Abstract

Failure rates for systems development projects are estimated to approach 50% (Hirsch, 2002). In such an environment, a growing number of developers propose the use of so-called agile methodologies as one means of improving the systems developed while simultaneously decreasing failure rates. Agile proponents insist that adherence to The Agile Manifesto will improve the entire systems development process. This chapter begins by describing some of the agile methodologies, follows that with an overview of current research in the area, and closes with thoughts on possibilities for future applied research into the agile methodologies that could provide evidence supporting or disputing the many claims for success emerging from the field.
Introduction

What are the determinants of success, or conversely, failure, regarding information systems deployments? It seems that IT developers and implementing companies have found as many ways to fail as to succeed. The failure rate of systems development projects is estimated to be more than 50% (Hirsch, 2002). Add to that the fact that many traditional development methodologies are extremely complex and difficult to use, the choice of development and implementation methodology can assume critical proportions. Businesses have come to accept the environment as unarguably turbulent, and the (systems) development environment as a subset appears equally unsettled. In such an arena, it might seem an afterthought that one size does not fit all when it comes to choosing a specific development methodology (Henderson-Sellers & Serour, 2004; Merisalo-Rantanen, Tuunanen, & Rossi, 2004). Enter the agile software development approach as a potential solution. Agile systems development has become the flavor du jour of a group of software developers. Extreme programming and agile modeling are two relatively recent and highly publicized (some would say hyped) specific types of agile development approaches.

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 extremely lacking, although recent work has begun to address at least some of the problems (Fruhling & De Vreede, 2006; Holmström, Fitzgerald, Ågerfalk, & Conchúir, 2006). Currently, the only exceptions seem to be research into two areas.

One, although researchers have begun to study extreme programming, most of the research comprises case studies and field or action research conducted by the principal researcher(s) and related as a case or field report. While this exposition does not intend to detract from the value of a well-conducted case study, additional research into the specific details of the purported benefits of extreme programming would lend some much-needed weight to what appears to be a rather anecdotal body of work. Two, a well-established stream of research into pair programming has generated a set of mixed results that in part provide support for at least one core practice of extreme programming.
A Graphical Interface to Genome Multidatabases
[www.igi-global.com/article/graphical-interface-genome-multidatabases/51190?camid=4v1a](www.igi-global.com/article/graphical-interface-genome-multidatabases/51190?camid=4v1a)

Data Quality Assessment
[www.igi-global.com/chapter/data-quality-assessment/20722?camid=4v1a](www.igi-global.com/chapter/data-quality-assessment/20722?camid=4v1a)