Chapter 2
Methodologies:
Outlining the Process Framework

ABSTRACT

In this chapter, the authors discuss software development methodologies. These are adaptive process frameworks adjustable to software product size and scope. They usually include a set of methods, principles, techniques, and software development tools. Every methodology can implement any of the lifecycle models. The authors discuss the difference between formal and agile methodologies. The formal methodologies include more artifacts. For each activity, every role assigned to it produces a deliverable. Agile methodologies are applicable in uncertain conditions. These agile methodologies rely on self-disciplined and self-manageable teams. They are more constrained by human-related factors. As in lifecycle models, there is no "silver bullet" in software development methodologies. Due to rigorous processes and more artifacts, formal methodologies are suitable for large-scale product development. Agile methodologies require special techniques and a high level of discipline. Otherwise, they likely result in a low-quality software.

INTRODUCTION

The previous chapter introduced lifecycle models. In this chapter, the authors discuss a few software development methodologies. These are adaptive process frameworks adjustable to software product size and scope. They include a set of methods, best practices and tools. The above-mentioned crisis in software development triggered the advent of these methodologies.

DOI: 10.4018/978-1-5225-5589-6.ch002

Each methodology can implement any of the lifecycles introduced in the previous chapter. Some methodologies are more formal in terms of development process and product artifacts. Others are more flexible. In agility requiring conditions agile methodologies are applicable. These conditions are usually more uncertain. Agile methodologies often require fewer product artifacts. However, a disciplined team is mission-critical (Kamthan & Shahmir, 2016).

Similar to lifecycle models, there is no universal methodology. The earlier, formal methodologies are suitable for mission-critical and large-scale applications. The later, agile ones are better for uncertain conditions. However, in case of undisciplined development they often degrade into build-and-fix lifecycle and low quality software (Walkinshaw, 2017).

This chapter is organized as follows. Section called "What is a Methodology" presents an overview of process frameworks for the methodologies. Section "Responsive Methodologies" presents the key features of Rational Unified Process methodology. Section "Scalable frameworks" describes the Microsoft Solution Framework methodology. Section "Agile Approaches" provides an overview of the flexible methodologies. Section "Combining Best Practices" describes OpenUP methodology.

The conclusion summarizes the results of the chapter.

WHAT IS A METHODOLOGY

The previous chapter described the lifecycle models of software systems and the main stages of their development. It started with a conceptual idea and went through requirements, specification, design, implementation, and maintenance/support to retirement. The authors discussed how these steps are performed in certain lifecycle models. Some of the models, such as object-oriented, include all of the lifecycle steps. The others, such as build-and fix, do not. There are certain models with linear sequence of phases, such as waterfall. The others, such as object-oriented or spiral, support cyclic or iterative lifecycles. Some models, such as object-oriented, allow parallel or concurrent execution of some lifecycle stages, e.g. analysis and design. There are dependent lifecycle models such as rapid prototyping. It is often reasonable to combine rapid prototyping with other lifecycle models, such as spiral or waterfall. Such combinations save implementation time and costs. They also help to avoid severe design errors. They facilitate functionality demonstration of the product in its early stages: analysis, preliminary design and requirements specification. Moreover, rapid prototyping helps to