# Chapter 7
# A Historical and Bibliometric Analysis of the Development of Agile

**Cherie C. Trumbach**
*University of New Orleans, USA*

**Kenneth R. Walsh**
*University of New Orleans, USA*

**Sathiadev Mahesh**
*University of New Orleans, USA*

## ABSTRACT

*This chapter starts with a brief history of software development from a summary of traditional approaches and presents the conditions that led to agile approaches such as product complexity, shortened life cycle of the market and eventually to the widespread acceptance of Scrum. The authors then compare the narrative to the bibliometric analysis of abstract records that can be found in the Web of Science database. They parse the terms from the abstract records to identify research trends over time and map the underlying structure of agile research. Finally, they consider the future of Agile-Scrum in light of the current pandemic.*

## INTRODUCTION

"Project management" can be defined as a series of activities and processes performed as part of a project by a defined set of people, from same or different areas with the aim of generating new or improved organizational products, services, and/or processes (Jrad and Sundaram, 2015). In this chapter, we discuss agile project management. This is a method of project management focused on collaboration and frequent communication. Project management techniques should help the chances of project success. In the early years of software development, project failure was too common, resulting in the development

DOI: 10.4018/978-1-6684-3702-5.ch007

of structured approaches that sought to ensure successes. Traditional structured approaches to software project management, sometimes referred to as the systems development life cycle or the waterfall method, became accepted practice. As these structured methods became standard practice, software project failure remained high and experts recognized that such a dynamic environment may require rethinking software development management. Agile methods then took the assumption that all requirements cannot be known at project initiation and that the dynamic business and technology environments will cause change during the project.

Agile software development was formalized as a methodology in 2001 with the creation of the Agile Manifesto. It refers to a set of computer programming methodologies that emphasize flexibility, collaboration, efficiency, simplicity, and most of all, delivering working products to end users within short timeframes (Codington-Lacerte, 2018). However, as time has passed, agile principles have spread into additional industries including general project management. According to Hayat (2019) almost every software company uses agile development, particularly Scrum, and these companies have experienced many positive results from its use.

This chapter starts with a brief history of Software Development from a summary of traditional approaches and presents the arguments made for agile approaches and the eventual widespread acceptance of Scrum. We then compare the narrative to the bibliometric analysis of abstract records that can be found in the Web of Science database. Finally, we consider the future of Agile-Scrum in light of the current pandemic.

## BACKGROUND

### From Traditional Software Development to Scrum

Early software development was done by developers who had both responsibility for analysis and coding. Such a two-step process was thought to be adequate for small scale development, but led to numerous problems for large software systems (Royce, 1970). A simple solution may have been to introduce a more sophisticated methodology with increased documentation and decomposition to allow more people to work in concert. However, such an approach is risky because errors found will inevitably lead to rework of previous steps (Royce, 1970). As an answer to this, a more sophisticated development timeline including iteration, documentation, and testing was proposed (Royce, 1970). The result of this line of thinking was an iterative and adaptive model of the phases of large-scale software development (Royce, 1970). Boehm (1983) reiterated the importance of a planned and phased approach. Boehm (1983 also emphasized the importance of documenting and completing phases before moving on to subsequent phases, although he also discussed using prototyping, incremental development, and scaffolding as ways to not necessarily defer coding until specifications were complete. This process was formalized into the familiar "analysis-design-implementation-test" steps. As the process became more formalized, particularly in the 1980s, the opportunity to automate parts of the process became more evident. However, with formalization by a wide range of practitioners, the structure of the phases may have been refined and the expense of the concepts of iteration and adaptation.

In the early stages of agile software development, prototypes followed by incremental steps were utilized. These were the first steps toward agile processes. However, these increments and prototypes were limited. They may have been applied in just one cycle where a prototype was refined to produce a