Applying design patterns in product line search-based design: feasibility analysis and implementation aspects

Giovani Guizzo*, Thelma Elita Colanzi*† and Silvia Regina Vergilio*
*DInf - Federal University of Parana, CP: 19081, CEP 19031-970, Curitiba, Brazil
†DIN - State University of Maringa, Av. Colombo, 5790, Maringá, Brazil
Email: {gguizzo, thelmae, silvia}@inf.ufpr.br

Abstract—Some works have manually applied design patterns in Product Line Architectures (PLAs) in order to improve the understanding and reuse of the PLAs artifacts. However, there is no search-based approach that considers such subject. Applying design patterns in conventional architectures through mutation processes in evolutionary approaches has been proven as an efficient technique. In this sense, this work presents results of a feasibility analysis for the automated application of GoF design patterns with a search-based PLA design approach. In addition to this, the paper proposes a metamodel representation for suitable scopes to receive the design patterns application and a mutation operator procedure to apply the patterns identified as feasible.

Keywords—Software product line; design patterns; search-based design.

I. INTRODUCTION

Software Product Line (SPL) encompasses commonality and variability that are present in several software products of a specific domain. The Product Line Architecture (PLA) is a key asset that provides a common overall structure containing all the SPL commonalities and variabilities. The PLA allows to derive the architecture of each product [1].

The PLA design is a hard SPL engineering activity that can benefit a lot from search-based approaches. In this sense, in a previous work [2], we proposed a multi-objective search-based approach to optimize the PLA design in order to reduce SPL development efforts and costs. The approach produces a set of good solutions with the best trade-off among different objectives (metrics), such as the SPL extensibility and modularity. The focus is the UML class diagrams, which are commonly used to model software architectures in a detailed level [3].

Since “GoF” design patterns [4] can optimize some software metrics, such as cohesion and coupling, their application helps to obtain a PLA more flexible, understandable, and able to accommodate new features during the SPL maintenance or evolution. Räähä et. al. [5] used genetic algorithms to apply design patterns using mutation operators to conventional software architectures. However, their approach is not feasible to PLAs and the patterns are not necessarily applied into suitable scopes. Moreover, the adopted architecture representation is different from that one used in [3], which is also used in this paper. Furthermore, we did not find studies about the automated application of design patterns in the search-based PLA design. Considering the evidences that mutation operators to apply design patterns can contribute to obtain better PLAs [6], this paper discusses aspects to their implementation in the search-based approach, such as automatic identification of suitable application scopes and mutation operators. Such implementation aspects have as fundamental a feasibility analysis conducted with the GoF catalog. A suitable scope is a part of a class diagram that has architectural elements satisfying the minimum requirements for the application of a specific pattern.

In this way, the work has the following contributions: i) results from a feasibility analysis of the GoF design patterns application; ii) a metamodel representation and specification for suitable scopes; iii) a mutation operator to the application of feasible design patterns by evolutionary algorithms, according to the search-based PLA design approach [2].

The paper is organized as follows. Section [II] contains the feasibility analysis results. Section [III] introduces the metamodel representing suitable scopes and discusses implementation aspects. Section [IV] describes the mutation operator procedure. Section [V] concludes the paper.

II. THE FEASIBILITY ANALYSIS

Since the approach proposed in [2] is fully automated, the feasibility analysis is concerned with the possibility of automatically identifying suitable scopes for application of design patterns, preventing the introduction of design anomalies. Therefore, patterns are only applied to these suitable scopes. The analysis described in this section considers the following factors for each design pattern: i) structure; ii) consequences; iii) applicability in PLAs; iv) flexibility; and v) implementation aspects. Each factor is considered only for class diagrams, which is the one used in [2].

In short, we consider feasible the patterns whose suitable scopes can be automatically identifiable and whose consequences are profitable to PLA in terms of coupling, cohesion, reuse and understanding. Some design patterns which have suitable scopes for their application in conventional architectures, do not have specific application scopes for PLA. However, not having a particular context of a PLA to be applied does not necessarily set the pattern as infeasible, since it can still be applied to a PLA but in conventional scopes.

The main results of this analysis are: all Creational patterns are infeasible, since they are appropriate only to create objects and are applied in a low range of scopes. It is also hard to automatically identify suitable scopes for Behavioral patterns...
using only class diagrams. Five of them could be feasible if other diagrams were considered (Template Method, Chain of Responsibility, Observer, Command and Visitor). Structural patterns are the most suitable in the context of this work, since they are more compatible with the architecture representation used. Briefly, four patterns were considered feasible: Bridge, Strategy, Facade and Mediator. Suitable PLA scopes could be identified only for Bridge and Strategy.

III. IMPLEMENTATION ASPECTS

In the considered scenario implementation, an existing PLA design outset is the goal of the optimization by applying patterns. Some PLA quality requirements expected for a solution are evaluated as objectives through different measures such as that ones used in [6]. In the approach, the solution (PLA) is represented by a metamodel (as described in [3]). In such scenario, the feasible design patterns are applied by a mutation operator. To the definition of the mutation operators, there are some points that must be ensured: i) the pattern is being applied into a suitable scope of the architecture; ii) the applied pattern is coherent and does not bring any anomaly to the architecture; iii) effectively apply the design pattern as a mutation in the evolutionary process; and iv) make the process of identification of the suitable scopes and application of the design patterns totally automated.

To satisfy such requirements and to provide an easy way to deal with suitable scopes during the mutation process, this section introduces a generic metamodel (Figure 1). Such model is instantiated for each feasible pattern, and criteria for the identification of their scopes are described.

Figure 1. Metamodel representing PS and PS-PLA

If a scope is suitable for the application of a design pattern, it is called “Pattern application Scope” (PS). The notation “PS<Pattern Name>” is used to designate a scope for a specific design pattern. A scope may be a PS for more than one design pattern. In addition to the PS specification, there is a category specific for SPL scopes: “Pattern application Scope in Product Line Architecture” (PS-PLA). Each PS/PS-PLA is composed by a set of architectural elements and satisfies some requirements expected by a particular design pattern, which in turn may influence some software metrics.

IV. THE MUTATION OPERATOR PROCEDURE

Algorithm 1 presents the proposed mutation operator procedure. First, a design pattern DP is randomly selected from the set of feasible patterns (line 4). Thereafter, the mutation operator uses the selection function \( f_s(A) \) to obtain a scope \( S \) from the architecture \( A \) (line 5). A possible implementation for \( f_s(A) \) is to select a random number of randomly selected classes and interfaces from \( A \), which together with the random selection of design patterns, keeps the randomness aspect of the mutation process. However, it can be redefined by the user.

Using the verification method \( \text{verification}(\cdot) \), the scope \( S \) is verified as a PS-PLA or a PS for the application of the design pattern \( DP \) (line 6). The parameter \( \rho_{\text{pla}} \) in the method invocation is a random number to determine which verification method of the design pattern \( DP \) will be used. If the \( \rho_{\text{pla}} \) probability is not achieved, then the PS verification is used, otherwise the PS-PLA verification is used. Therefore, if the verification method returns true and the mutation probability \( \rho_{\text{mutation}} \) is achieved, then the method apply() applies \( DP \) to \( S \) (line 7). At the end, the architecture \( A \) is returned even without any modification (line 9). Each design feasible pattern has its own verification and application methods, which are also described and exemplified in this section.

V. CONCLUDING REMARKS

This work presented results from a feasibility analysis for the automated application of GoF design patterns in product line search-based design. Four patterns are feasible, and two of them have SPL specific application scopes involving variabilities. To allow the patterns application and as a result of the conducted analysis, a metamodel to represent suitable application scopes was defined. A mutation operator procedure for applying the feasible design patterns in suitable scopes was also proposed. As future work, we intend to analyze the possibility of combining design patterns to improve the quality of the generated solutions. We also intend to use UML interaction diagrams in addition to the class diagram in order to automatically apply other patterns.

REFERENCES


