Chapter 89
Agile Model–Driven Engineering

Kevin Lano
http://orcid.org/0000-0002-9706-1410
King's College London, UK

Hessa Alfraihi
http://orcid.org/0000-0001-8169-3766
Princess Nourah Bint Abdulrahman University, Saudi Arabia

ABSTRACT

In this article, the authors describe the field of agile model-driven software engineering methods and present the results of a survey of the research and industrial practice in this field over the last 25 years. They evaluate how the field has grown and diversified over the surveyed time period, describe the range of applications of agile model-driven development, and identify the key milestones and points of progress of the field. The evolution of the area in terms of the techniques employed and the contribution of industrial practitioners is also described.

1. INTRODUCTION

Agile model-driven engineering (Agile MDE) refers to the integration of agile methods of software development with the model-driven engineering (MDE) paradigm.

Agile methods and model-driven engineering both originated in the late 1990's/early 2000's, and both offered new but apparently opposed solutions for the high failure rate of software projects using conventional software development processes. Agile methods emphasise manual code production and flexibility to changing requirements, whilst MDE has emphasised the construction of software models, and the use of tool support for automated code generation from models.

1.1 Model-Driven Engineering

MDE was based on work carried out in the 1990's by the Object Management Group (OMG) and several major companies to produce an integrated and comprehensive object-oriented modelling language. The resulting "Unified Modelling Language" (UML) was formed as a combination of three leading object-oriented modelling approaches: OMT, the Booch method, and Jacobson's Objectory. The UML

DOI: 10.4018/978-1-6684-7366-5.ch089

has been managed by the OMG since 1997 and is currently at Version 2.5.1 (www.omg.org/spec/UML/2.5.1/PDF). UML provides a wide range of modelling notations, including dynamic behaviour models such as state machines and interaction diagrams, but the most widely-used notations are class diagram and use case models.

MDE places primary emphasis on the construction of software models (such as UML models) during software development, with the aim that executable code and documentation can be generated from these models automatically. The mappings from models to code and between models are defined as *model transformations*, usually written in specialised languages such as QVT (OMG, 2016), also standardised by the OMG.

Initial MDE process concepts, such as the Model-driven Architecture (MDA) (OMG, 2023) and Rational Unified Process (Kroll & Kruchten, 2003), appeared to be heavyweight and oriented to traditional staged development approaches. Hence, MDE was considered to be more appropriate for systems with relatively low requirements change, and in practice it has been most widely utilised in high-integrity domains such as vehicle control systems (AUTOSAR, 2023) and aerospace (NASA, 2023). For such domains it provides the benefits of a rigorous and systematic approach to software construction, and enables systems to be defined with specified architectural properties and organisation (Lano & Rahimi, 2014).

However from 2000 onwards, requirements change and rapid system evolution were increasingly becoming the norm in software projects in general. Thus traditional staged development processes, with complete up-front requirements analysis of an entire system, were becoming less relevant. As a consequence, research activities and practical experiments in using an agile approach with MDE were initiated.

1.2 Agile Methods

Following from the publication of the Agile Manifesto (Beck et al., 2001), the concepts of agile methods became widely disseminated and increasingly adopted in the software industry. Agile practices emphasise lightweight, incremental and iterative development, designed to deliver value to customers quickly. Instead of staged development of an entire application, an Agile development process performs a series of timeboxed iterations or *sprints*, each of which delivers a specific subset of functionalities/features of the products to customers (Figure 1).