**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.

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.

**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.

**Lack of Cohesion**  
Measure 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.

**Size**

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.

**LOC (Class Lines of Code)**  
Related Quality Attributes: Size  
The number of all nonempty, non-commented lines of the body of the class. CLOC is a measure of size and also indirectly related to the class complexity.