Chapter VII

From Bad Smells to Refactoring: Metrics Smoothing the Way

Yania Crespo, Universidad de Valladolid, Spain
Carlos López, Universidad de Burgos, Spain
María Esperanza Manso Martínez, Universidad de Valladolid, Spain
Raúl Marticorena, Universidad de Burgos, Spain

Abstract

This chapter presents a study on the relation of refactoring, bad smells, and metrics. The notions of refactoring and bad smells are revised as well as metrics that can be used as guides in the refactoring process. Connection among those metrics, the usual flaws that could be suggested by them, and the required corrective actions to reduce or erase these flaws are analyzed. The usual flaws can be described in terms of bad smells and the corrective actions, in terms of the refactoring operations suggested by each bad smell. Then, we can go from metrics to bad smells and from this, to refactoring. The chapter also describes solutions for tool support in a language independent manner. In this sense, it describes the tool architecture which can be defined as metamodel-centered. A metamodel representing a family of languages is defined as well as framework based solutions for collecting metrics, as well as for a refactoring engine and repository. These solutions allow reusing the effort on a wide family of object-oriented languages.

Copyright © 2007, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.
developed frameworks were instantiated to work on instances of our own metamodel. In addition to this, it also describes how to use the approach and its support, with other metamodels. Finally, a case study on the use of metrics in bad smells detection is presented.

**Introduction**

One of the key subjects in code refactoring process is: *when* and *where* do we perform refactorings? In Fowler (2000) he proposes a list of clues or symptoms that suggest refactorings. These symptoms or stinks are named “bad smells” and their detection must be achieved from “the programmer intuition and experience.”

Currently, there are a big number of integrated development environments (Eclipse, NetBeans, Visual Studio .NET, Refactoring Browser, etc.) which include refactoring. These environments contain or allow adding plug-ins for obtaining metrics. The programmer is also able to customize the warning messages and corrections for every metric over the threshold.

However, there are common points between these concepts not connected until now. Although we have metrics, they are not used to determine refactorings. There is not a direct connection among these metrics, the usual flaws that could be suggested by them, and the required corrective actions to reduce or erase these flaws. The usual flaws can be described in terms of bad smells and the corrective actions, in terms of the refactoring operations suggested by each bad smell. Then, we can go from metrics to bad smells and from this to refactoring.

On the other hand, metrics should be implemented for each object-oriented environment/language that we use. Nevertheless, one of the intrinsic properties of most of them, especially in object-oriented metrics, is their language independence.

Therefore, starting from the current state of the question, we can go forward in two directions:

- Use the metrics as bad smells clues, to hint or suggest the suitable refactorings.
- Define a language independent metric collection support. The main issue when defining this support must be to fit solution for reuse in most of integrated development environments or in a multi-language environment.

This complements a language independent approach to software refactoring.