Chapter XVIII
A Composite Software Framework Approach for Mobile Application Development

Mohammed Maharmeh
University of Western Sydney, Australia

Bhuvan Unhelkar
MethodScience.com & University of Western Sydney, Australia

ABSTRACT

This chapter presents the use of Composite Application Software Development Process Framework (CASDPF) for Mobile Applications Development. This framework for software development, as its name suggests, is made up of the waterfall, iterative, and agile approaches to software development. There is a need to apply such a framework in developing mobile applications. The chapter explains and provides details on what comprises a CASDPF and how it can be used to develop a mobile application.

INTRODUCTION

The increasing use of mobile technology in business provides people and organization with flexibility to access information remotely at anytime and from anywhere. For example, people who were earlier reliant on a form of a desktop computer in an office or a laptop in a hotel room (Mark et al., 2001) are now able to access that information independent of even these locations. The development of mobile applications is considered challenging and complicated due to the specific demand and technical constraints of the environments. These constraints relate to the nature of mobile devices, security requirements and wireless networks. Thus, there is a need to have a proper and formal software development process framework to model and construct a mobile application. The composite process framework described in this chapter combines the business rules and processes that are involved in mobile application development. This chapter proposes the use of a Composite Process Framework that comprises elements of each of the process lifecycles concurrently from software processes such as Waterfall, Iterative-Incremental or Agile, to enable developers of mobile applications to create models as well as adopt the good points in the application software development methods. A composite software development process framework, as envisaged here, retains the flexible aspects of the agile development approach and, at the same time, facilitates exchange of information between project stakeholders (such as business users, developers and testers) during the

project life-cycle. Therefore, the CASDPF increases the chance of project success.

The aim of this chapter is to provide an insight on the background of software development processes, and the potential use of a composite application software development process framework for development of mobile applications. The chapter is organized as follows. The next section provides a background about software development processes, it followed by another section that highlights the definition of a composite software development process framework. Next it provides details of using the composite framework for mobile applications development and finally the conclusion and future direction.

SOFTWARE DEVELOPMENT PROCESSES BACKGROUND

Software development methodology or process is the application of best-practice business analysis and project management techniques to facilitate exchange of project information and knowledge between the project stakeholders (business users, developers, and testers), to shorten the development Life-cycle, and deliver the product on time and within the budget. Depending on the client's, project's or department's needs, a combination of one or more of proven methodologies and/or emerging standards (i.e. Waterfall approach (Royce, 1970), Rapid Application Development – RAD (Martin,1991), Spiral (Boehm, 1988), Rational Unified Process – RUP (Kruchten, 2000), or Agile approach - eXtreme Programming (XP) (Beck, 1999) could be used during the System Development Life-Cycle (SDLC) process.

The process itself can be made of a reusable 'process-components'; and a process-component can be defined as a collection of activities, tasks, roles, and deliverables in a process. Unhelkar (2003) has described the process components that can be used in three modelling spaces (Problem space, Solution space and Design space). According to Unhelkar (2003), the architecture of a process-component is made up of three major parts:

- **Deliverable:** What is produced at the end of a process, such as a suite of UML diagrams, programs, databases or quality checks.
- **Activity Task:** This is the step-by-step guide to how a particular process component is created.
- **Role:** That represents the person who is carrying out the activities and tasks.

A high-ceremony process provides the necessary guidelines (such as Deliverables, Activities, and Roles) within a project that will help in defining and controlling the project. This high-ceremony development process requires that every step of the development to be defined and followed in details, which make it relatively inflexible.

A low-ceremony process does not require any pre-definition of deliverables, activities and roles, which make it more flexible to adopt rapid changes, but lack for scalability and project control. The CASDPF achieves a balance between flexibility and rigour when using it for developing a mobile application.

COMPOSITE SOFTWARE DEVELOPMENT PROCESS FRAMEWORK

Overview

The Composite Software Development Process Framework is a standard procedure for adopting a combination of software development approaches in a single project within an organisation in such a way that could resolve some of the issues and problems associated with the implementation of these processes in software development projects.

Composite Process Conceptual Model

The Composite approach shown in Figure 1 below splits the project into a set of increments; each increment consists of a set of project life-cycle phases. For each increment, it attempts to bring the best aspects of available software development processes (Waterfall, Iterative and Agile) together that can be used to manage application software development. In this iterative incremental approach, development of increments goes throughout circles, a careful consideration required during the planning and execution of increments to ensure that it goes across the project life-cycle phases.