A statistical analysis of the effects of Scrum and Kanban on software development projects

Howard Lei *, Farnaz Ganjeizadeh, Pradeep Kumar Jayachandran, Pinar Ozcan

Article history:
Received 10 April 2015
Received in revised form 16 September 2015
Accepted 9 December 2015

Keywords:
Project management factors for software development
Agile movement
Scrum methodology
Kanban methodology

1. Introduction

While project management methodologies have been used for a very long time and date back to the Egyptian era, organizations adopted the methods only half a century ago. During the mid-1900s, the defense, navy, and space research industries were the first to adopt effective project management methodologies to achieve organizational goals. By the early 1990s, with the boom in hardware and software engineering industries, project management methodologies found many takers and have proved effective in helping organizations achieve tremendous results in its products. Adopting one of the project management methodologies made organizations more efficient in terms of planning, setting timelines and budgets, and improving quality of the products that were produced.

By the late 1950s, there have been many trial-and-error methods in managing software development projects. The early methods were used to find better ways of gathering and defining project requirements, analyzing problems, and conducting systematic implementations of problems. Some of the methods were incremental and iterative in nature [1] and others were linear and sequential, known as “Waterfall Model” [2]. The Waterfall model assumes that the team has nearly perfect information about the project requirements, the solutions, and ultimately the goal. Hence, changes in requirements were not encouraged, and became an expensive affair. Nevertheless, the sequence of steps in the Waterfall model is yet another system process model that starts with a user need and ends with a completed system [3]. In this model, testing and verification are performed at each stage of the system development, starting with the low-level components and ending with the higher-level components until the entire system has been verified. In the mid-1990s, other software development methods evolved due to problems of these so-called “heavyweight software methodologies,” which are complex and require detailed documentation and expensive design [5].

In 2001, the Agile movement was introduced in response to the failures of the Waterfall software development methodology [6]. One of the models based on the Agile movement, known as Scrum, is based on principles of lean manufacturing [6]. A different methodology based on the Agile movement is called Kanban,
which was inspired by the Toyota Production System [7] and by Lean manufacturing [8]. Recently, the Kanban and Scrum methodologies are two powerful methods adopted by organizations for software development. Although there has been a debate for years about which of these methods are preferred, there has not been sufficient evidence based on statistical analysis for selecting a preferred method. The work of Sjoberg et al. in 2012 attempted to quantify the effects of Kanban versus Scrum in a case study with one company, but the authors suggested that their study should be replicated in other environments [9].

The focus of this research is to provide statistical evidence to determine if there is a significant difference between the Scrum and Kanban methods in terms of their effects on different project management factors. The project management factors examined in this work are based on the 6-point star model associated with the Project Management Body of Knowledge [10]. The factors include project scope, budget, schedule, risk, quality, and resources.

In this work, the data is first collected via a web-based survey. Each survey question relates to one of the six project management factors listed above. Next, the correlation between these factors and the success of a project is verified by a Pearson correlation analysis. Finally, Kanban and Scrum methodologies are compared to each other in terms of their effect on the six project management factors. Core results of this work have been previously published and presented at the FAIM 2014 conference in San Antonio, Texas [11]. However, this paper describes the history and backgrounds of the methodologies in greater detail, presents results in greater detail, and provides additional analysis of the results.

This paper is structured as follows: Section 2 presents the history of the Agile movement, and provides background on Scrum and Kanban theory. Section 3 describes the approach and techniques used in this work, and Section 4 provides a statistical analysis and discussion of the results. Section 5 gives a conclusion, and suggests future work.

2. Agile software development methodologies

In this section, the evolution of Agile software development, the creation of Scrum and Kanban methodologies, and their applications, are explained. Figures are also included to illustrate their usage for project development.

2.1. History of Agile software development

Prior to the emerging of “Agile software development,” there had been many trial-and-error-based methods in software development by the late 1950s. These methods were employed to find better ways of defining the project requirements, analyzing the problem, and implementing it in a systematic manner. Some of these were incremental and iterative in nature [1] and others were linear and sequential, known as the “Waterfall Model” [2]. These approaches generally lacked the effectiveness in addressing the needs of customers, managing rapidly changing project scope, delivery time and cost [4]. In the mid-1990s, other software development methods evolved due to the problems of these “heavyweight software methodologies,” which are complex methods with expensive design [5].

As a response to the heavyweight software methodologies, lightweight methodologies, including iterative and incremental methods, had been developed and implemented. Examples of lightweight (Agile) methodologies are Scrum, Extreme Programming, Dynamic Systems Development Method (DSDM), Feature-Driven Development (FDD) and Adaptive Software Development (ASD) [12]. In 2001, these lightweight methodologies had been discussed and the Manifesto for Agile Software Development had been published to define the framework and goals of these methods [13].

2.2. Early history of Scrum

Scrum is a project management methodology for Agile software development that uses iteration and incrementation. It has been designed to manage rapidly-changing project requirements by improving communication between project developers, project owners, and other team members. In 1986, Hirotaka Takeuchi and Ikujiro Nonaka named Scrum the new product development standard in auto and consumer product companies [14]. Scrum has been defined, formalized, and published as the first Agile methodology for software development [15]. In 1993, Jeff Sutherland, John Scumiaetales, and Jeff McKenna, at Easel Company, had used Scrum for software development projects for the first time [16]. In 2002, Schwaber and Beedle wrote the book “Agile with Scrum” to describe Scrum methodology [17]. Although Scrum had become a common methodology since then, a study of Agile software development shows that only 3% of the existing scientific evidence on Agile software development focuses on Scrum [18], which this paper aims to address. The following sections describe Scrum methodology, including its implementation process and practices.

2.3. Scrum theory

Scrum, based on empirical process control theory, is an iterative and incremental project management methodology to control risk and optimize the predictability of a project. Transparency, inspection, and adaptation, which are defined below, are three important factors in the Scrum process [19].

Transparency: The process must be visible to everyone who is involved in the project.

Inspection: Scrum users must inspect Scrum artifacts frequently to detect problems in early stages.

Adaptation: If an inspector determines that some aspects of the project are unacceptable and outside of the project scope, the process can be adjusted to avoid further problems.

It should be noted that it is crucial to apply these factors during different project development phases. Details pertaining to these factors are presented in the following sections.

2.4. Content of Scrum

Scrum consists of Scrum teams, events, artifacts and rules. The rules are essential to bind teams, events and artifacts together during the project. They also provide an agreeable structure for resolving conflicts within a project. The following sections explain Scrum team, events and artifacts in detail.

2.4.1. Scrum team

The Scrum team consists of a Product Owner, a Scrum Master, and Development Team Members. Teams are self-organized and cross-functional. Therefore, they have control of the project and know how to accomplish the goals without relying on directions from people outside the team [19]. The team delivers products iteratively and incrementally, maximizing the feedback they receive. Below are descriptions of the different Scrum team roles:

The Product owner is responsible for managing the Product Backlog, the list of requirements of the product, and maximizing the value of the project. His roles also include explaining the Product Backlog items and goals of the project to the Development Team, ensuring that the team understands these goals and performs at a high level.

The Scrum Master manages the Product Backlog and instructs
دریافت فوری
متن کامل مقاله

امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
پذیرش سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات