CodeMR Software Metrics Tool

Alexandru Stoica

March 2019

1 Introduction

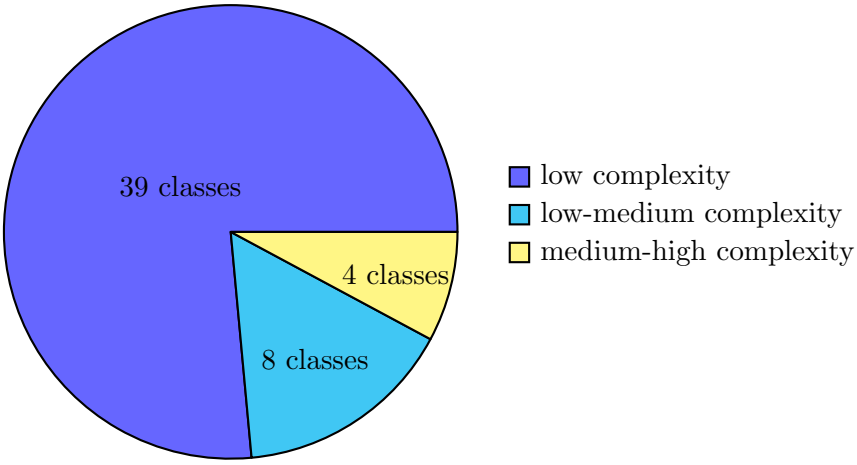
`CodeMR` is a software quality and static code analysis tool that helps software companies developing better code, better quality. `CodeMR` supports multiple programming languages such as Java, Kotlin Scala and C++.

2 Results

Data	Value
lines of code	880
number of classes	55
number of packages	9
number of external packages	50
number of external classes	120
number of problematic classes	2
number of highly problematic classes	0

2.1 Complexity

Definition 1 *Complexity implies being difficult to understand and describes the interactions between a number of entities. Higher levels of complexity in software increase the risk of unintentionally interfering with interactions and so increases the chance of introducing defects when making changes.*



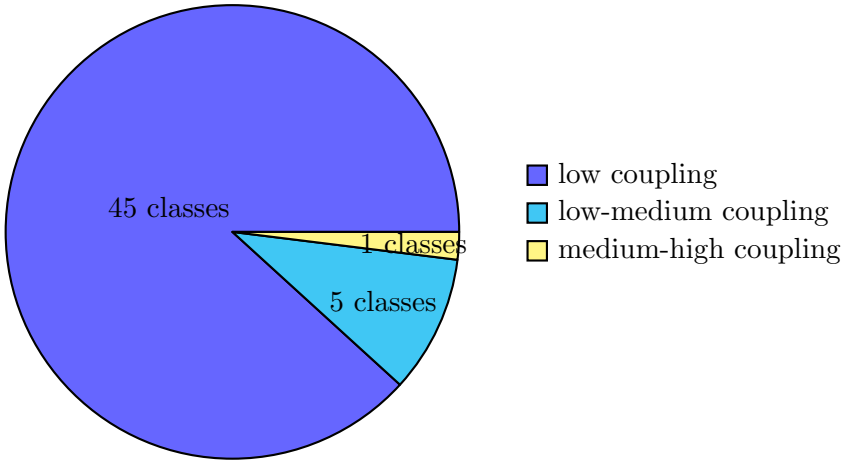
Problem Files:

`/security/JsonWebTokenAuthenticationFilter.kt`
`/security/JsonWebTokenAuthorizationFilter.kt`
`/report/ReportNotFound.kt`
`/report/UserNotFound.kt`

2.2 Coupling

Coupling between two classes A and B if:

- A has an attribute that refers to (is of type) B.
- A calls on services of an object B.
- A has a method that references B (via return type or parameter).
- A has a local variable which type is class B.
- A is a subclass of (or implements) class B.



Problem Files:
`/notification/NotificationAspect.kt`

Tightly coupled systems tend to exhibit the following characteristics:

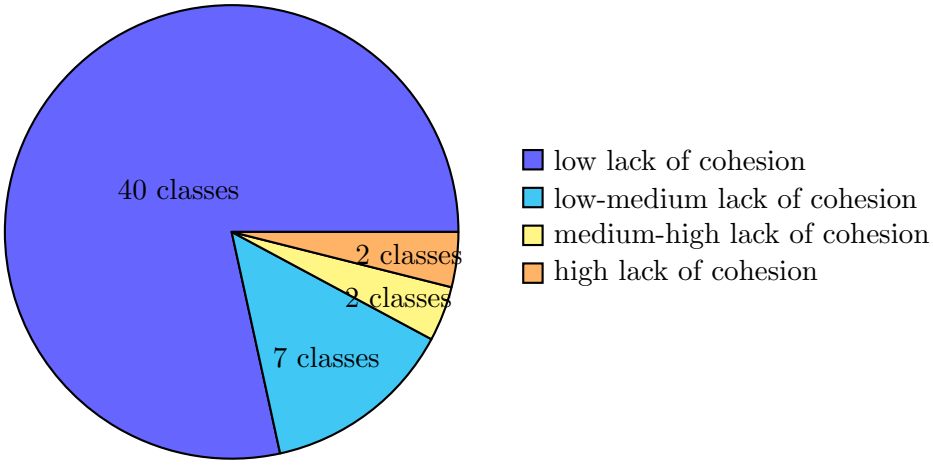
- A change in a class usually forces a ripple effect of changes in other classes.
- Require more effort and/or time due to the increased dependency.
- Might be harder to reuse a class because dependent classes must be included.

2.3 Lack of Cohesion

Definition 2 ***Lack of cohesion*** *measures how well the methods of a class are related to each other.*

High cohesion (low lack of cohesion) tend to be preferable, because high cohesion is associated with several desirable traits of software including robustness, reliability, reusability, and understandability.

In contrast, low cohesion is associated with undesirable traits such as being difficult to maintain, test, reuse, or even understand.

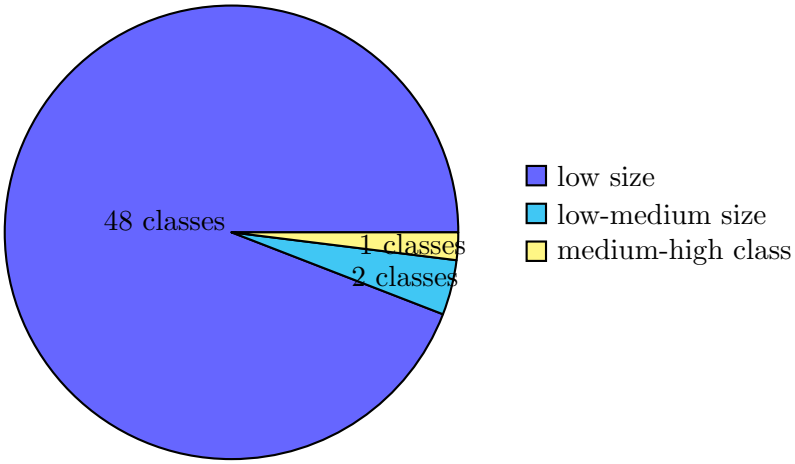


Problem Files:
`/user/User.kt`
`/report/Report.kt`

2.4 Size

Definition 3 ***Size*** *is one of the oldest and most common forms of software measurement. Measured by the number of lines or methods in the code.*

A very high count might indicate that a class or method is trying to do too much work and should be split up. It might also indicate that the class might be hard to maintain.



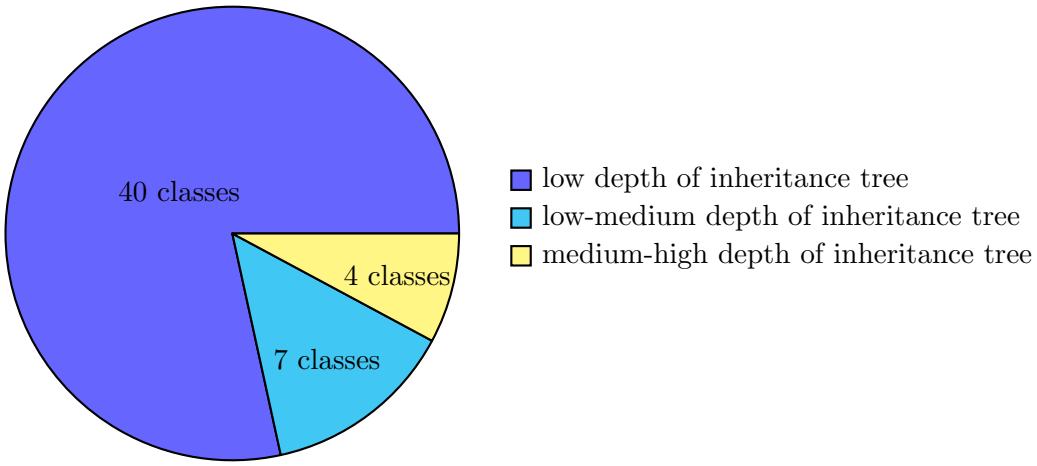
Problem Files:
`/user/User.kt`

2.5 Depth of Inheritance Tree

Related Quality Attributes: Complexity

Definition 4 *Depth of inheritance tree represents the position of the class in the inheritance tree. Has 0 (zero) value for root and non-inherited classes.For the multiple inheritance, the metric shows the maximum length.*

Deeper class in the inheritance tree, probably inherit. Therefore, it is harder to predict its behaviour. Also this class relatively complex to develop, test and maintain.



Problem Files:

`/security/JsonWebTokenAuthenticationFilter.kt`
`/security/JsonWebTokenAuthorizationFilter.kt`
`/report/ReportNotFound.kt`
`/user/UserNotFound.kt`

3 Advantages

Advantage 1 *CodeMR provides intuitive diagrams which show the connections between packages. This feature helps us identify hard-coded dependances (relationships based on concrete classes with no abstraction between the endpoints) between different systems within our application.*

Advantage 2 *Support for Java, Kotlin and Scala all in one package; in today's applications, we require tools able to support various related programming languages since most of our codebases use at least two of them.*

Advantage 3 *CodeMR presents useful diagrams of the codebase's structure, based on projects, packages and classes; this feature helps us examine our code from several perspectives and identify predicaments with ease.*

4 Disadvantages

Disadvantage 1 *The lack of customisation may be CodeMR's weakest spot. Although our engine supports the most commonly calculated metrics such as complexity, lack of cohesion and others, developers are unable to create plugins or extensions to measure additional software quality metrics.*