Towards Introducing Agile Architecting in Large Companies: The CAFFEA Framework

Antonio Martini(✉), Lars Pareto, and Jan Bosch

Computer Science and Engineering, Chalmers University of Technology,
Hörselgången 5, 417 56, Göteborg, Sweden
{antonio.martini,jan.bosch}@chalmers.se

Abstract. To continuously deliver value both in short-term and long-term, a key goal for large product lines companies is to combine Agile Software Development with the continuous development and management of software architecture. We have conducted interviews involving several roles at 3 sites from 2 large companies employing Agile. We have identified current architect roles and gaps in the practices employed at the organizations. From such investigation, we have developed an organizational framework, CAFFEA, for Agile architecting, including roles, teams and practices.

Keywords: Agile architecture · Agile software development · Organizational framework · Architect roles · Software process improvement

1 Introduction

Large software industries strive to make their development processes fast and more responsive, minimizing the time between the identification of a customer need and the delivery of value. Short term responsiveness is given by Agile Software Development (ASD) [1]. A gap in the current Agile frameworks is the lack of activities to enhance agility in the task of developing and maintaining software architecture (*Agile architecting*), necessary for long-term responsiveness [1][2]. The role of architects becomes crucial, but there is a lack of knowledge, in literature, on how such roles are implemented in ASD. Therefore, the research questions that we want to inform are:

RQ1 What are the challenges in conducting architecture practices in Agile software development employed in large software product line organizations?

RQ2 Which roles and teams are needed in order to mitigate the challenges in conducting architecture practices in large product line organizations employing Agile?

We have combined literature review, interviews involving several roles in large product line companies employing Agile Software Development and a combination of structured inductive and deductive analysis in order to find the gaps in the architect roles and their activities. We have developed an organizational framework, *CAFFEA* (Continuous Architecting Framework For Embedded software and Agile), comprehending roles and teams to address the challenges related to the architecture practices in ASD.

C. Lassenius et al. (Eds.): XP 2015, LNBIP 212, pp. 218–223, 2015.
DOI: 10.1007/978-3-319-18612-2_20

2 Research Design

We have employed an embedded multiple-case study [3], where the unit of analysis is an (sub-part of the) organization: the unit needed to be large enough, developing 2 or more sub-systems involving at least 10 development teams. We selected, following a literal replication approach [4], 2 companies: A and B (3 sub-cases) large organizations developing software product lines, having adopted ASD and had extensive in-house embedded software development.

As for data collection, we selected [5] as an up-to-date (2008) and comprehensive categorization of "*what do software architects really do*". From such classification, we conducted a literature review for each class of practices, selecting the articles containing condensed knowledge. Then we conducted 3 in-depth sets of interviews involving 3 of the cases, in particular A, B_1, and B_2. The interviews lasted 4 hours and involved developers, testers and architects responsible for different levels of architecture (from low level patterns to high level components). During the interviews we assessed if the architecture practices found in step 1 were carried out, who was responsible, and what challenges they were facing. With this step we identified the current gaps in ASD with respect to architecture management.

The interviews and workshops were recorded and transcribed. The analysis was done following an approach based on Grounded Theory [6], alternating structured inductive and deductive techniques (described below) and using a tool for qualitative analysis, to the trace the code to the quotations.

A preliminary evaluation of the framework is being carried out by the authors, but we could not report the data here for lack of space.

3 Results

First we show the identified architect roles in the companies, highlighting the challenges connected to such roles. We have divided the challenges in 4 main groups: *risk management*, *architectural decisions and changes*, *providing architectural knowledge* and *monitor the current status of the system*. Then we present the teams, the organizational mechanism to address the challenges involving more than one role. The overall components and framework CAFFEA is visible in Figure 1.

3.1 Architects

3.1.1 Chief Architect (CA)

The main role of the CA is to take high-level decisions and to drive the rest of the architects and the Agile teams in order to build an architecture able to support strategic business goals. In all the organizations that we have studied, the role of CA is present and well recognized, and there are few challenges related to ASD.

Risk management - The CA is usually not directly involved in the detailed development: however, in order to take decisions on feasibility and to assist the sales unit with technical expertize, the CA needs to elicit the information about the current

status of the system. The current challenge is the lack of such reliable information and therefore the risk of taking business decisions based on wrong assumptions made on the system.

Monitoring the current status of the system (communication input) - As mentioned before, the current communication practices lack good mechanisms for providing input to the CAs to take informed decision and to address past erroneous decisions (e.g. tool chains not working as expected).

3.1.2 Governance Architect (GA)

We found that the key for the scalability of Agile architecting in a large setting is an intermediate role between the CA and the teams. Such role, (the Governance Architect, GA) functions as a coordinator and support, giving strategic directions for a group of Agile teams developing features within the same (sub-) system. Many architecture practices were mapped by the informants to this role as the main responsible, and we found many challenges in the current organizations. Such role is not always formally recognized: this causes lack of coordination among isolated teams, which favors the accumulation of architectural debt. Also, the non-recognition of this role leads to the lack of resources allocated for carrying out the needed architecture practices.

Risk management - The prioritization of short-term and long-term goals in the team is done by Product Owners through the backlog of the teams. However, such risk management activity usually leads to the down-prioritization of refactoring and architecture improvements, especially the long-term ones. A GA is needed to participate in prioritization to balance the focus between feature development and the long-term goals.

Managing decisions and changes - The architecture needs to support several features and the safe cooperation of the Agile teams. The investigation highlighted either the lack of such responsible for inter-feature architecting or the lack of communication and cooperation between the GA and the Agile teams.

Providing Architecture Knowledge (communication output) - With the shift to ASD, in some of the organizations (B_1 and B_2) the teams have changed from "component teams" to "generalized teams", free to change any part of the code given a feature to be implemented. However, such approach caused, in the teams, a lack of deep expertize about the components. The role of GA becomes therefore critical for assisting the teams and maintaining the architecture, both with face-to-face communication but also supported by documentation when the architecture knowledge is complex and extensive.

Monitoring the current status of the system (communication input) - One of the most emphasized challenges during data collection was the accumulation of architectural debt [7]: the implementation in the code quickly drifted away from the architecture defined and used for strategic decisions and risk management by the CA and other management activities. GAs need to monitor and react to architecture erosion and need for evolution, together with the support of TAs in the Agile teams.

3.1.3 Team Architect (TA)

The TA, the responsible for the architecture in the FT, is often present in the current organizations in the form of a technical leader or experienced developer. Such role is however not formally recognized, which bring the lack of responsibilities for the architecture practices in some of the teams.

Risk management - A challenge was the lack of participation of the team in risk management activities, such as tracking and reporting risky technical debt accumulated during the iterations (activity led by the TA) or to represent the interest of the teams in feasibility discussions with CA, GA and Product Owners (participation of TA).

Providing Architecture knowledge (communication output) - As mentioned for the CA and GA, the lack of capillary spread of architecture knowledge need to be mitigated by a peer in the team, which has been identified with the presence of TA, who would transfer the architectural knowledge from GAs.

Monitoring the current status of the system (communication input) - We found a lack of responsibilities, in the team, about tracking and reporting the status of technical debt that might affect other FTs. The TA would cover such responsibility, as well as lifting proposals for architecture evolution.

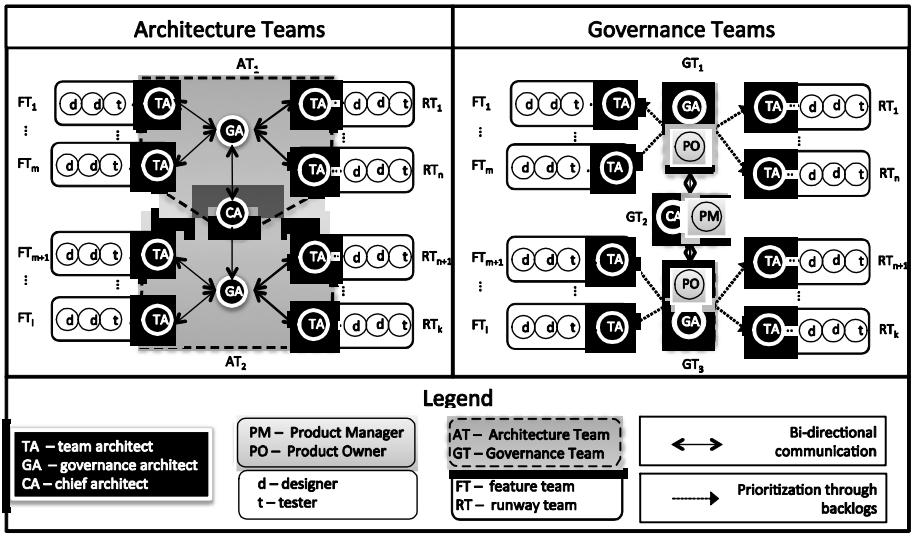

Fig. 1. The components of CAFFEA: teams, roles and their relationships

3.2 Teams

Analyzing the current gaps and the relationships among the architect roles previously mentioned, we found that most of the practices need the roles to coordinate and cooperate in order to mitigate the challenges. To achieve such coordination, suitable organizational mechanisms are non-permanent teams responsible for such practices visible in Figure 1. A special case is the Runway Team (RT), which involves a whole Agile team (see next section).

3.2.1 Runway Team (RT)

As mentioned about the GA and also confirmed by [7], a challenge in the studied companies is the down-prioritization of long-term refactorings or architecture improvements, causing the constant accumulation of architectural debt leading to responsiveness crisis. Such refactorings cannot be prioritized as stories in the backlog of the Agile teams, and therefore remains excluded from the development. In order to conduct such refactorings, a whole Agile team needs to be dedicated for one or more sprints to focus on the "architecture feature" rather than on customer-related features. The RT can be appointed dynamically by a team of Product Owners and architects (see "Governance Team") together, when a long-term refactoring is needed. RTs are visible on the right in Figure 1.

3.2.2 Architecture Teams (ATs)

Most of the architecture practices need coordination and collaboration among different architects in Architecture Teams (ATs in Figure 1): for example, in *monitoring the current status of the system*, no single architects can have all the information needed: the system might have different inconsistencies with architecture at different levels (e.g. low-level design and high-level components). Coordination is also important for spreading the architecture knowledge, from high level concepts expressed by the CA to low level design implemented by the teams and known by the TAs. Also when assessing the risk of architectural debt and taking decisions about solutions and changes, for example the prioritization of refactorings, the architects need to have resources allocated together for communication, analysis and tools.

3.2.3 Governance Teams (GTs)

For those practices regarding "risk management" and "architecture decisions and changes", we found a strong relationship between the architects and the Product Owners or higher-level Product managers. The risk assessment of architecture changes and decisions determines the ratio of resources allocated to the improvements or of the architecture with respect to the resources used for feature development. We found the need, in the organizations, of a team involving Architects and Product Owners or Managers (Governance Teams on different levels, as illustrated in Figure 1) with the responsibility of strategically prioritizing the backlogs of the teams (dotted arrows in Figure 1) between features and architecture improvements, in order to balance the short-term with the long-term value output.

3.3 Framework CAFFEA

The framework CAFFEA is the overall framework of roles, teams and practices. A representation is shown in Figure 1, which combines the visualization of different views: the relationships among the organizational components (architects, managers, teams) with respect to different perspectives (*Architecture* and *Governance*). Figure 1 shows also the communication needs by the architect roles (central area on the *Architecture Perspective*), between the roles and the Agile teams (left) and among the different GTs (*Governance Perspective*). Figure 1 shows the prioritization relationships among the roles and the teams (dotted arrows) and outlines, in both the perspectives, the RTs, our new concept for some of the Agile teams.

Several companies are adopting the CAFFEA framework in practice and we are currently evaluating its application through several case-studies.

4 Discussion and Conclusions

Our work takes inspiration from Leffingwell's work [8] and the concepts of architecture runway. However, the work done by Leffingwell is not supported by scientific investigation following a rigorous research process. Kructhen, in [5], defines several anti-patterns for software architects, based on several experiences in architecture teams. However, the anti-patterns are not specific for the context that we have studied.

The short-term responsiveness in delivering value offered by ASD needs to be enhanced, in large software organizations developing embedded software, by Agile architecting, the management of a software architecture supporting long-lasting responsiveness. We contribute by highlighting current challenges with respect to architectural practices (RQ1): such gaps point at the need for specific architect roles; Team architects, Chief architects and especially important is the Governance Architect, an intermediate key role for coordinating Agile architecting and scaling Agile in large organizations. Such architect roles need organizational mechanisms to cooperate, Architecture Teams, and to interface with Product Management for prioritization and decisions. We developed the CAFFEA framework, including roles, teams and practices, to give support for Agile architecting (RQ2). Such framework, given the current identified gaps, has a specific focus on architecture technical debt management and is being applied and evaluated in practice by several companies.

References

1. Dingsøyr, T., Nerur, S., Balijepally, V., Moe, N.B.: A decade of agile methodologies: Towards explaining agile software development. J. Syst. Softw. **85**(6), 1213–1221 (2012)
2. Daneva, M., van der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., Ajmeri, N., Ramteerthkar, U., Wieringa, R.: Agile requirements prioritization in large-scale outsourced system projects: An empirical study. J. Syst. Softw. **86**(5), 1333–1353 (2013)
3. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. **14**(2), 131–164 (2008)
4. Yin, R. K.: Case Study Research: Design and Methods. SAGE (2009)
5. Kruchten, P.: What do software architects really do? J. Syst. Softw. **81**(12), 2413–2416 (2008)
6. Strauss, A., Corbin, J. M.: Grounded Theory in Practice. SAGE (1997)
7. Martini, A., Bosch, J., Chaudron, M.: Architecture Technical Debt: Understanding Causes and a Qualitative Model. In: 40th Euromicro Conference on Software Engineering and Advanced Applications, Verona, pp. 85–92 (2014)
8. Leffingwell, D.: *Scaling Software Agility: Best Practices for Large Enterprises*. Pearson Education (2007)