Dependency Metrics

Rafael Tavares 60608

Introduction

As required, we will be analyzing the code metrics for the FreeCol codebase, in this report, more specifically, the Dependency Metrics, extracted using the MetricsReloaded plugin for IntelliJ IDEA.

Metrics Data

The extensive dataset of metrics collected can be found in the spreadsheet sent alongside the report, however, to make this metrics overview simpler, we will only take a look at the average values for each field.

Metrics Fields

To better comprehend what these metrics mean, and why are they valuable, we will first understand the purpose of each field:

1. **Number of Cyclic Dependencies:** Measures how many times classes or packages participate in cycles with other classes or packages.
2. **Number of Dependencies:** Counts the total number of direct relationships where a class or package relies on other classes or packages to function.
3. **Number of Transitive Dependencies:** Tallies the dependencies that are not direct but are through a chain of other classes or packages. For instance, if A depends on B, and B depends on C, then A has a transitive dependency on C.
4. **Number of Dependents:** Indicates the count of other classes or packages that directly rely on a given class or package.
5. **Number of Transitive Dependents:** Similar to transitive dependencies, this counts how many classes or packages indirectly rely on a given class or package through a chain of dependencies.
6. **Number of Package Dependencies:** The specific count of direct dependencies at the package level, reflecting how many other packages a single package depends on.
7. **Number of Dependent Packages:** This metric shows how many other packages directly depend on a given package, indicating its level of responsibility and importance within the application's package structure.

Class Values

Average values for each field, in regards to Classes:

Number of Cyclic Dependencies	639.131
Number of Dependencies	10.778
Number of Transitive Dependencies	804.308
Number of Dependents	11.574
Number of Transitive Dependents	890.477
Number of Package Dependencies	3.725
Number of Dependent Packages	3.007

From these average values, it’s evident that the codebase exhibits a high degree of interconnectivity, particularly in terms of cyclic dependencies among classes. With an average of 639.131 cyclic dependencies, there's a strong indication that the codebase could suffer from tight coupling and poor separation of concerns, making it difficult to maintain and evolve.

A significant number of transitive dependencies (804.308 on average) also suggests that classes are not only depending on many other classes directly but also through chains of dependencies, this could lead to challenges in understanding the impact of changes, potential for ripple effects across the codebase, and difficulties in ensuring the isolation of components for testing and maintenance.

The number of dependents and transitive dependents (11.574 and 890.477, respectively) are indicative of certain classes being heavily relied upon, either directly or indirectly, which may point towards classes that are too central or hold too much responsibility. Such classes could become problematic “Large Class", with an excessive number of responsibilities, making them critical points of failure.

On the packaging side, while the direct package dependencies and dependents (3.725 and 3.007, respectively) don't seem to be as extreme, they still suggest a moderate level of coupling between packages. This could potentially lead to package coupling, where changes in one package may have a cascading effect on others, indicating that refactoring towards more modular and loosely coupled packages might be beneficial.

Overall, the extreme values in cyclic and transitive dependencies strongly suggest the presence of code smells related to class design and package structure. Refactoring to address these issues could greatly improve the maintainability and robustness of the codebase.

Interface Metrics

Average values for each field, in regards to Interfaces:

Number of Cyclic Dependencies	431.415
Number of Dependencies	1.756
Number of Transitive Dependencies	603.780
Number of Dependents	12.439
Number of Transitive Dependents	832.171
Number of Package Dependencies	0.878
Number of Dependent Packages	3.683

For interfaces, the average number of cyclic dependencies is significant at 431.415, though not as high as with classes, suggesting that interfaces also contribute to the cyclic complexity of the codebase, which might imply an overuse or misuse of interfaces in the system.

The average number of dependencies for interfaces is relatively low at 1.756, indicating that interfaces tend to have fewer direct relationships compared to classes. However, just like in classes, the number of transitive dependencies remains high (603.780), which points towards a complex network of indirect interface relationships that could lead to hidden dependencies and make changes more difficult to manage.

The number of dependents and transitive dependents is higher for interfaces (12.439 and 832.171, respectively) compared to classes, this reflects interfaces' purpose in the system but also flags a potential over-reliance on interfaces, which could suggest some interfaces are not cohesively defined or are carrying too much responsibility.

Unlike in classes, the number of package dependencies and dependent packages for interfaces are considerably lower (0.878 and 3.683, respectively), implying that interfaces are less tied to the structure of packages and may be more reusable across the codebase. This is a positive sign in terms of package design, but given the other metrics, it still requires careful management to prevent scenarios where interfaces contain methods that not all implementers actually need, violating the interface segregation principle.

Package Metrics

Average values for each field, in regards to Packages:

Number of Cyclic Package Dependencies	40.133
Number of Package Dependencies	9.556
Number of Dependent Packages	9.444
Number of Transitively Dependent Packages	43.933

For packages, the metrics suggest a relatively contained level of cyclic package dependencies, with an average of 40.133, this is a concern as it indicates some level of tight coupling at the package level, potentially complicating the modularity of the system.

The average number of direct package dependencies and dependent packages is roughly equal (9.556 and 9.444, respectively), which suggests a balanced but possibly intricate web of inter-package relationships, this could be a sign that packages are not as independent as they could be, potentially leading to difficulties in isolating changes and impacts across the codebase.

The number of transitively dependent packages stands at 43.933 on average, which indicates that the dependency chain can extend significantly beyond immediate connections. While this may point to a robust system of interactions, it also raises the risk of unintended side effects when changes are made in one package.

Overall, the metrics related to the packages suggest a complexity in the package structure with a noticeable number of cyclic dependencies and inter-package relationships, this reflects a scenario where packages may be too closely tied to each other, leading to challenges in maintaining and evolving the codebase due to the potential for cascading changes and the difficulty in isolating package responsibilities.