ABSTRACT

Code smells are indicators of a problem that lies within a source code. The existence of *code smell* does not necessarily mean a problem currently exists, but it implies that a problem could exist. Two of the most common *code smells* are *Long Methods* and *God Class*. These fall into the bloater category of *code smells*, which means they are attempting to do too much and violate the Single Responsibility Principle. Developers often write such codes without noticing, and thus tools to detect such flaws are required, and it is also required that these tools be accurate. JDeodorant provides a detection of these *code smells* for Eclipse IDE and it uses a detection strategy called refactoring opportunity. This method has been an outlier in previous researches that attempt to analyze the accuracy of these types of tools. A reference list containing all the confirmed *code smells* is made by analyzing the source code using software metrics, a method of detection different to JDeodorant. It has been found that JDeodorant's detection rate possesses an accuracy rate of 0.98 for *Long Methods* detection and 0.83 for the detection of *God Classes* with respect to the reference list.

Key words: Code smell, Long Method, God Class, Eclipse IDE, detection rate