Chapter 12
Software Architecture Practices in Agile Enterprises

Veli-Pekka Eloranta
Tampere University of Technology, Finland

Kai Koskimies
Tampere University of Technology, Finland

ABSTRACT

This chapter is based on the results of a survey carried out in 11 IT companies in Finland in Fall 2010. In this survey, the existing practices regarding software architecting work in agile enterprises using Scrum are mapped out. Four main practices to cope with software architecture in Scrum are identified and analyzed. The adoption of these practices is discussed in relation with the characteristics of the teams and project types. Further, the interaction of these practices and Scrum patterns is analyzed. The results indicate that most of the found practices are in conflict with Scrum. The analyzed relationships between Scrum patterns and the identified architecture practices help to understand how software architecture work is aligned with Scrum in real life, as well as the problems of the practices from the Scrum point of view.

INTRODUCTION

During the last decade, software industry has widely embraced agile approaches in the development of software systems. Today, the most popular agile approach applied in industrial software development projects is Scrum (Nord & Tomayko, 2006); according to a recent survey (VersionOne, 2010) Scrum has 78% market share of agile methods in software industry. Even though Scrum is not particularly intended for software development, it has proved to be very suitable for this kind of production work, improving the quality of the products as well (e.g. Huo, Verner, Zhu, & Babar, 2004; Korhonen, 2010). Scrum combines agile principles (Agile Alliance 2001) with lean manufacturing philosophy (Poppendieck & Poppendieck, 2006). Essentially, Scrum defines how the work is decomposed into tasks, how the tasks are ordered, managed, and carried out, which are

DOI: 10.4018/978-1-4666-2199-2.ch012

the roles of the project members, and how persons in these roles interact during the development process. A central concept of Scrum is a sprint, a working period during which the team produces a potentially shippable product increment.

In software engineering, agile approaches are often contrasted with plan-driven approaches. This is sometimes misinterpreted to imply that planning is not required in agile software development. Obviously, careful planning is required in most industrial software development projects involving tens, hundreds, or even thousands of persons. However, instead of a rigid, detailed pre-existing work plan enforced during the entire process, in agile development there is typically only a fairly loose up-front project plan, and the plan is allowed to evolve and become more precise during the process.

Software architecture is traditionally defined as the high-level organization of a system (e.g. Bass, Clements & Kazman 2003; "ISO/IEC 42010", 2011). However, from the viewpoint of a software development project, software architecture is a plan of a new software system, in the same sense as a blueprint is a plan of a house. This creates certain tension between software architecture and non-plan-driven agile philosophy. Indeed, the role of software architecture design was questioned by the early advocates of agile approaches, claiming that software architecture emerges without up-front design efforts. Later, the role of software architecture in agile projects has been studied (Abrahamsson, Babar & Kruchten, 2010; Kruchten, 2010; Nord et al., 2006), but there is still considerable variation among practitioners concerning the ways architecture is involved in agile development. This is particularly visible in Scrum, which itself does not give any advice specifically regarding software related activities, like software architecture work. Lack of up-front planning is one of the main concerns about adopting agile and barriers for further adoption (VersionOne, 2009).

In the following, we will limit the discussion to Scrum. The main aim for this work is to identify the ways of working that architects have taken while using Scrum. Furthermore, we want to investigate to what extent architecture work practices are in conflict with Scrum. In addition, we wanted to find out if there are correlations between these practices and the characteristics of the companies and teams running the projects. To analyze the Scrum conformance of these practices in more detail we used a set of core Scrum patterns (Scrum Pattern Community, 2011) as a reference and analyzed the potential problems arising for each practice/pattern pair. Presenting Scrum as an extendable pattern language has been recently adopted as a preferred policy in the Scrum community (Scrum.org, 2012).

In this study, the term architecture work refers mainly to architecture design, which is the most time-consuming and critical phase related to software architectures. Some practitioners consider architecture documentation as part of architecture design, and therefore documentation activities may be counted as architecture work in this study as well. However, architecture evaluation activities are explicitly excluded.

This work is based on an analysis of the results of a survey concerning the use of Scrum and the role of software architecture related work in Scrum, carried out in Finnish software industry in fall 2010. The target companies (11) ranged from small to large global companies, developing both embedded systems and more traditional business applications. The survey was carried out by conducting on-site interviews with project managers, product managers and software architects.

The chapter is structured according to the research setup given above. We will first briefly discuss existing work on the role of software architecture in agile development. We continue with an account of the survey and its results concerning the Scrum maturity in the target companies. The results related to the architecture work practices