Chapter 11
An Aspect-Oriented Framework to Model Non-Functional Requirements in Software Product Lines of Service-Oriented Architectures

Germán Harvey Alférez Salinas
Universidad de Montemorelos, Mexico

Edward Mauricio Alférez Salinas
Universidade Nova de Lisboa, Portugal

ABSTRACT

Non-functional requirements (NFRs) are of primary importance in Software Product Lines (SPLs) of Service-Oriented Architectures (SOAs) as they specify the quality characteristics of a software system within a SPL. However, they are difficult to manage because they are found in many contexts with varying concerns and crosscut multiple concerns along the software lifecycle.

Existing variability management techniques in the context of SPL engineering tend to concentrate at the code level and do not address NFRs in SOAs. The analysis of variability from the beginning benefits the management of services from requirements to design and vice versa.

Also, there is a need for a navigation chart to help practitioners to model NFRs in SPLs of SOAs while separating difficult to modularize and maintain crosscutting concerns.

This chapter presents and applies an extended version of an aspect-oriented framework for SPLs that exploits aspect-oriented software development (AOSD) techniques in order to model variability of NFRs.

DOI: 10.4018/978-1-60566-794-2.ch011
INTRODUCTION

Software Product Line (SPL) engineering is about exploiting commonalities among a set of systems while managing their variabilities in order to shorten time to market, achieve systematic reuse goals, and improve product quality (Clements & Northrop, 2002).

There is a close relationship between SPLs and Service-Oriented Architectures (SOAs) because both of them have similar goals: to implement, integrate and maintain software with reused artifacts in order to reduce costs (Bichler & Lin, 2006; Papazoglou, 2003).

Aspect-Oriented Software Development (AOSD) is a paradigm that has a direct relationship with SPL and SOA because one of its main objectives is to separate concerns to promote flexibility and configurability; these two objectives are also vital when constructing SPLs and SOAs. In addition, AOSD can improve the way in which software is modularized by means of the encapsulation of variabilities in functional requirements (FRs) and non-functional requirements (NFRs) into aspects that crosscut various services.

Our previous research (Alférez & Poonphon, 2007) presents a framework that uses AOSD to manage variability from the early stages of the SPL lifecycle in order to improve the traceability of variations throughout the development of SPLs of Web applications. The current research extends the aspect-oriented framework focusing on managing variability of NFRs at initial stages (requirements, architecture, and design) in the lifecycle of SPLs of SOAs. The success of a SPL approach depends on early variability management, not only at the implementation level (Voelter & Groher, 2007).

Also, this chapter briefly describes variability implementation with AspectJ.

Existing requirements engineering techniques such as goals, scenario, and use case modeling support the analysis of requirements but they do not focus on crosscutting concerns. As a result, research on aspect-oriented requirements engineering complements these approaches by providing systematic means for handling such crosscutting concerns (Chitchyan et al., 2005).

In addition, practitioners need an easy-to-follow framework to model NFRs with aspects in SPLs of SOAs to manage difficult to modularize and maintain crosscutting concerns.

The framework is closely related to the Core Asset Development and Product Development activities in product line development proposed by the SEI (Clements & Northrop, 2002). Also, it is designed as a process description and recommendation to use Unified Modeling Language (UML) models with extension mechanisms.

The analysis of the proposed framework is driven by a SPL of a SOA for help desks where metrics were applied in order to assess the performance of the framework.

The remainder of this chapter is structured as follows. Section 2 presents the background where three software engineering techniques are covered and related: SPL, AOSD, and SOA. Section 3 presents the aspect-oriented framework to model NFRs in SPLs of SOAs. Section 4 shows the application of the proposed framework in a SPL of help desks for plant services departments. Section 5 outlines the future trends and finally, section 6 provides some conclusions of this chapter.