**Dependency Metrics**

Dependency metrics are used to measure and analyze the relationships between classes and packages in a software system (interfaces are also measured). These metrics help software developers and architects to understand the complexity and maintainability of their codebase

The following values are shown **at the class level**:

Number of Cyclic Dependencies

* This metric indicates the number of cyclic dependencies within a class. Cyclic dependencies occur when two or more classes depend on each other in a circular way, creating a cycle.

Number of Dependencies

* This metric measures the total number of classes or components that depend on a specific class. It reflects how interconnected a class is with other parts of the codebase.

Number of Transitive Dependencies

* Transitive dependencies are dependencies that are inherited through other classes. This metric counts the total number of classes, both direct and indirect, that depend on a specific class.

Number of Dependents

* This metric represents the total number of classes that a specific class depends on. It shows how many other classes are needed for the proper functioning of the analyzed class.

Number of Transitive Dependents

* Similar to the number of transitive dependencies, this metric counts the total number of classes, both direct and indirect, that a specific class depends on.

Number of Package Dependencies

* This metric measures the number of other packages or modules that a class depends on.

Number of Dependent Packages:

* This metric represents the total number of packages or modules that depend on a specific class. It indicates how many different packages rely on the functionality provided by the analyzed class.

**At the Interface level**, we have the same 7 metrics as the class level, maintaining the same definition but related to interfaces and their relationships.

**At the package level,** we have 4 package related metrics, (“Number of Cyclic Package Dependencies”, “Number of Package Dependencies”, “Number of Dependent Packages”, “Number of Transitively Dependent Packages”), respectively equivalent to the metrics “Number of Cyclic Dependencies”, “Number of Dependencies”, “Number of Dependents”, “Number of Transitive Dependents”, in class and interface level’s, but related to packages.

Potencial Trouble Spots

In the FreeCol code these metrics in some cases show high values in classes, interfaces and packages. Those suggest that the code might be complex, hard to maintain and in some cases over-used, being that it should be reviewed and adjusted according to the class’s responsibilities, method complexity, efficient and appropriate use of dependencies etc.

An example is the class TileType (scr.net.sf.freecol.common.model.TileType), with 922 transitive dependents, which means that it is a potentially complex and heavily used class.

Another example is the BaseProduction interface (scr.net.sf.freecol.common.model.BaseProduction), with 804 cyclic dependencies, it means that many classes depend on this interface as well as the interface depends on those classes, which leads to maintainability issues and makes the codebase difficult to work with.

A solution for this case might be refactoring the interface to reduce its dependencies, and if possible, even split the interface into smaller interfaces, each serving a specific purpose.

**Relation to Code Smells**

André Branco nº62482