Metrics: A Deeper Understanding

Isaac Griffith

April 19, 2010

Lack of Cohesion in Methods

$$LCOM = \Pi_{f \in FieldNodes} \frac{|f.edges - 1| - |f.edges.node \in [MethodNodes \notin f.class.nodes]|}{|f.edges - 1|}$$

In other words LCOM = the product of the division between the number of edges - 1 for a given field node - each edge that contains a node which is a method node within the fields class, all divided by the number of edges connected to this field (within this class) minus 1. We want to maximize this metric.

Coupling Between Objects

$$CBO(C) = \sum_{C' \in ClassNodes} Number(C'_{methods} UsingC_{Methods}) + Number(C'_{methods} UsingC_{Fields})$$

In other words this is the summation of the number of methods for each class excluding the current class C which use the methods of C and the fields of C. There are two ways to compute this. We can either (a) Count only a single connection regardless of how many times a connection between classes, that is if a given class X is couple to class Y via two fields and three methods, we would not count this as 5 separate coupling events but instead as a single one. This then allows us to move on after a single coupling event is found. (b) We can count each coupling event separately and add sum the totals together. The first method allows us to know the overall sum (1 for each class) and we can divide by the Number(Classes) - 1 to find the coupling factor. On the other hand we can not do this if we count every instance of coupling events. We want to minimize this metric.

Weighted Methods per Class

$WMC = \sum_{m \in Methods} Complexity(m)$

Summation of the complexity for all methods of a given class. This number is then associated with the class. We want to minimize this metric.

Cyclomatic Complexity

$CC(G) = Number(edges) - Number(nodes) + 1$

This instance of Cyclomatic Complexity calculates for a given Graph G. In our case, G is a subgraph of the internal statements of a given MethodNode. Thus, in order to calculate this metric and the above WMC metric we must implement Control Flow Graphs for the internal Methods. That is given a method, we should be able to find the statement nodes attached to it and determine the Control flow graph by following a certain type of edge. Thus differentiating it from the associated use edges that connect the CFG to the rest of the graph.