Software Development AntiPatterns

Good software structure is essential for system extension and maintenance. Software development is a chaotic activity, therefore the implemented structure of systems tends to stray from the planned structure as determined by architecture, analysis, and design.

Software refactoring is an effective approach for improving software structure.

The resulting structure does not have to resemble the original planned structure.

The structure changes because programmers learn constraints and approaches that alter the context of the coded solutions. When used properly, refactoring is a natural activity in the programming process.

For example, the solution for the Spaghetti Code AntiPattern defines a software development process that incorporates refactoring. Refactoring is strongly recommended prior to performance optimization. Optimizations often involve compromises to program structure. Ideally, optimizations affect only small portions of a program. Prior refactoring helps partition optimized code from the majority of the software.

Development AntiPatterns utilize various formal and informal refactoring approaches. The following summaries provide an overview of the Development AntiPatterns found in this chapter and focus on the development AntiPattern problem. Included are descriptions of both development and mini-AntiPatterns. The refactored solutions appear in the appropriate AntiPattern templates that follow the summaries.