This report is referring to the Chidamber and Kemere metrics.

These are the parameters that where evaluated:

**- Coupling between objects (CBO)**

Two classes are coupled when methods declared in one class use methods or instance variables defined by the other class. And the reciprocal can also happen.

It is desirable to have a low value of CBO. The multiple classes of the project have an average value of 10.6 which is acceptable.

Nevertheless, there are 161 class out of 786 that have a CBO over 14, which according to Houari A. Sahraoui, Robert Godin, Thierry Miceli study, "Can Metrics Help Bridging the Gap Between the Improvement of OO Design Quality and Its Automation?", is the max acceptable value.

This can cause a code smell, like shotgun surgery.

**- Depth of inheritance tree (DIT)**

The deeper a class is in the hierarchy, the more methods and variables it is likely to inherit, making it more complex. The deeper the tree, the more likely is to exist some problem.

According to Visual Studio .NET documentation, the recommended value is lower than 5.

In the project, the average value is 1.84 and there are only 21 out of 758 with a value greater than 5.

**- Lack of Cohesion of Methods (LCOM)**

This metric can be used to identify classes that are attempting to achieve many different objectives, and consequently are likely to behave in less predictable ways than classes that have lower LCOM values. Such classes could be more error prone and more difficult to test and could possibly be disaggregated into two or more classes that are more well defined in their behavior.

The ideal is to have a value lower than 1. In average, the project classes have a value of 2.55, with a mode of been 1, with 62 classes having a value lower than 1.

**- Number of children (NOC)**

NOC equals the number of immediate child classes derived from a base class.

There is no magic number to this parameter, just that a high NOC has been found to indicate fewer faults.

In the project, the NOC is in average 0.52, with a maximum value of 86 and a mode of 0.

**- Response for class (RFC)**

The response set of a class is a set of methods that can potentially be executed in response to a message received by an object of that class. RFC is simply the number of methods in the set.

A large RFC has been found to indicate more faults. Classes with a high RFC are more complex and harder to understand.

In the project, the average value of RFC is 20.7, with the most frequent value been 1 in 7.6% of the classes.

**- Weighted method complexity (WMC)**

This metric evaluates the number of methods in each class.

It is recommended not to have a large number of classes because it is found to lead to more faults.

In the project, there is an average of 13.74 methods in each class with a mode of 4.