Discovering Volatile and Aspectual Requirements
Using a Crosscutting Pattern

José María Conejero, Juan Hernández, Ana Moreira and João Araújo
1Quercus Software Engineering Group, University of Extremadura, Cáceres, Spain
{chemacm, juanher}@unex.es
2CITI/Dept. Informática, FCT, Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
{ja, amm}@di.fct.unl.pt

Abstract

A modern software development approach needs to cope with the constant requirements changes observed in current business markets. Such volatile requirements are usually tangled with other requirements making systems evolution difficult. In this poster, we propose a conceptual framework to identify volatile requirements as well as requirements that classic approaches are unable to modularize. This framework is based on what we called the crosscutting pattern and uses traceability matrices and matrix operations.

1. Introduction

In [4], the authors present an approach to cope with volatile concerns by externalizing them. They propose the use of aspect-oriented techniques to isolate such concerns in the same way as crosscutting concerns. Crosscutting is usually described in terms of scattering and tangling. Scattering is the occurrence of elements that belong to one concern in modules encapsulating other concerns. Tangling is the occurrence of multiple concerns mixed together in one module. Crosscutting is the scattering and tangling of concerns arising due to poor support for their modularization [2]. However, the distinction between these concepts is vague, sometimes leading to ambiguous statements and, therefore, to confusion. A formal definition of crosscutting allows both crosscutting concerns and crosscutting requirements to be precisely identified. In [3] the authors proposed a conceptual framework where crosscutting is defined by means of a crosscutting pattern.

In this poster we propose the use of the framework presented in [3] to identify volatile and crosscutting concerns at the requirements level. This identification allows an aspectual requirements refactoring using the model notation introduced in [4]. The poster shows the applicability of the framework to identify crosscutting concerns and to cope with other situations where maintainability and evolution must be improved.

2. Identifying crosscutting and volatile concerns

The process introduced in [4] consists of three main steps. Firstly, concerns are identified by inspecting documents that describe the problem, existing catalogs [1], stakeholders’ interviews transcripts and searching techniques. Next, concerns are classified in three different categories: service concerns represent functionality that the system must provide and are not amenable to frequent changes; crosscutting concerns are concerns that are scattered and tangled with other concerns; and volatile concerns are considered concerns which are likely to change during the system development or after the system has been put into operation. Finally, requirements containing crosscutting or volatile concerns are refactored and modeled by means of the Use Case Pattern Specification presented in [4]. This poster contributes to the second phase of the process by offering a mechanism to systematically classify the concerns in a system. This classification is performed by applying the framework presented in [3]. This framework identifies crosscutting concerns and tangled concerns that are not scattered. Our view is that tangled concerns are good volatile concerns’ candidates. The framework uses a crosscutting pattern (see Figure 1) that represents relations between elements of two different domains (i.e., concerns and requirements related to these concerns).

* This work has been partially supported by MEC under contract TIN2005-09405-C02-02 and also by Portuguese FCT Grant POSC/EIA/ 60189/2004
These relationships are represented and visualized by means of a special kind of traceability matrix, the dependency matrix (step 1 in Figure 2). A dependency matrix \((\text{source} \times \text{target})\) represents the dependency relation between source elements and target elements. In this case, this dependency matrix shows the relationships between concerns and requirements. From this traceability matrix, we derive two different matrices: the scattering and the tangling matrix (they can be seen in step 2 in Figure 2). While the former represents the concerns scattered in the system, the later allows the representations of requirements where several concerns are being addressed. We use this matrix to obtain volatile concerns.

Since crosscutting concerns have been previously identified, we disregard them focusing only on requirements where one or more concerns are tangled (note that these are not crosscutting concerns). Then, we obtain a table with several candidate volatile concerns. Based on this table, we decide whether a candidate is a volatile concern or not. Figure 2 outlines the process.

At the end of the process concerns are classified into the three aforementioned groups, namely crosscutting, volatile and service (base) concerns. Crosscutting and volatile concerns are refactored and externalized through the framework presented in [4].

3. Conclusions.

The framework proposed identifies volatile and crosscutting concerns at the requirements level. The identification of such concerns is based on a crosscutting pattern and simple matrix operations. In [4] the identification of such concerns is mainly based on the requirements engineer’s experience or intuition. In that sense, the crosscutting pattern presented adds new benefits to the approach introduced in [4] providing a systematic way to identify and obtain the volatile and crosscutting concerns. Once the volatile and crosscutting concerns have been identified by means of simple matrix operations, the requirements can be refactored so that each concern is modularized.

References


