Agile processes evolution challenges

Ayed, Hajer; Habra, Naji; Vanderose, Benoît

Publication date:
2013

Document Version
Peer reviewed version

Agile processes evolution challenges

Hajer Ayed, Naji Habra and Benoît Vanderose
PReCISE Research Center, University of Namur, Belgium
hajer.ayed@unamur.be —naji.habra@unamur.be —benoit.vanderose@unamur.be

I. Introduction

In the last few decades, agile processes have grown increasingly popular among the software engineering community. The rationale for using agile in many environments is to allow change. Indeed, the traditional methodologies have the implicit assumption that requirements can be final and that only minor variations can be accommodated later. Conversely, the agile methodologies assume that change is inevitable all along the software development life-cycle and thus, encourage rapid and flexible response to it. The core value of the agile paradigm is therefore to enable change management and not to prevent it.

Paradoxically, the agile practitioners promote the evolution and adaptation of the software but do not focus enough on the changes that may affect the process itself. A common misuse of agile is to adopt a ready-made method without much discernment and that, just because it worked well elsewhere. To avoid failure, most experts and researchers recommend process adaptation and evolution in order to make it suitable to the specific circumstances of the organisation and project. There's a large spectrum of techniques that can be used to support agile process evolution. This paper investigates a model-driven approach for agile processes evolution and assessment. This approach promotes the co-evolution between the software products and their process.

II. Key challenges

A. Agile process customisation and assessment

Practitioners very often experience the challenge of distinguishing convenient agile techniques and practices based on their culture, their values, and the types of systems that they develop. This procedure results in a context-specific process that combines two or more ready-made agile practices and/or blends agile and non-agile practices. This kind of evolution is called agile process customisation. Another important challenge is agile process assessment : the extent to which the process meets the needs of the project should be confirmed.

Most studies that have been undertaken on agile methods customisation are specific to a particular situation and concentrate on reporting the organisation way of customising[1][2]. The problem with such studies is that they are hardly reusable and/or generalisable, since no automation techniques are provided. Few studies such as [3] advocate formal methods for initial agile processes adaptation but do not provide support for later evolution of the agile process model.

Most of the existent agile assessment approaches (such as [4][5]) focus on the agile or plan-driven practices selection based on a comparative analysis. Many of them are also limited to the working software scope (e.g., assessment of the iteration velocity, assessment of the product quality, etc.) [6] and do not provide any support for the enacted process assessment. Such approaches provide a good starting point but cannot be used for assessing the suitability of the enacted agile processes either they are customised or not.

B. Model-driven Process Evolution

Researchers from both industry and academia have pointed out that software processes, including agile processes, need to be rigorously defined through relevant models, in order to support and facilitate their understanding, assessment and automation. We should also be able to analyse them through metrics and well defined quality assessment based analysers, in order to be able to improve them iteratively.

Moreover, in order to model the evolution of the process overtime and the co-evolution with the software, we need to be able to capture the interactions between the modelled process and the enacted process. We therefore need to raise the abstraction level and design an agile processes metamodel.

In fact, despite the differences in fine-grained details, all agile processes follow a common paradigm and can be described by a generic agile processes metamodel. We have found few studies about agile process metamodelling in the literature [3][7] and none of them is targeted for process evolution or assessment.

For example, [3] proposed a metamodel for partial agile method adaptation. This research aims at the description of a formal roadmap of how to configure a method for a partial adaptation, i.e., how agile methods can be broken down into a set of elements and how they can be combined using techniques like merging and generalizing similar elements. The proposed metamodel focuses primarily on partial agile method composition and do not address any evolution issue.

In order to effectively support the goals of modelling analysis and automation, an ideal agile process metamodel must exhibit several characteristics. To give but

a few examples :

- It must describe the activities, practices, stakeholders and the expected resulting artefacts
- It must describe evolution and assessment metrics that will be used as an input for process analysers
- It must provide means to capture the dynamic behaviour, for example, by clearly defining the interaction between stakeholders and the operations they perform.

III. AM-QuICK: A MODEL-DRIVEN APPROACH FOR AGILE PROCESSES EVOLUTION

In order to address the challenges discussed in II, we investigated in [8] and [9] a model-driven approach for agile methods evolution that we called AM-QuICK : Agile Methods Quality-Integrated Customisation Framework. AM-QuICK aims to continuously assist agile methodologists, i.e., during the design of the process in the organisation level and throughout its enactment in the process level (Fig.1). The process design is performed thanks to an agile meta-model adapted from the more generic process metamodels SPEM , SMSDM and OPF [10]. More details about this metamodel can be found in [9].

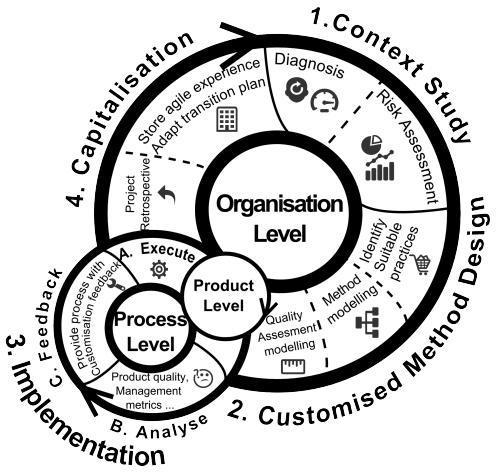

Figure 1. AM-QuICk overview

The AM-QuICk lifecycle, depicted in Fig. 1, consists of three cycles, each corresponding to one level : the first cycle handles the organisation strategy high-level (i.e., the agile transition strategy, the agile values in the business level, the agile culture adoption as an organisation shift of thinking, etc.) and occurs once during one project; the second is for process refinement and takes place continuously during the process execution and the third concerns the working product evolution.

This third cycle addresses the co-evolution between various software products. Any effort provided to ensure this form of co-evolution is therefore reflected at this level. In the meantime, any change monitored on the products is also taken into account in the second lifecyle and may result in a revision of the process for the next development or maintenance iteration. This way, the framework we propose extends the notion of co-evolution to the process itself, allowing to review the way software is developed or maintained, based on the last evolution of its constitutive products.

IV. CONCLUSION

In order to ensure the agile processes evolution and to assess their suitability, researchers form both academia and industry highlights the need to model them rigorously and to assess their suitability through metric-based analysers.

In this paper, we introduced a model-driven approach to support agile processes design and evolution according to environment changes. The approach implements a generic agile processes metamodel that will serve as a basis for specific processes composition.

The metamodel should evolve in the future, in order to include dynamic behaviour between its elements, so it is able to represent the interaction between various software products and their related processes (and therefore their co-evolution). This will also allow to support working product evolution as data are gathered during process enactment and vice versa.

REFERENCES

[1] B. Fitzgerald, N. Russo, and T. O'Kane, "Software development method tailoring at motorola," *Communications of the ACM*, vol. 46, no. 4, pp. 64–70, 2003.

[2] L. Layman, L. Williams, and L. Cunningham, "Exploring extreme programming in context: An industrial case study," in *Agile Development Conference, 2004*. IEEE, 2004, pp. 32–41.

[3] G. Mikulėnas, R. Butleris, and L. Nemuraitė, "An appraoch for the metamodel of the framework for a partial agile method adaptation," *Information Technology And Control*, vol. 40, no. 1, pp. 71–82, 2011.

[4] B. Boehm and R. Turner, *Balancing agility and discipline: A guide for the perplexed*. Addison-Wesley Professional, 2003.

[5] P. Lappo and H. C. Andrew, "Assessing agility," in *Extreme Programming and Agile Processes in Software Engineering*. Springer, 2004, pp. 331–338.

[6] S. Soundararajan and J. D. Arthur, "A structured framework for assessing the "goodness" of agile methods," in *Engineering of Computer Based Systems (ECBS), 2011 18th IEEE International Conference and Workshops on*. IEEE, 2011, pp. 14–23.

[7] E. Damiani, A. Colombo, F. Frati, and C. Bellettini, "A metamodel for modeling and measuring scrum development process," *Agile Processes in Software Engineering and Extreme Programming*, pp. 74–83, 2007.

[8] H. Ayed, B. Vanderose, and N. Habra, "Am-quick : a measurement-based framework for agile methods customisation," in *IWSM-MENSURA*, 2013.

[9] H. Ayed., B. Vanderose., and N. Habra, "A metamodel-based approach for customizing and assessing agile methods," in *The International Conference on the Quality of Information and Communications Technology (QUATIC)*, 2012.

[10] B. Henderson-Sellers, C. Gonzalez-Perez, and J. Ralyté, "Comparison of method chunks and method fragments for situational method engineering," in *Software Engineering, 2008. ASWEC 2008. 19th Australian Conference on*. IEEE, 2008, pp. 479–488.