Towards a Framework of Software Design Defects Correction with Refactoring Plans

Javier Pérez

May 12, 2008

Abstract

Software evolution is a fundamental part of the software development process, which usually results in an increase of software entropy and, as a consequence, in the decay of software structure. The most desirable approach would be to prevent this, but a systematic technique to detect and correct software design defects once they have appeared is still obviously needed.

Much work is being done to develop techniques for detection of software design defects, using metrics, structure analysis, etc., which have been proved to be very useful in revealing defects and in suggesting changes to reduce or to remove them. A convenient way to present these suggestions is through proposals of transformations which preserve the observable behaviour of the system, well known as refactorings. The problem is that it is rare that the preconditions of the desired transformations could be fulfilled by the system's source code at its current state, this means that additional transformations are needed in order to "prepare" the system for the desired evolution. Therefore, suggesting refactorings is not enough to allow systematic correction of design defects.

We define refactoring plans as the specification of a refactoring sequence that matches a system redesign proposal and that can be executed over the current system's source code. Formal theories, such as graph transformation, can be used to analyse a set of available refactorings, within a context defined by the current system's source code and a redesign proposal, in order to obtain the refactoring plan.