The Journal of Systems and Software 115 (2016) 157–173

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Preserving architectural styles in the search based design of software product line architectures

Thainá Mariani[a,*], Thelma Elita Colanzi[b], Silvia Regina Vergilio[a]

[a] Computer Science Department, Federal University of Paraná (UFPR), CP 19:081, Curitiba, CEP: 81531-970, Brazil
[b] Informatics Department (DIN), State University of Maringá (UEM), Av. Colombo, 5790, Bloco C-56, Maringá, CEP: 87.020-900, Brazil

ARTICLE INFO

Article history:
Received 5 August 2015
Revised 14 January 2016
Accepted 25 January 2016
Available online 11 February 2016

Keywords:
Architectural style
Software product line
Search based design

ABSTRACT

Architectural styles help to improve the Product Line Architecture (PLA) design by providing a better organization of its elements, which results in some benefits, like flexibility, extensibility and maintainability. The PLA design can also be improved by using a search based optimization approach, taking into account different metrics, such as cohesion, coupling and feature modularization. However, the application of search operators changes the PLA organization, and consequently may violate the architectural styles rules, impacting negatively in the architecture understanding. To overcome such limitation, this work introduces a set of search operators to be used in the search based design with the goal of preserving the architectural styles during the optimization process. Such operators consider rules of the layered and client/server architectural styles, generally used in the search based design of conventional architectures and PLAs. The operators are implemented and evaluated in the context of MOA4PLA, a Multi-objective Optimization Approach for PLA Design. Results from an empirical evaluation show that the proposed operators contribute to obtain better solutions, preserving the adopted style and also improving some software metric values.

1. Introduction

A Software Product Line (SPL) represents a set of software products that share features in order to satisfy a specific market segment (van der Linden et al., 2007). A feature represents a functionality that is visible for the user. The SPL offers a set of common artifacts for building products, including mandatory and variable elements. A feature can be designed like a variability, which represents a variable functionality that can be or not present in a product. On the other hand, mandatory features are common for all SPL products.

The Product Line Architecture (PLA) is a fundamental artifact of the SPL engineering, and contains all commonalities and variabilities of the SPL (van der Linden et al., 2007). It is used to obtain the architecture of every derived product of the SPL. The PLA design can benefit from the use of metrics to evaluate basic design principles, such as cohesion and coupling, and other more specific SPL characteristics, such as feature modularization. Furthermore, some authors point out that the adoption of architectural styles benefits the PLA design, providing flexibility, maintainability and understandability (Gomaa, 2004; Clements and Northrop, 2001; Pohl et al., 2005). An architectural style (Garlan and Shaw, 1994) determines ways to select and show parts of an architecture, providing a specific organization of the elements, in order to improve the architecture understandability. To do this, an architectural style defines specific components, connectors and rules on how they can be combined.

The PLA design is a difficult and people-intensive task, because the architect needs to deal with conflicting factors (measures) and different ways to organize the architectural elements (styles) (Taylor et al., 2010). To help the architect in this task, Colanzi et al. (2014) proposed MOA4PLA (Multi-objective Optimization Approach for PLA Design), which aims at both optimizing some basic design principles and improving feature modularization. The approach includes a representation to the PLA design, and a model to evaluate the PLAs found by the search based algorithms, that are also guided by some specific search operators.

In order to obtain new solutions, which consist on alternative PLAs, the search operators used by MOA4PLA modify the PLA organization by adding and moving its elements. This may violate rules of the architectural style adopted in the PLA, similarly to what happens during the architecture evolution. In experiments applying MOA4PLA, Colanzi et al. (2014) obtained some solutions that

* Corresponding author. Tel.: +554197994159.
E-mail addresses: marianithaina@gmail.com, tmariani@inf.ufpr.br (T. Mariani), thelma@din.uem.br (T. Elita Colanzi), silvia@inf.ufpr.br (S. Regina Vergilio).

http://dx.doi.org/10.1016/j.jss.2016.01.039
0164-1212/

do not satisfy the rules required by the layered architectural style, leading to a confusing piece of design. It is very important to satisfy architectural style rules during the optimization of PLAs, because to adopt a style makes the system easier to maintain, and can reduce fault propagations due to changes.

There are two common styles that are usually adopted in the SPL context: layered and client/server styles (Clements and Northrop, 2001; Pohl et al., 2005; Gomaa and Hussein, 2004). The layered architectural style eases extension and contraction (Gomaa and Hussein, 2004). In the SPL context, software contraction regards to the design of mandatory elements, and software extension regards to the design of variables elements. The use of the client/server style also eases PLA evolution, since servers can be easily added or removed.

Given the importance of such styles (Garlan and Shaw, 1994; Gomaa and Hussein, 2004) and its large use in the literature (Berger et al., 2015; Abbas and Andersson, 2015; Patel and Shah, 2015; Arrieta et al., 2015; Fang et al., 2015), in a previous work, we introduced a set of operators, named SO4LAR (Search Operators for Layered ARchitectures) (Mariani et al., 2015). They were successfully applied with MOA4PLA to avoid layered rules violation. Now, in the present paper we summarize our ideas and results with layered architectures, and extend previous work by introducing a set of operators, named SO4CSAR (Search Operators for Client/Server ARchitectures). The introduced operators allow operations that move methods, attributes, and add classes and packages, usually applied with search based approaches of architectures in general, such as that ones described in Räihä (2010). In addition to this, they allow the application of feature-driven operations, which are specific to the PLA context (Colanzi et al., 2014).

Because the PLA context is more specific and operators proposed for this context can also be applied for architectures in general, the introduced operators are implemented to work with MOA4PLA. Results from empirical evaluation show that the operators produce solutions where the original style is preserved and, that are also quantitatively better, considering objectives related to basic design principles, such as cohesion and coupling, and other metrics related to feature modularization.

This paper is organized as follows. Section 2 reviews related work. Section 3 describes MOA4PLA, the search based approach for PLA design, adopted in this work. Section 4 introduces the proposed search operators for layered and client/server architectures. Section 5 presents some implementation aspects. Section 6 describes the empirical study conducted and presents the results. Finally, Section 7 contains some concluding remarks and possible future works.

2. Related work

The violation of architectural styles has been addressed in the area of software maintenance and evolution. Some works check the conformance between architectural design and implemented architecture (Postma, 2003). Other ones propose some system repairs by using refactorings (Bourquin and Keller, 2007; Terra et al., 2013). Works on the design of architectures with styles are the most related to ours, mainly considering SPL and search based approaches. They are briefly described below.

In the literature we found some works using architectural styles in the SPL context.Gomaa and Hussein (2004, 2007) introduce some software reconfiguration patterns for dynamic evolution of SPL. Such patterns are created according to the rules of some architectural styles.Ihme (2001) introduces a process to help architects in the PLA design and in the application of architectural styles. This process is specific for complex embedded systems.Fant et al. (2013) introduce an SPL engineering approach that connects architectural patterns to SPL features of a specific domain. The authors claim that applying architectural patterns based on the software domain is a good software practice to incorporate patterns in SPL and in its derived products.

Stoermer and O'Brien (2001) present an approach to manage the process of architecture mining. In this process, existing architectures are analyzed in order to extract a PLA and to apply architectural styles in it.Morisawa and Torii (2001) introduce classifications for the client/server style based on the data storage and in the connection types existing between clients and servers. Each classification is treated like a style in order to satisfy the characteristics needed for different distributed system. The authors also introduce a method of manual selection and application of these styles in SPL.Kim et al. (2008) introduces a framework named DRAMA, where the goal is to select feasible architectural styles for an SPL, based on the requirements of a previously known domain.

The presented approaches confirm that the use of an architectural style is very important in the SPL context. However, most of the mentioned approaches are not automated. They address either the selection or application of the styles, but most times in a manual way. Even though such approaches were automated, they could not be used in this work, because most of them can be applied only in specific domains. Other limitation is that they do not consider search based approaches.

Search based design approaches are found in the literature (Räihä, 2010; Aleti et al., 2013; Ramírez et al., 2015; Hasheminejad and Jalili, 2014; Martens et al., 2010; Du et al., 2015). Such approaches are a research subject of the Search Based Software Engineering (SBSE) field (Harman and Mansouri, 2010). Most of them are based on Multi-Objective Algorithms (MOAs), that is, search based algorithms that can properly deal with a problem influenced by many factors. The works address the context of general software design (Ramírez et al., 2015; Hasheminejad and Jalili, 2014; Martens et al., 2010; Du et al., 2015; Räihä, 2011; Simons et al., 2010), and also PLA design context (Colanzi et al., 2014; Guizzo et al., 2014).

In both contexts, we found works introducing search operators for the application of design patterns, such as that ones from the GOF catalogue. Räihä et al. (2011) apply the patterns *Facade, Adapter, Strategy, Template Method* and *Mediator* in the general software architecture design.Guizzo et al. (2014) introduce operators for the application of the patterns *Strategy* and *Bridge* in PLAs. However, we did not find search based approaches that take into account the architectural style. This lack of works regarding SBSE, SPL and architectural styles is one of the motivations of this paper.

We acknowledge that operators to preserve the architectural style can benefit the search based design. To perform our investigations we chose the search based PLA design, and the MOA4PLA approach, because SPL is a broader application context than traditional software. Search based design of PLAs is the subject of the next section.

3. Search based design of PLA

A search based design approach for PLAs was proposed by Colanzi et al. (2014). This approach, named MOA4PLA (Multi-objective Approach for PLA Design) and described next, is used in this work to apply and evaluate our operators.

MOA4PLA uses multi-objective optimization algorithms to improve a PLA provided as input. It includes a meta-model to represent the PLA, allowing its symbolic manipulation by the algorithm, which is guided by an evaluation model composed by metrics that are specific for PLA. The approach uses traditional search operators and an operator specific to improve feature modularization. The MOA4PLA activities are shown in Fig. 1 and described in the next paragraphs.