Chapter 2
Is Lean Agile and Agile Lean?
A Comparison between Two Software Development Paradigms

Kai Petersen
Blekinge Institute of Technology, Sweden & Ericsson AB, Sweden

ABSTRACT

Lean and agile development are two development paradigms that were proposed to help dealing with highly dynamic markets and the resulting rapid changes in customer needs. As both paradigms address a similar problem, it is interesting to compare them and by that, determine what both paradigms can learn from each other. This chapter compares the paradigms with regard to goals, principles, practices, and processes. The outcome of the comparison is: (1) both paradigms share the same goals; (2) the paradigms define similar principles, with one principle ("see the whole") being unique to lean; (3) both paradigms have unique as well as shared principles; (4) lean does not define processes, while agile has proposed different ones such eXtreme programming and SCRUM.

INTRODUCTION

The nature of software development has changed in recent years. Today, software is included in a vast amount of products, such as cars, mobile phones, entertainment and so forth. The markets for these products are characterized as highly dynamic and with frequent changes in the needs of the customers. As a consequence, companies have to respond rapidly to changes in needs requiring them to be very flexible.

Due to this development, agile methods have emerged. In essence agile methods are light-weight in nature, work with short feedback and development cycles, and involve the customer tightly in the software development process. The main principles that guided the development of different agile practices such as eXtreme programming (Beck 2000) and SCRUM (Schwaber 2004) are summarized in the agile manifesto (AgileManifesto). As shown in a systematic review by (Dybå and Dingsøyr 2008) agile has received much attention from the research community.

DOI: 10.4018/978-1-60960-215-4.ch002

While agile became more and more popular lean software development has emerged with the publication of the book (Poppendieck and Poppendieck 2003), which proposes ways of how practices from lean manufacturing could be applied in the software engineering context. Lean has a very strong focus on removing waste from the development process, i.e. everything that does not contribute to the customer value. Furthermore, according to lean the development process should only be looked at from an end-to-end perspective to avoid sub-optimization. The aim is to have similar success with lean in software development as was the case in manufacturing. That is, delivering what the customer really needs in a very short time.

Both development paradigms (agile and lean) seem similar in their goal of focusing on the customers and responding to their needs in a rapid manner. Though, it is not well understood what distinguishes both paradigms from each other. In order to make the best use of both paradigms it is important to understand differences and similarities for two main reasons:

- Research results from principles, practices, and processes shared by both paradigms are beneficial to understand the usefulness of both paradigms. This aids in generalizing and aggregating research results to determine the benefits and limitations of lean as well as agile at the same time.
- The understanding of the differences shows opportunities of how both paradigms can complement each other. For instance, if one principle of lean is not applied in agile it might be a valuable addition.

The comparison is based on the general descriptions of the paradigms. In particular, this chapter makes the following contributions:

- Aggregation of lean and agile principles and an explicit mapping of principles to practices.
- A comparison showing the overlap and differences between principles regarding different aspects of the paradigms.
- A linkage of the practices to the principles of each paradigm, as well as an investigation whether the practices are considered part of either lean or agile, or both of the paradigms.

The remainder of the chapter is structured as follows: Section 2 presents background on lean and agile software development. Section 3 compares the paradigms with respect to goals, principles, practices, and processes. Section 4 discusses the findings focusing on the implications on industry and academia. Section 5 concludes the chapter.

BACKGROUND

Plan-driven software development is focused on heavy documentation and the sequential execution of software development activities. The best known plan-driven development model is the waterfall model introduced by Royce in the 1970s (Royce 1970). His intention was to provide some structure for software development activities. As markets became more dynamic companies needed to be able to react to changes quickly. However, the waterfall model was built upon the assumption that requirements are relatively stable. For example, the long lead-times in waterfall projects lead to a high amount of requirements being discarded as the requirements became obsolete due to changes in the needs of the customers. Another problem is the reduction of test coverage due to big-bang integration and late testing. Testing often has to be compromised as delays in earlier phases (e.g. implementation and design) lead to less time for testing in the end of the project.