CHIDAMBER-KEMERER metrics

By

Gustavo Monteiro da Silva

59472

Software Engineering

07/11/2023

# Chidamber-kemerer

## Chidamber-Kemerer metrics, often referred to as CK metrics, are a set of software complexity metrics designed to assess various aspects of object-oriented software code. These metrics focus on the whole class itself instead of parts of it.

# CHIDAMBER-KEMERER metrics

Uma imagem com texto, Tipo de letra, captura de ecrã, número

Descrição gerada automaticamente

1. **CBO – Coupling Between Objects**

CBO counts how many other classes a given class is directly linked to. A class with a high CBO value depends on many other classes, which means it has a higher degree of coupling.

* 1. Data

Interfaces are not considered.

* 1. Analysis

The program indicates a high level of coupling, with the average being 21. That means the code might be difficult to understand, teste, and modify. The most extreme case for this metric are:

Uma imagem com texto, Software de multimédia, software, captura de ecrã

Descrição gerada automaticamente

One being an abstract class that represents the FreeCol root class and the other being the class that represents the Player itself. Values are expected to be high on these 2 classes since both must interact with many aspects of the program.

1. **DIT – Depth of Inheritance Tree**

DIT determines the depth of the inheritance hierarchy for a class. It computes how many ancestor classes a class has. A class with a high DIT value has a deeper class hierarchy, which can increase the code complexity.

1. Data

Interfaces and enumerators are not considered.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

1. Analysis

The average DIT for this project is 3.5. Which could be considered a good value given the project scope.

The 3 classes with the highest DIT are:

Uma imagem com texto, captura de ecrã, Tipo de letra, file

Descrição gerada automaticamente

All 3 classes are related a GUI that displays reports to the player.

1. **LCOM – Lack of Cohesion in Methods**

LCOM evaluates the cohesion within a class. It identifies how many groups of methods do not share attributes. A class with a higher LCOM value has lower cohesion, which can make the code more difficult to maintain.

* 1. Data

Interfaces are not considered.

* 1. Analysis

The metric indicates that the program has a good level of cohesion (average of 2.65), that is an excellent characteristic given the open-source nature of the code. That means that the code is easier to maintain and reuse.

The extreme case of lack of cohesion is in the following class:



This class is the API and common reusable functionalities of the overall GUI. For that reason it is expected that the class would not have a high cohesion.

1. **NOC – Number of Children**

NOC measures the number of immediate subclasses a class has. A class with a high NOC value has many child classes, which can affect its maintainability and complexity.

* 1. Data

Interfaces and enumerators are not considered.

* 1. Analysis

Most of the Classes does not have children, that means that the classes are likely to be less complex and serve a specific role or functionality within the software.

The most extreme case is found in the following class:



This class implements the basic functionalities of messages. Since the game has a lot of messages throughout the gameplay, its expected that this class would have a higher number of immediate subclasses.

1. **RFC – Response For a Class**

RFC estimates the total number of methods that can be invoked in response to a message to an object of a class. A class with a high RFC value responds to many different messages and can be harder to understand and maintain.

* 1. Data
  2. Analysis

The average value for this metric is 40 RFC. The highest value is found in the following class:



Being the game controller class, its expected that the value of RFC would be extremely high since it must respond to every single command in game.

1. **WMC – Weighted Methods per Class**

WMC calculates the complexity of a class by counting the number of methods it contains. Methods with more complexity receive higher weight. A class with a high WMC value has complex functionality, which can make it more challenging to maintain.

* 1. Data Interfaces are not considered.
  2. Analysis

The average value of this metric is 24.5, which is not a high value. The class with the highest value of this metric is once again the Game Controller Class, also something expected because of the high number of controller inputs that the class must manage.

