Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research

John Erickson, University of Nebraska at Omaha, USA
Kalle Lyytinen, Case Western Reserve University, USA
Keng Siau, University of Nebraska - Lincoln, USA

ABSTRACT

While there are many claims for the successful use of extreme programming (XP) and agile modeling (AM), and the proponents can often be vocal in the extreme regarding their supposed benefits, research evidence supporting proponents’ claims is somewhat lacking. Currently, the only research appearing to investigate the phenomena consists of two prominent streams. A small number of case studies and experience reports that generally promote the success of XP in various development environments, and a well-established stream of research into pair programming has generated results that in part support the idea of XP. Research into AM appears to be even more sparse than that for XP. Case studies, comparative analyses, and experience reports comprise the majority of the research in the area, while very few empirical research efforts have been conducted. This article reviews the state of research in XP and AM, and recommends areas that could benefit from further study. Since nearly all empirical XP research relates to pair programming, a closer look into the unstudied XP core practices would be beneficial, although interaction between related core practice areas could confound such efforts. It might also be possible to group related core XP concepts and study the groups individually. Finally, there are those who claim that XP and AM, or even agility in general, are really nothing more than a repackaging of old concepts. This claim needs to be investigated.

Keywords: agile software development; agility; agile modeling; AM; extreme programming; XP

INTRODUCTION

In a world where system-development methodologies abound, and an entire area of research and practice, as evidenced in method engineering (Siau, 1999), has grown up focusing on the creation of software-development methodologies, it can often appear that there might be a different software-development methodology for every system. This means that the choice of system-development approach can be a daunting and an extremely difficult task.

Given the unacceptably high failure rates associated with systems-development efforts...
(Hirsch, 2002; Siau, Wand, & Benbasat, 1997), and the fact that many traditional development methodologies are extremely complex and difficult to use, the choice of methodology becomes even more critical. Recently, the notion of theoretical and practical complexity was introduced (Erickson & Siau, 2004; Siau, Erickson, & Lee, 2002, 2005). In such a turbulent environment, where it seems obvious that one size does not fit all (Henderson-Sellers & Serour, in press; Merisalo-Rantanen, Tuunanen, & Rossi, in press), the agile software-development approaches would appear to be just what the doctor ordered. Extreme programming (XP) and agile modeling (AM), under the umbrella of the agile approaches to systems development, are two relatively recent emergent forces of the genre.

While there are many claims for the successful use of extreme programming and/or agile modeling (C3 Team, 1998; Grenning, 2001; Manhart & Schneider, 2004; Poole & Huisman, 2001; Schuh, 2001; Strigel, 2001), and the proponents can often be vocal in the extreme regarding the supposed benefits of both (Ambler, 2001b, 2001c, 2002a, 2002b; Beck, 1999), research evidence supporting the claimed benefits is somewhat lacking. At this point, the only exceptions seem to be research into the phenomena consisting of two prominent streams. First, researchers have conducted a scant few studies of extreme programming, consisting primarily of case studies and experience reports. While not detracting from the value of a well-conducted case study, additional research into the details of the purported benefits of the approaches would lend some much-needed weight to the existing body of work. The body of research into agile modeling appears to be even sparser than that for extreme programming. Case studies, comparative analyses, and experience reports comprise the majority of the scant research in the area, while very few empirical research efforts have been conducted. Other research efforts encompass the agile software-development approach as a whole.

This article reviews the state of research in extreme programming and agile modeling. In addition, research into agile software development will be examined. These goals will be accomplished by first briefly presenting the details of agility, XP, and AM. A literature review of the approaches follows. The article then identifies gaps in the literature, and proposes possible areas where future study would benefit both research and practice. Finally, we conclude the article.

AGILITY, XP, AND AM

Agility

Agility is often associated with such related concepts as nimbleness, suppleness, quickness, dexterity, liveliness, or alertness. At its core, agility means to strip away as much of the heaviness, commonly associated with traditional software-development methodologies, as possible to promote quick response to changing environments, changes in user requirements, accelerated project deadlines, and the like. The reasoning is that the traditional established methodologies are too set and often too full of inertia so that they cannot respond quickly enough to a changing environment to be viable in all cases, as they are often marketed to be.

The “Agile Manifesto” was composed by several XP leaders, promoters, and early adopters (e.g., Kent Beck, Martin Fowler, Robert Martin, Steve Mellor, etc.), and outlines the principles embodied in software and system agility (Lindstrom & Jeffries, 2004). Agile methodologies attempt to capture and use the dynamics of change inherent in software development in the development process itself rather than resisting the ever-present and quickly changing environment (Fowler & Highsmith, 2001). Among the agile approaches are XP, Crystal methodologies, SCRUM, adaptive software development, feature-driven development (FDD), dynamic systems development, and AM.