Chapter 18
Using Wiki for Managing Knowledge in Agile Software Development

Pankaj Kamthan
Concordia University, Canada

ABSTRACT

The movement towards agility is one of the most significant changes in industrial software engineering over the past decade. In the practice of agile methodologies, there are different types of knowledge that is created, communicated, and consumed. For the benefit of the stakeholders involved, there is a pressing need to manage this knowledge, both during development and beyond deployment of a software system. This chapter proposes a framework comprising related conceptual models as means for understanding the use of Wiki for managing knowledge in agile software development. In doing so, Wiki is considered beyond that of a technology or a tool, as a facilitator of knowledge, and placed in a larger context of the Social Web environment. For the sake of practicality, a number of illustrative examples are given, and implications of deploying a Wiki are highlighted.

INTRODUCTION

In the past decade, there have been a number of paradigmatic changes in the ecosystem of industrial software engineering, including the movement towards agility. The agile methodologies (Highsmith, 2009; IIBA, 2013) are part of a shift from predictive to adaptive approach towards software development. It has been shown in a number of empirical studies that agile methodologies are being increasingly deployed in many organizations, of different sizes, for software projects with teams that are geographically collocated or dispersed (Šmite, Moe, & Ågerfalk, 2010; Brown, 2012).

The involvement of people in the process is among the hallmarks of agile software development (Martin, 2003). In pursuing an agile project, certain knowledge is created, communicated, and consumed by people in that project. This knowledge plays a significant role in different ways. The ability of an organization to manage such knowledge effectively is among the critical success factors of the project (Perkins, 2006; Schneider, 2009). For the organization respon-

DOI: 10.4018/978-1-4666-8637-3.ch018

sible for the project, such knowledge forms its *competitive advantage* (North & Kumta, 2014) and contributes to its *organizational memory* that needs to be recorded and preserved for subsequent action, be it social, technical, legal, or otherwise (Girard, 2009). There could be irreparable loss of valuable knowledge if knowledge is left unattended, if organizational personnel leave (Rus & Lindvall, 2002), or if there are long gaps in the use of knowledge (Boughzala & Dudezert, 2012). It is therefore crucial to explore and examine means that can accommodate the technical as well as non-technical dimensions of managing knowledge in agile software development.

There are a number of candidate means in the constantly-evolving *technological ecosystem* of an agile project. In the past decade, one technological means, namely that of the *Social Web* (O'Reilly, 2007), has gained acceptance and prominence. The noteworthy applications within the Social Web environment include *Wiki* (Leuf & Cunningham, 2001). The purpose of this chapter is to examine the potential of Wiki and to explore the prospective uses of Wiki for managing the knowledge created, communicated, and consumed in agile software development, from the perspectives of software engineering as well as that of the Social Web.

The rest of the chapter is organized as follows. First, background and previous work on agile methodologies and Wiki is presented. This is followed by introduction of elements of a conceptual framework for systematically integrating Wiki for managing knowledge in agile software development, and the limitations of a commitment to Wiki. Next, directions for future research are outlined. Finally, concluding remarks are given.

BACKGROUND

This section provides relevant background, and previous work relating agile software development, knowledge management, and Wiki.

An Overview of Agility from a Software Project Management Perspective

In the 1990s, a number of limitations of rigidity in approaches for the development of certain types of software systems in certain industrial settings were realized. The drive to cope with these limitations led to the inception of agility.

The *Agile Manifesto* (http://agilemanifesto.org/) characterizes the term 'agile' and provides a vision for agile software development. It is motivated by a number of reasons, including the need for organizations to adequately respond to variability in the software ecosystem, to enable releases of software approximations so that the suitability of a release could be checked early and regularly, and, if necessary, the production velocity (speed and direction) could be adjusted accordingly, and to improve the relationship between technical and non-technical stakeholders, thereby reducing risk and increasing the likelihood of success of software projects.

An *agile methodology* is a software development methodology based on the Agile Manifesto. The other terms can be derived similarly.

An agile methodology is usually equipped with a process that outlines how the development of software should proceed over time, and what it should produce during that time. An *agile process* is typically nonlinear (specifically, it is both iterative and incremental).

The Agile Manifesto constitutes the basis for a number of agile methodologies, including *Crystal Clear*, *Dynamic Systems Development Method* (DSDM), *Extreme Programming* (XP), *Feature-Driven Development* (FDD), *Open Unified Process* (OpenUP), and *Scrum*. The surveys conducted regularly by many organizations, including *IBM* and *VersionOne*, over the past few years have consistently shown that XP and Scrum are among the agile methodologies currently in common use.