Change-Impact driven Agile Architecting*

Jessica Díaz, Jennifer Pérez, Juan Garbajosa, Agustín Yagüe
Technical University of Madrid (UPM) - Universidad Politécnica de Madrid
CITSEM, Systems & Software Technology Group (SYST), ETSISI, Madrid, Spain
{jdiaz,jenifer.perez,jgs}@eui.upm.es, agustin.yague@upm.es

Abstract. This summary presents a solution based on the use of change-impact knowledge as the main driver for agile architecting. The solution consists of a Change Impact Analysis technique and a set of models to assist agile architects in the change (decision-making) process by retrieving the change-impact architectural knowledge resulting from adding or changing features iteration after iteration.

Keywords: Agile software development, agile architecting, change-impact analysis

1. Summary

Software architecture (SA) is a key factor to scale up Agile Software Development (ASD) in large software-intensive systems. Aligning fruitfully SA and ASD requires leveraging the inherent qualities of architectures (e.g. abstraction, communication, analysis) while complying with agile principles (e.g. open to change). This alignment can be achieved as long as practitioners are able to count on mechanisms for enabling: (i) Incremental design of features. (ii) Accommodation of new features or customizations on existing features. We refer to both of them as *agile architecting*.

Mechanisms for enabling agile architecting should assist and guide agile architects, specifically in (i) the decision-making process of implementing changes in each agile iteration, and (ii) the maintenance of the architecture integrity, i.e. the preservation of earlier architectural design decisions iteration after iteration. Regarding the former, the knowledge about the effects of a change upon the architecture provides architects with information that can be advantageously deployed to reason about how and where to implement that change. It also allows architects to make better evolution decisions based on risks, cost or viability of the change. Regarding the latter, the continuous process of architecting should never result in the software degradation as a consequence of intentionally or accidentally violation of earlier design decisions or constraints. In this sense, agile architects need knowledge about dependencies between design decisions, constraints, tradeoffs, etc., which can assist them in countering or even avoiding several well-known negative effects of software evolution—e.g. architectural erosion.

*This communication is a summary of the following published article:Díaz, J., Pérez, J., Garbajosa, Yagüe A.: Change-Impact driven Agile Architecting. Proceedings of the 46th Hawaii International Conference on System Sciences (HICSS '13) doi: http://dx.doi.org/10.1109/HICSS.2013.127 IEEE Computer Society ISBN 978-0-7695-4892-0 (2013) pp. 4780-4789.

The presented solution[1] uses *change-impact* architectural knowledge as the main driver for agile architecting. This solution provides agile architects with knowledge to (i) assist and guide them in the *change (decision-making) process*, and (ii) favor the preservation of the architecture integrity during the iterative architecting process. This knowledge results from analyzing the impact that changes—feature increment and/or evolution—introduce into the architecture, iteration after iteration in an agile process. The solution consists of a *Change Impact Analysis* (CIA) technique and modeling artifacts for: (i) documenting architectural knowledge—the design decisions and rationale driving the iterative architecture solution—, and (ii) tracing architecturally significant features with their realization in the architecture. These models are traversed using the proposed CIA technique to retrieve the architectural design decisions and architectural components and connections that are impacted as a consequence of changing features. Therefore, these models promote communication and support (semi-)automatically reasoning over the space of architectural knowledge. This solution was built on previous work to design iteratively and incrementally software architecture (see FPLA[2]) and deployed in Scrum (see Fig. 1).

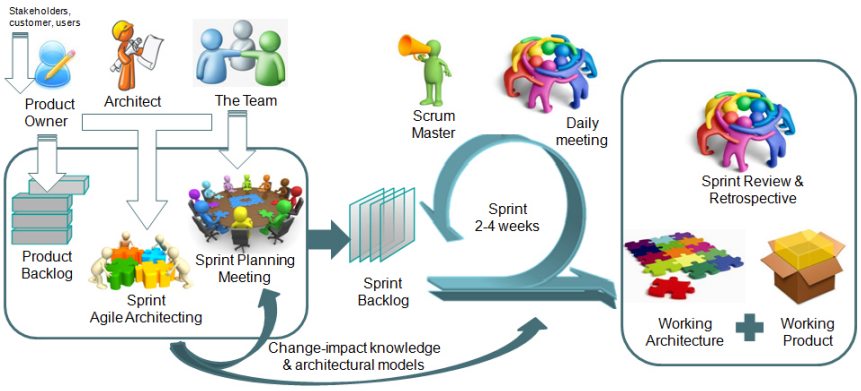

Fig. 1. A tailored Scrum with Agile Architecting

The novelty of this contribution was to prove how the output from a CIA technique can be effectively used to assist and guide agile software architecting. To empirically validate our approach we conducted a case study about smart grids in a software factory, combining both academic and industry efforts. The results prove that (i) the CIA algorithm is effective in locating the impact resulting from a change, and (ii) this change-impact helps architects to take better decisions, especially when architectural knowledge may be lost or vaporized as a result of a staff turnover. These promising results did not interfere with other agile practices and did not incur a big upfront design, making the agile construction and evolution of architecture possible. One of the main conclusions of this research is that agile architecting is feasible so that ASD can be scaled up to large and complex software-intensive systems.

[1] Sponsored by the Spanish fund (INNOSEP TIN2009-13849, IMPONET TSI-020400-2010-103, i-SSF IPT-430000-2010-038, and NEMO&CODED ITEA2 08022 IDI-20110864).
[2] The modeling framework FPLA is available on https://syst.eui.upm.es/FPLA/home