Metrics definitions

- **ACAIC** Ancestor Class-Attribute Import Coupling
- **ACMIC** Ancestors Class-Method Import Coupling
- **AID** Average Inheritance Depth of an entity
- **ANA** Count the average number of classes from which a class inherits informations
- **CAM** Computes the relatedness among methods of the class based upon the parameter list of the methods
- **CBOin** Coupling Between Objects of one entity
- **CBOout** Coupling Between Objects of one entity
- **CIS** Counts the number of public methods in a class
- **CLD** Class to Leaf Depth of an entity.
- **cohesionAttributes**
- **connectivity**
- **CP** The number of packages that depend on the package containing entity
- **DAM** Returns the ratio of the number of private(protected) Attributes to the total number of Attributes declared in a class
- **DCAEC** Returns the DCAEC (Descendants Class-Attribute Export Coupling) of one entity
- **DCC** Returns the number of classes a class is directly related to(by attribute declarations and message passing
- **DCMEC** Returns the DCMEC (Descendants Class-Method Export Coupling) of one entity
- **DIT** Returns the DIT (Depth of inheritance tree) of an entity.
- **DSC** Count of the total number of classes in the design
- **EIC** the number of inheritance relationships in which superclasses are in external packages
- **EIP** the number of inheritance relationships where the superclass is in the package containing entity and the subclass is in another package
- **ICHClass** Compute the complexity of an entity as the sum of the complexities of its declared and inherited methods
- **LCOM1** Returns the LCOM (Lack of COhesion in Methods) of an entity
- **LCOM2** Returns the LCOM (Lack of COhesion in Methods) of an entity
- MFA The ratio of the number of methods inherited by a class to the number of methods accessible by member methods of the class
- **MOA** Count the number of data declarations whose types are user defined classes
- **NAD** number of attributes declared
- **NADExtended** Number of attributes declared in a class and in its member classes
- **NCM** Returns the NCM (Number of Changed Methods) of an entity.
- **NCP** the number of classes package containing entity
- **NMA** Returns the NMA (Number of New Methods) of an entity
- **NMD** number of methods declared
- **NMDExtended** Number of methods declared in the class and in its member classes
- **NMI** Returns the NMI (Number of Methods Inherited) of an entity
- **NMO** Returns the NMO (Number of Methods Overridden) of an entity
- **NOA** Returns the NOA (Number Of Ancestors) of an entity
- **NOC** Returns the NOC (Number Of Children) of an entity
- **NOD** Returns the NOD (Number Of Descendents) of an entity
- **NOH** Count The number of class hierarchies in the design
- **NOM** Counts all methods defined in a class

- **NOP** Returns the NOP (Number Of Parents) of an entity
- **NOParam** Compute the average number of parameters of methods
- **NOPM** Count of the Methods that can exhibit polymorphic behavior
- **PIIR** The number of inheritance relationships existing between classes in the package containing entity
- **PP** The number of provider packages of the package containing entity
- **REIP EIP** divided by the sum of PIIR and EIP
- **RFP** the number of class references from classes belonging to other packages to classes belonging to the package containing entity
- **RPII PIIR** divided by the sum of PIIR and EIP.
- **RRFP RFP** divided by the sum of RFP and the number of internal class references
- **RRTP RTP** divided by the sum of RTP and the number of internal class references
- **RTP** The number of class references from classes in the package containing entity to classes in other packages.
- **SIX** Returns the SIX (Specialisation IndeX) of an entity
- **WMC1** Computes the weight of an entity considering the complexity of a method to be unity.
- **McCabe Complexity**: Number of points of decision + 1
- **CBO** Coupling Between Objects of one entity
- **LCOM5** Returns the LCOM (Lack of COhesion in Methods) of an entity
- **WMC** Computes the weight of an entity by computing the number of method invocations in each method