Chapter 14
Systematic Use of Software Development Patterns through a Multilevel and Multistage Classification

Sofia Azevedo
Universidade do Minho, Portugal

Ricardo J. Machado
Universidade do Minho, Portugal

Alexandre Bragança
Instituto Superior de Engenharia do Porto, Portugal

Hugo Ribeiro
Primavera Business Software Solutions, Portugal

ABSTRACT

Software patterns are reusable solutions to problems that occur often throughout the software development process. This chapter formally states which sort of software patterns shall be used in which particular moment of the software development process and in the context of which Software Engineering professionals, technologies and methodologies. The way to do that is to classify those patterns according to the proposed multilevel and multistage pattern classification based on the software development process. The classification is based on the OMG modeling infrastructure or Four-Layer Architecture and also on the RUP (Rational Unified Process). It considers that patterns can be represented at different levels of the OMG modeling infrastructure and that representing patterns as metamodels is a way of turning the decisions on their application more objective. Classifying patterns according to the proposed pattern classification allows for the preservation of the original advantages of those patterns and avoids that the patterns from a specific category are handled by the inadequate professionals, technologies and methodologies. The chapter illustrates the proposed approach with the classification of some patterns.

DOI: 10.4018/978-1-61692-874-2.ch014

INTRODUCTION

In the context of software development, patterns are provided as reusable solutions to recurrent problems. In other words, software patterns are reusable solutions to problems that occur often throughout the software development process. Pattern classifications emerged as a way to organize the many patterns that have been synthesized. *Pattern classification* is the activity of organizing patterns into groups of patterns that share a common set of characteristics. The simple fact of organizing patterns into classifications is a way of building a stronger knowledge on patterns, which allows understanding their purpose, the relations between them and the best moments for their adoption (Gamma, Helm, Johnson, & Vlissides, 1995).

Despite their use within the software development process, the use of patterns may not be systematic. In the context of this chapter, the systematic use of software development patterns means that decisions on the application of patterns are less subjective and more objective. Besides that, a lot of pattern classifications were conceived until the present day, yet none of them formally stated which sort of patterns shall be used in which particular moment of the software development process. This chapter will provide for specific directives on how to systematically adopt patterns within a multilevel and multistage software development process. A multilevel and multistage classification of patterns will be the foundation of such systematic use of patterns.

A multistage software development process can be defined as a software development process composed of some stages organized in a consecutive temporal order. Each stage is separated from the contiguous ones by well defined borders. Moreover each particular stage is composed of a flow of well defined activities. Each stage's activities are conducted by specific professionals, using specific technologies (frameworks, languages, tools), under the directives of specific methodologies (processes, notations and methods) to achieve specific goals. Borders are well defined if the shift in the professionals, technologies, methodologies and goals that takes place when moving from one stage to another is identified in terms of the development process. A multilevel software development process can be defined as a software development process concerned with the levels of abstraction in which the different artifacts involved in the development of software are handled. In the context of this chapter, those levels are the levels of the OMG (OMG, 2009a) modeling infrastructure or Four-Layer Architecture (Atkinson & Kühne, 2003), depicted in Figure 1. The OMG modeling infrastructure comprises a hierarchy of model levels just in compliance with the foundations of MDD (Model-Driven Development) (Atkinson & Kühne, 2003). Each model in the Four-Layer Architecture (except for the one at the highest level) is an instance of the one at the higher level. The first level (*user data*) refers to the data manipulated by software. Models of user data are called *user concepts models* and are one level above the user data level. Models of user concepts models are *language concepts models*. These are models of models and so are called metamodels. A metamodel is a model of a modeling language. It is also a model whose

Figure 1. The OMG modeling infrastructure or Four-Layer Architecture