of the metrics, a null hypothesis is formulated, for instance:

H0'1: Productivity measured on LOC/minute is the same using evolutionary or planned design approach

Independent variables that can affect quality and productivity are identified:

- Design Approach: is the factor under study, and has two levels: evolutionary approach (XP's approach) and planned approach.
- Participant's Design Experience: is an undesirable factor of influence and the experimental design will focus on minimizing its impact.
- Problem to Solve: If different problems are used on the study, the impact must be minimized or evaluate if the impact on results is statistically significant.
- XP Knowledge: to minimize the influence of this undesirable factor, participants are trained on XP's practices an process, and a guided exercise applying them is performed.

Business logic complexity can be addressed during the detailed design by creating an appropriate structure of classes, methods and interfaces that provides a maintainable and flexible solution, or assigning complexity to a few and highly complex methods, attempting to maintainability and understandability of the code. In order to evaluate software quality, methods' algorithms complexity is measured.

Process Productivity:

- LOC/minute (PTLOC)
- Number of Classes/hour (PNOC)
- Number of Methods/minute (PNOM)

Product Internal Quality (Design Quality):

- Decision Count (DC)
- Maximum McCabe Cyclomatic Complexity (MCC), of the more complex method coded by each subject.
- Number of Code Statements (NSTMNT)

2.3 Original Experimental Design

The original experimental design was presented in (Noël et al, 2005), consisting on a random blocked design, with one factor (the Design Approach) and two levels (XP and planed approach). Participants were undergraduate students that had previously approved an Object Oriented Analysis and Design course. Two blocks were conformed by subjects with design experience and without design experience (marked with different grey levels in table 1), previously evaluated by a survey. Subjects were 31 development teams conformed by a pair of developers, with equivalent design experience. A training session of 1.5 [hours] was performed.

Table 1: Original Experimental Design.

Subject	XP Approach	Planned Approach
1	X	
2		X
…		
30	X	
31		X

Remarkable results of acceptance test are the following:

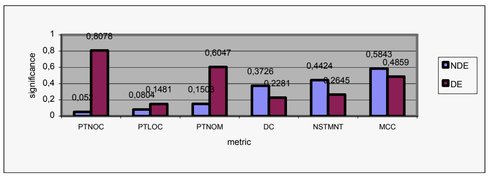

Figure 1: Significance Test Results for Original Experimental Design.

Chart shows the t-Test 2-tailed probability of rejecting null hypothesis when it's true, on subjects blocked by experience on design experienced subjects (DE) and no experienced subjects (NDE). Significant differences on productivity for subjects with no design experience where found ($p<0.07$), obtaining better results with a planned design approach. Differences on software design quality are less significant, but show that experienced subjects produce higher quality software with a planned design approach. The difference between experience blocks leads us to try to mitigate the influence of this factor instead of blocking it, in order to isolate the method effect from the experience effect.

2.4 Second Experimental Design

To mitigate the influence of the subjects' particular experience, each of them must use both design approaches. This implies having two distinct problems to solve, in order to not introduce maturation bias when applying both approaches on the same problem. Thus a 2x2 factorial design with repeated measures is proposed where the factors are the Design Approach and the problem to be solved (Table 2).

Table 2: Second Experimental Design.

Group 1		Group 2	
Method 1	Training	Method 2	Training
	Problem 1		Problem 1
Method 2	Training	Method 1	Training
	Problem 2		Problem 2

To avoid carryover effects, counterbalancing is applied: if the whole Group 1 uses Method 1 in first place and then Method 2, a bias can be introduced due to the learning of tools, programming language or others on the first session. Thus, one half of Group 1 will use method 1 and problem 1, and the other half will use method 2 and problem 2. On the second day, methods and problems will be inverted. Same strategy is applied to group 2. Participants were 22 senior-level students, with Object Oriented Design knowledge and some professional practice. A detailed comparison of the two experimental designs is presented in (Noël et al, 2007).

2.5 Experimental Results Summary

Given the 2x2 factorial design with repeated measures, and the factors Development Method and Problem, 3 sets of hypotheses can be formulated. For Main Effect Development Method:

- *H0: There is no difference between subjects using XP's Design Approach and subjects using Planned Approach with respect to VAR[i]*
- *H1: There is a difference between subjects using XP's Design Approach and subjects using Planned Approach with respect to VAR[i]*

Where i = PTLOC, PNOC, PNOM, DC, MCC, NSTMNT. Similar hypotheses are formulated for Main Effect Problem and Interaction Effect Method X Problem. Looking for method main effect over dependent variables will allow testing hypotheses raised on section 2.2

Data was analysed applying Analysis of Variance for the 2x2 factorial with repeated measures design. All the tests of significance for the influence of factors and interception applied led to the same significance levels for each metric, that are presented on figure 2. Looking at method's effect, we can see that null hypotheses can't be rejected; no differences in quality and productivity between evolutionary and planned design approaches can be demonstrated by the influence of the Method factor. Estimated marginal means for each metric for methods 1 and 2, presented on table 3 suggest that although not statistically significant, the Planned Design Approach (Method 2) is more productive than XP Design Approach (Method 1), for every metric of productivity. Product design quality metrics are favourable to XP Design Approach for Number of Statements and Maximum Cyclomatic Complexity. Decision Count metric suggests that more control flow statements were coded when using XP Design Approach than using a Planned Design Approach.

2.6 Validity Discussion

The main threats to validity are related to three key factors:

Activity Design for Process Implementation. Available time for perform the activity forces us to adapt XP practices and to work with toy problems, so it might be a threat for *construct validity.*

Metrics and Problems Size. Toy problems could be not complex enough to get significant differences on quality or productivity for the chosen metrics, attempting to *construct validity.*

Academic Environment. Students could not be representative of professional developers, attempting to the *external validity* of the study.

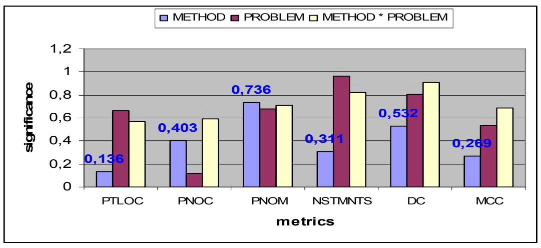

Figure 2: ANOVA Significance Test Results for 2x2 Factorial with Repeated Measures Design.

Table 3: Estimated Marginal Means for Method's Effect.

Measure	Method	Mean	Std. Error	95% Confidence Interval	
				Lower Bound	Upper Bound
PTLOC	1	,458	,053	,335	,582
	2	,568	,080	,385	,751
PNOC	1	1,106	,162	,732	1,480
	2	1,294	,133	,986	1,601
PNOM	1	,079	,020	,031	,126
	2	,090	,025	,033	,146
NSTMNT	1	22,167	2,26	16,957	27,376
	2	26,611	3,76	18,043	35,179
DC	1	3,611	,740	1,905	5,317
	2	3,056	,868	1,054	5,057
MCC	1	3,333	,717	1,680	4,986
	2	4,444	,966	2,216	6,673

3 CONCLUSIONS

Productivity results are consistent between original and second execution of the experimental study: results suggest that a Planned Design approach always yields a better productivity. For quality metrics, in the first experimental study planned design approach yields better quality, but in the second, the results suggest that subjects using XP evolutionary approach get better quality products. However, both quality results are far from being statistically significant, so this suggests that no product design quality differences exists when using distinct design approaches.

When facing the design activity, we can choose between a planned approach or an evolutionary approach. Our study suggests that no significant differences on quality between both approaches can be demonstrated, and that process productivity is better with a planned approach, so we can evaluate the trade-offs of increasing process productivity by planning the design, or empowering the process capability for embracing change through the adoption of XP original design approach, without affecting product design quality.

REFERENCES

Beck, K., 1999. *Extreme Programming Explained: Embrace Change*, Addison-Wesley.

Fowler, M., 2004. Is Design Dead? http://www.martinfowler.com/articles/designDead.html.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J., 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley.

Harrison, N., 2003. A Study of Extreme Programming in a Large Company. Avaya Labs. http://www.agilealliance.org/system/article/file/1292/file.pdf

Henderson-Sellers, B., 1996. *Object-Oriented Metrics, measures of Complexity*, Prentice Hall.

Jeffries, R., Anderson, A., Hendrickson, C., & Jeffries, R. E., 2000. *Extreme Programming Installed*, Addison-Wesley.

Keefe, K., & Dick, M., 2004. Using Extreme Programming in a Capstone Project. In *Proc. 6th Australasian Computing Education Conference, Dunedin*, New Zealand, pp. 151-160.

Müller, M., & Tichy, W., 2001. Case Study: Extreme Programming in a University Environment. In *Proc. 23rd Int'l Conference on Software Eng.*, Toronto, Canada, pp. 537-544.

Nöel, R., Visconti, M., Valdés, G., & Astudillo, H., 2007. Lab. Package for the Investigation about the Impact of Software Design Approaches on XP. http://www.labada.inf.utfsm.cl/~amigosisw/xpdesign_package.

Nöel, R., Astudillo, H., Visconti, M., & Pereira, J., 2005. Evaluating Design Approaches in Extreme Programming. In *Experimental Software Eng. Latin American Workshop*, Uberlandia, Brazil.

Rasmusson, J., 2003. Introducing XP into Greenfield Projects: Lessons Learned. *IEEE Software*, vol. 33, no. 7, pp. 21-28.

Wake, W., 2001. *Extreme Programming Explored*, Addison-Wesley.