Chapter XIX
Formalizing Patterns with the User Requirements Notation

Gunter Mussbacher
University of Ottowa, Canada

Daniel Amyot
University of Ottowa, Canada

Michael Weiss
Carleton University, Canada

ABSTRACT

Patterns need to be described and formalized in ways that enable the reader to determine whether the particular solution presented is useful and applicable to his or her problem in a given context. However, many pattern descriptions tend to focus on the solution to a problem, and not so much on how the various (and often conflicting) forces involved are balanced. This chapter describes the user requirements notation (URN), and demonstrates how it can be used to formalize patterns in a way that enables rigorous trade-off analysis while maintaining the genericity of the solution description. URN combines a graphical goal language, which can be used to capture forces and reason about trade-offs, and a graphical scenario language, which can be used to describe behavioral solutions in an abstract manner. Although each language can be used in isolation in pattern descriptions (and have been in the literature), the focus of this chapter is on their combined use. It includes examples of formalizing Design patterns with URN together with a process for trade-off analysis.

INTRODUCTION

Patterns document common solutions to recurring problems in a specific context. They enable an efficient transfer of experience and skills. However, many pattern descriptions tend to focus on the solution to a problem, and not so much on how the various (and often conflicting) forces involved are balanced. Therefore, patterns need to be described and formalized in ways that enable

the reader to determine whether the particular solution presented is useful and applicable to his or her problem in a given context.

A large body of patterns has been documented to date, and the different efforts are not well-connected. The Pattern Almanac (Rising, 2000) alone, a major effort summarizing and linking the patterns published at patterns conferences and in books prior to the year 2000, lists over 1,200 patterns contained in over 800 different publications. Most of those are publications with a single pattern. The number of patterns has only increased since, but estimates are harder to obtain lacking a similar effort to the Pattern Almanac.

Much work on pattern formalization focuses on the solution domain. For instance, Taibi and Ngo (2001) describe why patterns should be formalized and suggest combining formal specifications of structural and behavioral aspects of Design patterns in one specification. However, the problem domain and relevant trade-offs are seldom handled formally.

In this chapter, we propose a formalization of patterns using the user requirements notation (URN) in a way that supports a rigorous trade-off analysis. In the following sections we first present the background for patterns and the formalization of patterns. Then, we review related work on reasoning about the trade-offs between patterns. We then introduce the user requirements notation, and our explicit model of the forces addressed by a pattern, which provides the basis for the trade-off analysis. The description of the approach is followed by a case study from the literature to which we have applied our approach. Finally, we present future trends and present our concluding remarks.

FORMALIZING PATTERNS

Patterns are three-part rules that describe a recurring problem that occurs in a specific context and its solution (Alexander, 1979). They capture important practices and existing methods uncodified by conventional forms of communicating design experience. The structure they capture is usually not immediately apparent. Perhaps the most significant contribution of patterns is that they make the trade-offs between forces explicit.

Each pattern describes the situation when the pattern can be applied in its context. The context can be thought of as a precondition for the pattern. This precondition is further refined in the problem description with its elaboration of the forces that push and pull the system to which the pattern is applied in different directions. Here, the problem is a precise statement of the design issue to be solved. Forces are design trade-offs affected by the pattern. They can be documented in various forms. One popular approach is to document the trade-offs as sentences like "on one hand …, but on the other hand …".

The solution describes a way of resolving the forces. Some forces may not be resolved by a single pattern. In this case, a pattern includes references to other patterns, which help resolve forces that were unresolved by the current pattern. Together, patterns connected in this way are often referred to as a pattern language. Links between patterns can be of different types, including uses, refines, and conflicts. Patterns that need another pattern link to that pattern with *uses*. Patterns specializing the context or problem of another pattern *refine* that pattern. Patterns that offer alternative solutions *conflict*.

Current pattern representations are textual. They include the gang-of-four (GoF) form, the Coplien form, and the Alexandrian form. The GoF form (Gamma, Helm, Johnson, & Vlissides, 1994) includes sections for intent, motivation, structure, participants, and collaborations. The emphasis of this format is on the structure of the solution. However, the discussion of the forces is spread out over multiple sections, which makes it challenging for a developer to get an overview of when to apply a particular pattern and the consequences of using it.