Software Development Practices Patterns

Herez Moise Kattan(✉) and Alfredo Goldman

Department of Computer Science, University of São Paulo (IME-USP),
São Paulo, Brazil
{herez,gold}@ime.usp.br

Abstract. Our ultimate goal is to propose a catalog with recommendations on how to organize the work of programmers. In this research we intend to provide experiments to explore the most suitable forms to allow programmers to develop software, either alone, in pair programming or in group. We also explore other approaches like code review. Our goal is not only to reduce the software development cost, but also to improve programmers life quality.

Keywords: Mob Programming · Swarming · Pair programming · Pair and review simultaneous in pairs · Code review · Coding Dojo

1 Introduction

The motivation of our research is to find better ways to organize the programmers work to develop quality software in a productive way suitable to their current context. Our goal is not only to reduce the software development cost, but also to improve the programming experience. Toward to do this a set of unanswered questions related on how many programmers should implement a task emerged:

- When Pair programming should be used?
- When it is interesting to perform Mob programming?
- What are the situations where it is better to do simultaneous work?
- What's the influence of the context and of the team?

2 Description of Points on Which We Would Like to Get the Most Advice on

We would like to have initial hints on when is better to use each one of the techniques and when alternating among them is a good idea. Our research is based upon the process of the Illuminated Arrow (see below).

H. Baumeister et al. (Eds.): XP 2017, LNBIP 283, pp. 298–303, 2017.
DOI: 10.1007/978-3-319-57633-6_23

3 Relevant Prior Work

Herez [9] did an extensive work on when to apply pair programming on several teams. The main conclusions were that pair programming should be applied when the task being developed is more complex or when there is a large gap on the programmers experience. On other situations, other more light techniques like code review can be applied without any drawback.

More recently we started to study also the benefits of Mob Programming.

There are points of convergence in the literature about the advantages of the use of Mob Programming over other techniques [9–13]. On a first experiment we figured out that Mob Programming was not very useful when no one in the team knew the language/framework being used [10].

4 Research Objective

Elaborate a catalog with suggestions on how the programmers should organize their work concerning pair programming and related techniques.

5 Research Approach, Study Design and Arrangements

The interpretation made in an interpretive case study is frequently impossible to be auditing posteriorly and, is very difficult to conduct controlled experiments. For this reason, Kattan [2] suggests to conduct application examples to produce raw data. After, to analysis this data, is suggested the use of the Grounded Theory techniques, to looking for one auditable Theory to explain the findings [5].

There are no silver bullets [6], but maybe together we could build illuminated arrows that somehow inspire the correct path to innovators. Figure 1 show the phases of this research method, that reduces the gap between software developers and academic researchers and, thus, produce more ready to use knowledge. The Illuminated Arrow [2] proposed application examples to deepen impartially the initial work of an action research, supported by systematic and tertiary revisions [4].

In Software Engineering it is very difficult to conduct controlled experiments or make convincing Double Blind experiments [3]. Furthermore, human expertise and human subjectivity interfere with the result of experiments. The types of software are very different, each software is unique, it depends on the problem it solves, so is different from medical research, where every human being has blood, lung, heart, brain, etc [2].

The reason to start with an action research is to fix the initial mistakes of the research and to be sure about the benefits and limitations of it. If the result of the initial work, is considered positive, the next step suggest by the Illuminated Arrow is systematically review the literature, making it easier to audit.

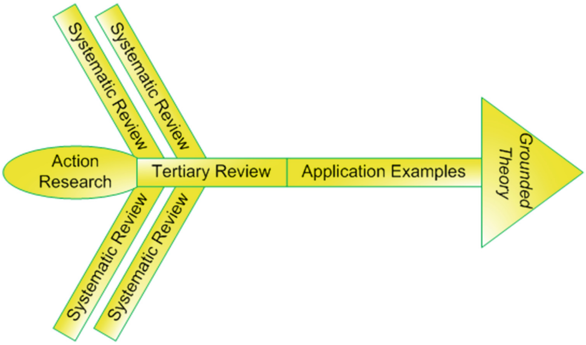

Fig. 1. Phases of illuminated arrow, starting from left and finishing in the right [2].

6 Action Research and Application Examples

The Empirical Study occurs twice. The first is in the beginning of the research as suggested by illuminated arrow, because start with an action research helps to deep the knowledge on this theme. Thus, makes easier the identification of some aspect possible to be improved and will guide the systematics reviews.

The second time, occurs after the literature review and is the empirical study by application examples. Thus, makes easier audits compared with interpretative case studies usely used. The applications examples will be careful design based at the literature reviews and action research.

These application examples will produce raw data about what we observe, toward to confirm and validated some aspects, provide new ideas and these raw data produced we hope that permit emerge one Theory in the way of one recommendation system to software developers about the better set of practices based on a specific context.

7 Data Analysis Methods and Techniques

The use of grounded theory is founded on the premise that the generation of theory at various levels is indispensable for a deep understanding of social phenomena [7,8]. The techniques of data analysis in grounded theory are:

- coding data (that comprises open, axial and selective):

  Open coding, to find categories;

  Axial coding, to find links between the themes/categories;

  Selective coding, to find the core category.

- memo writing;
- theoretical sampling.

8 Summary of the Current Status of the Research and Planned Next Steps

This proposal research is the continuation of Kattan [9] master's thesis. The technique is called Programming and review simultaneous in Pairs, is one extension to the pair programming. It's concluded when the goal is to reduce the time-to-benefit suggest use the Programming and review simultaneous in Pairs, when the pair is compose by professionals with the follows experience levels: intermediate and senior, or senior and senior, or junior and junior. The complexity of these tasks were classified as: low, medium and high.

Kattan reviewed the Mob Programming literature too in his master's dissertation and also applied Mob Programming in one application example.

Figure 2 illustrates the extension to pair programming, was used aspects of Simultaneous Engineering [9] to create one alternative to pair programming. The phases 1, 2, 3, 4, 5 and 6 are illustrated in Fig. 2. Phase 7 is illustrated in the form of the team with the work, because is the reflective rest and conflict resolution, is unformatted due to the miscellaneous possibilities for reflective/productive rest and conflict resolution.

The current status of the research and planned next steps are:

- We are conducting in companies experiments on Mob Programming, Programming and review simultaneous in Pairs, Pair Programming, Code Review and Coding Dojo [1].
- We are continuously reading the live science of this theme in literature in a frequently updating process.
- Beyond the use of questionnaire, we are analysing possible metrics [9].
- Based on feedback of international community we will rock the research and start the data collection.
- After conducting field studies, called here of application examples, we will analyse the data using Grounded Theory techniques.

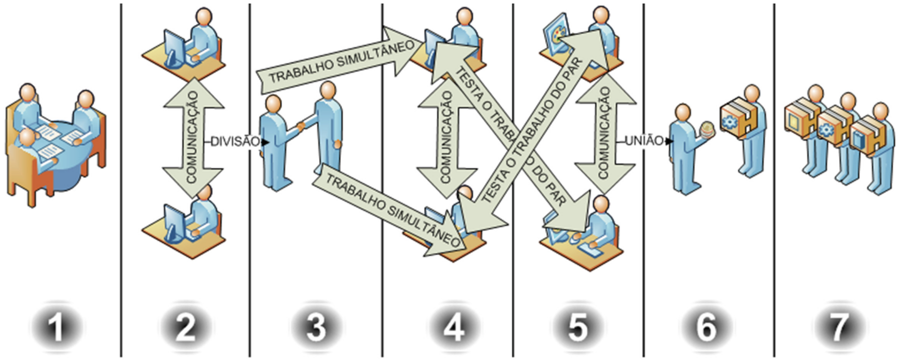

Fig. 2. Programming and review simultaneous in Pairs

References

1. Rooksby, J., Hunt, J., Wang, X.: The theory and practice of randori coding dojos. In: Agile Processes in Software Engineering and Extreme Programming: Proceedings of the 15th International Conference, XP 2014, Rome, Italy, vol. 179, pp. 251–259, 26–30 May 2014
2. Kattan, H.M.: Illuminated Arrow: a research method to software engineering based on action research, systematic review and grounded theory. In: CONTECSI - International Conference on Information Systems and Technology Management 2016, pp. 1971–1978, 21 July 2016
3. Budgen, D., Charters, S., Turner, M., Brereton, P., Kitchenham, B., Linkman, S.: Investigating the applicability of the evidence-based paradigm to software engineering. In: Proceedings of WISER Workshop, ICSE 2006, pp. 7–13. ACM Press, May 2006
4. Kitchenham, B., Charters, S., Budgen, D., Brereton, P., Turner, M., Linkman, S., JØrgensen, M., Mendes, E.: Guidelines for performing systematic literature reviews in software engineering, version 2.3. EBSE Technical Report EBSE-2007-01, Software Engineering Group, School of Computer Science and Mathematics, Keele University Keele, Staffs ST5 5BG, UK and Department of Computer Science, University of Durham, Durham, UK, 9 July 2007
5. Allan, G.: The legitimacy of grounded theory. In: Proceedings of Fifth European Conference on Business Research Methods, pp. 1–8 (2006)
6. Brooks, F.: The Mythical Man-Month: Essays on Software Engineering, 20th Anniversary Edition, 322 pages. Addison-Wesley, Reading (1995)
7. Glaser, E.G.: Advances in the Methodology of Grounded Theory: Theoretical Sensitivity. Sociology Press, Mill Valley (1978)
8. Glaser, E.G., Strauss, A.L.: The Discovery of Grounded Theory: Strategies for Qualitative Research (1967)
9. Kattan, H. M.: Programming and review simultaneous in Pairs: a pair programming extension. Master dissertation, Institute for Technological Research of the State of São Paulo (IPT) (2015). http://aleph.ipt.br/F or http://ipt.br, click on: Online Consultations, then click on: Library
10. Questionnaire. http://ccsl.ime.usp.br/wiki/SwarmQuestionnaire
11. Wilson, A.: Mob programming - what's works, what's doesn't. In: Agile Processes in Software Engineering and Extreme Programming: Proceedings of the 16th International Conference on Agile Software Development, XP 2015, Helsinki, Finland, pp. 319–325, 25–29 May 2015
12. Griffith, A.: Mob programming for the introverted. Experience report, Agile (2016)
13. Hohman, M., Slocum, A.: Mob Programming and the Transition to XP (2001)