



# SQMetrics v 0.7

## User's Manual

DIMITRIOS M. SOFRONAS  
HELLENIC OPEN UNIVERSITY

1/1/2020

# SQMetrics v 0.7

## User's Manual

### INTRODUCTION

- This is a simple application to measure the most common software quality metrics. It has a simple graphical interface, so the user should not have much experience to use it.

### USAGE

To run the program, you simply have to run the "SQMetrics\_x.x.jar" file (where x.x. is the current version). If, for any reason, the program wont start, run the "RunMe\_on\_Windows.vbs" file (if you are under Microsoft Windows) or "Run\_Me\_On\_Linux.bat" if you are under Linux. No failures have been noticed under Mas OS X. After that, you face the "Welcome Screen" (Picture 1).

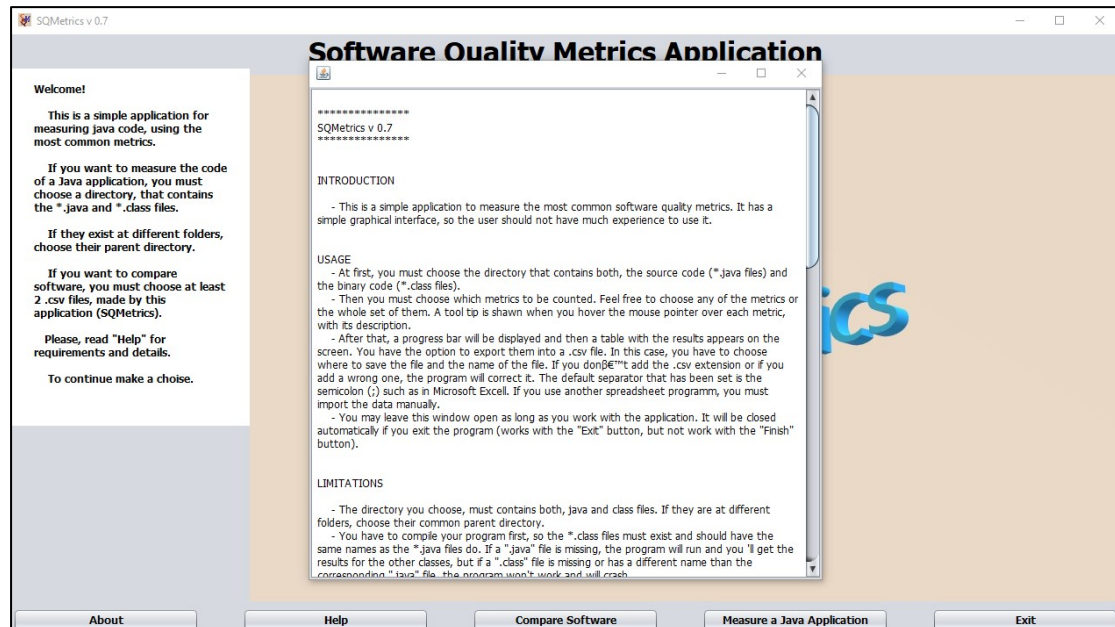


Picture 1: Welcome Screen

A short description exists, such as 5 buttons too.

The “About” button displays information for the application.

The “Help” button displays a relative help text in a new window (Picture 2). This window remains open until you close it, or you advance into another screen. This behavior is the same, for all the “Help” windows on all screens. The help text that appears is relative to the screen you called it.



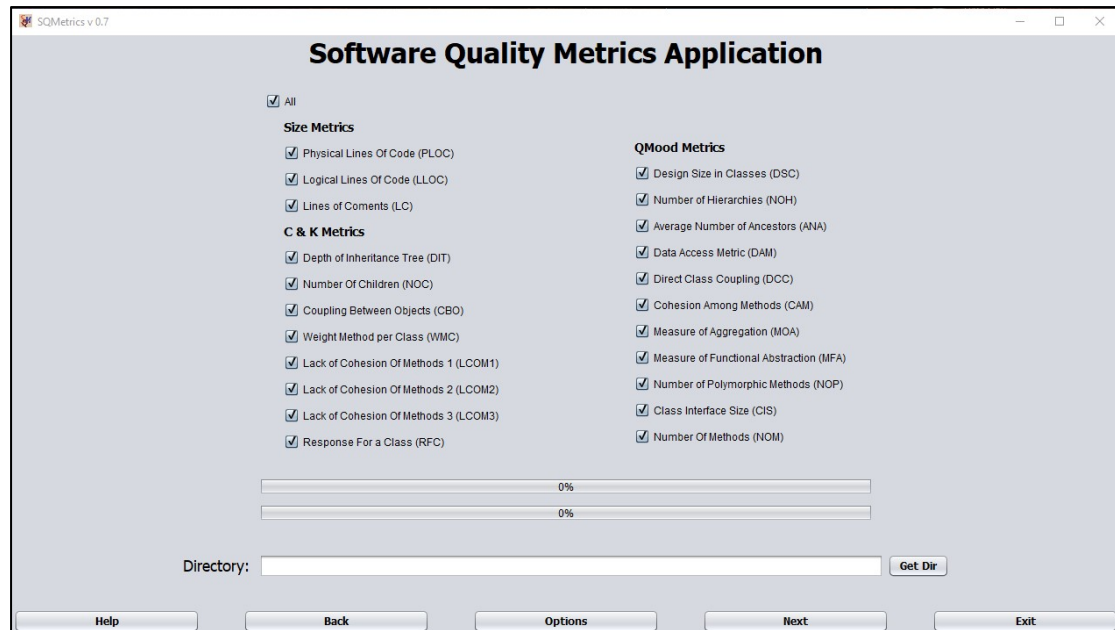
Picture 2: “Help” window

The “Exit” button, on all screens, terminates the application.

On this screen you can do the two major actions of the application. The primary “job” is to measure java code and the secondary is to compare software.

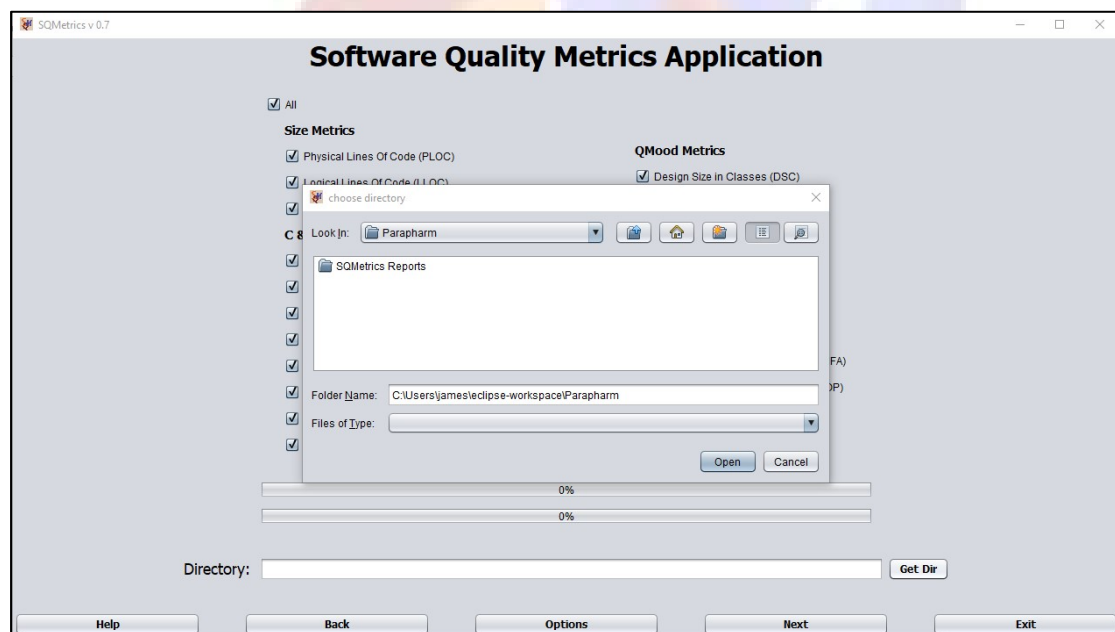
## MEASURING JAVA CODE

If you want to measure a java application, you must press the “Measure a Java Application” button. After that, the “Input” screen appears (Picture 3).



Picture 3: Input Screen

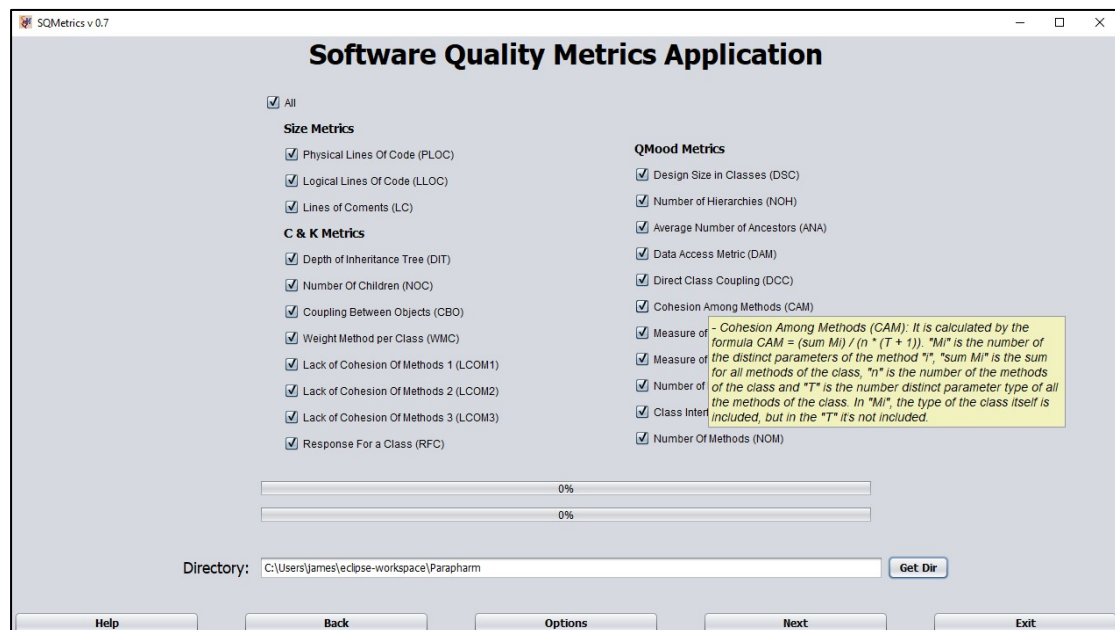
- At first, you must choose the directory that contains both, the source code (\*.java files) and the binary code (\*.class files). If they exist at different folders, then you must choose their common parent folder<sup>1</sup>. To do this, you must push the “Get Dir” button and the window to choose directory appears (Picture 4).



Picture 4: "Choose Directory" window

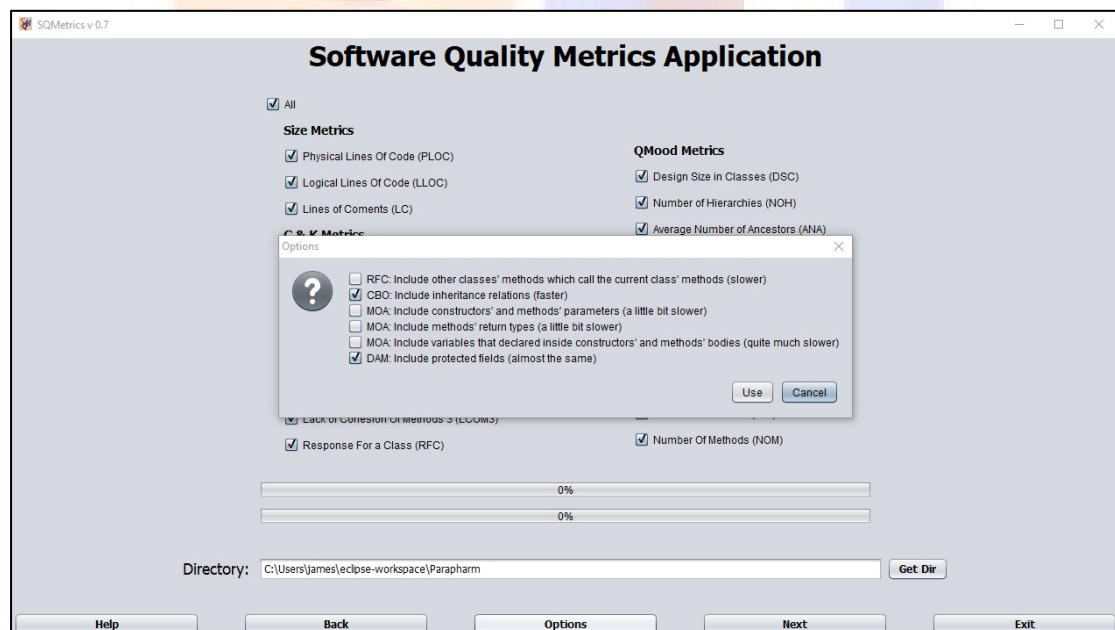
<sup>1</sup> If there is no binary code (\*.class files), the application will not measure anything. So, you have to compile your code first, so the \*.class files have been created.

- After choosing directory, you must choose which metrics to be counted. Feel free to choose any of the metrics or the whole set of them. A tool tip is shown when you hover the mouse pointer over each metric, with its description (Picture 5).



Picture 5: Tooltip for the "CAM" Metric

If you push the "Options" button, you have the opportunity to configure some of the metrics (Picture 6).



Picture 6: "Options" window

**Options:**

- RFC: According to C&K, there is no need to calculate the other classes' methods, which call the current class' methods, but for others they must be counted.

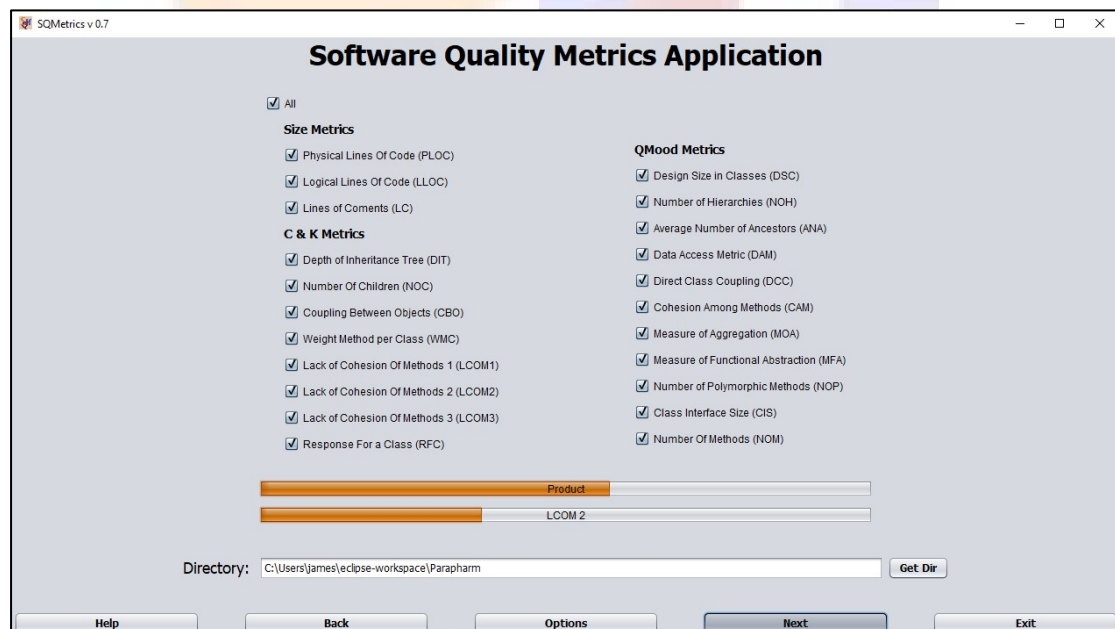
- CBO: According to C&K, the inheritance relations must not be calculated for coupling. Others believe that they must, so there is an option for the user to choose.

- MOA: B&D do not specify which variables to consider for the calculation of the MOA metric. Three options have been added. With the first one checked, the constructors' and methods' parameters, will be included. With the second, the return types of the methods, will be included. With the third option checked, the parameters that are defined into the method's bodies, will be included. The last one is very slow, because there is no other way to find the parameters that are declared inside the body of the methods, beside parsing.

- DAM: B&D, for the DAM metric, use the exact phrase: "the number of the private (protected) attributes". Of course, these are two different meanings in OO Programming. Therefore, the user will have to decide, whether the protected fields will be counted or not.

With the "Back" button you return to the "Welcome" screen. To proceed you have to push the "Next" Button.

- After that, a progress bar will be displayed (Picture 7). The name of the current class and the current metric is been displayed.



Picture 7: Bar progress

Then the “Output” screen appears (Picture 8).

**Software Quality Metrics Application**

Package	Class	PLOC	LLOC	LC	DIT	NOC	CBO	WMC	LCOM 1	LCOM 2	LCOM 3	RFC	DSC	NOH	ANA	Df
	Cosmetic	23	15	0	2	0	3	2	0	0	0	3	7	1	1,29	
	CosmeticsList	20	12	0	1	0	2	5	0	0	0	3	7	1	1,29	
	Paramedical	25	16	0	2	0	3	2	0	0	0	3	7	1	1,29	
	ParamedsList	23	12	0	1	0	2	6	0	0	0	4	7	1	1,29	
	Product	39	25	1	1	2	3	5	0	0,42	1	4	7	1	1,29	
	ProductsList	13	7	0	1	0	1	2	0	0	0	3	7	1	1,29	
	RunMe	145	104	11	1	0	4	32	0	0	0	8	7	1	1,29	
PROJECT'S	SUM OR AVG	288	191	12	1,29	0,29	2,57	7,71	0,0	0,06	0,14	4,0	7	1	1,29	

**Detailed Report**

REPORT FOR PROJECT: Parapharm, (by SQMetrics v 0.7) 05 Jul 2020 - 12.42

Class: Cosmetic  
 Package: -  
 Super Class: class Product  
 Sub Classes: none

Fields: myType: private

Methods: eatTimes: public CC= 1

Main Menu Back Separate char = ; Export Exit

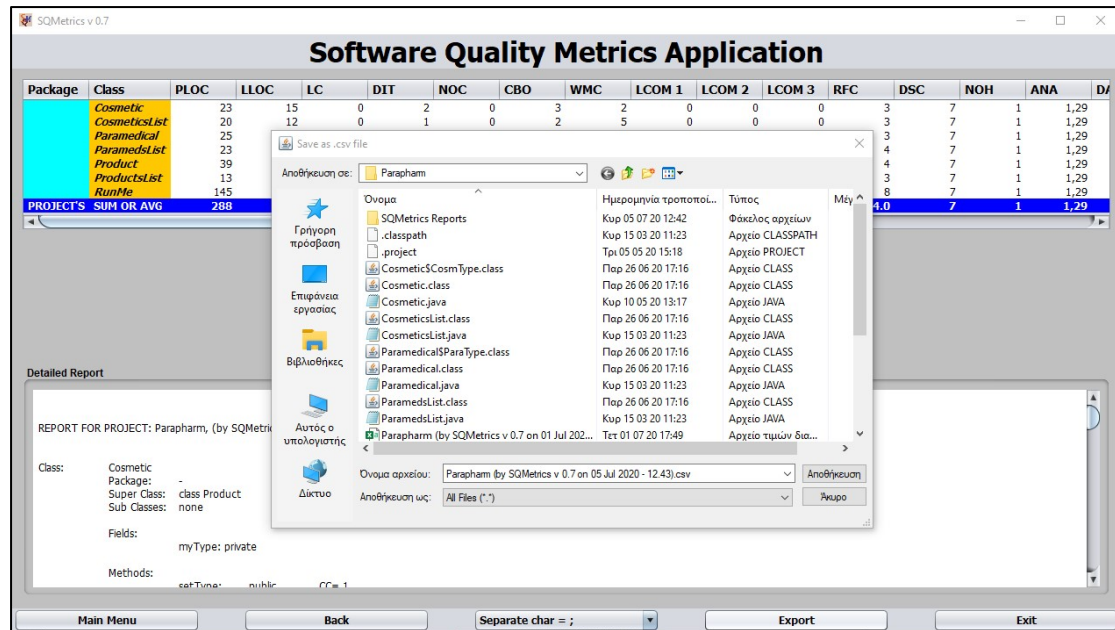
Picture 8: "Output" screen

A table with the metrics results appears on the upper half of the screen and a detailed text report on the bottom half of the screen. The report file is saved automatically into the directory you chose with the “Get Dir” button.

With the “Export” button, you have the option to export the results into a .csv file. In this case, you have to choose where to save the file (Picture 9). The default place is the directory you chose, but you can change it. The name of the file is already written but you can change it, too. If you don't add the .csv extension or if you add a wrong one, the program will correct it. The default separator is the semicolon (;), but you can change it from the pull-down menu<sup>2</sup>. The other possible choices are “coma” (,) and “TAB” character.

<sup>2</sup> Before you press the “Export” button.



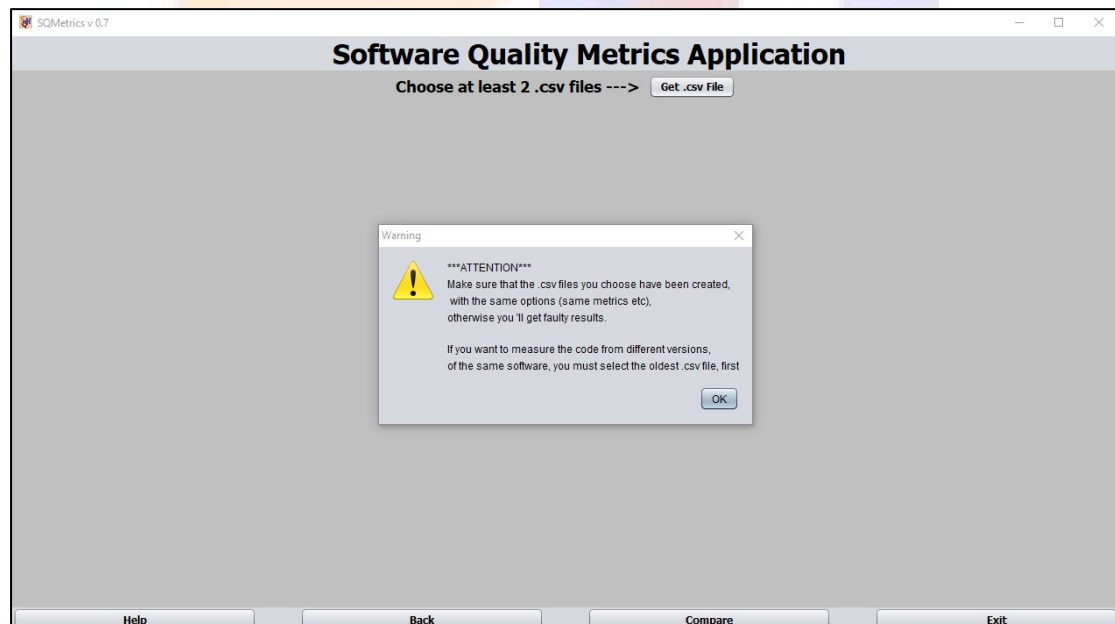


Picture 9: "Export" window

You can return to the "Input" screen with the "Back" button and choose a different directory or to the "Welcome" screen with the "Main Menu" button.

## COMPARING SOFTWARE

If you push the "Compare Software" button on the "Welcome" screen, you will face the "Compare" screen with a warning message (Picture 10).



Picture 10: "Compare" screen

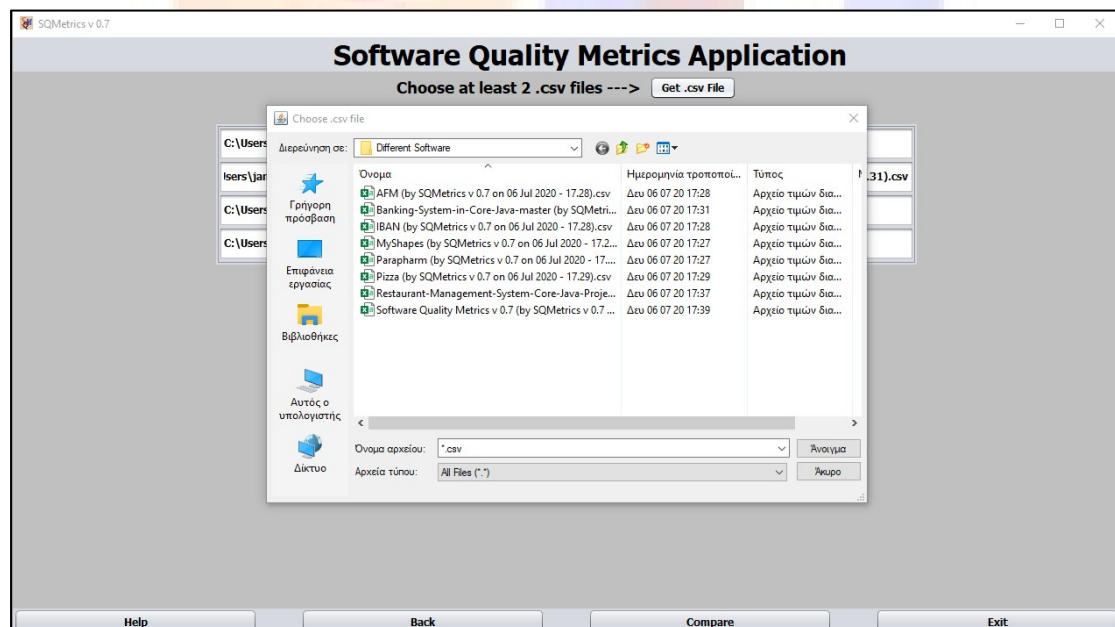


The SQMetrics application has the ability to compare either different versions of the same software, or different software applications, as suggested by Bansiya and Davis in 2002 (they proposed the QMOOD model). In both cases there is a need to normalize the values.

To compare different versions of the same software, the results corresponding to the first version are used as a basis. The prices of all editions are divided by those of the first edition. So, the range of the values will be similar. Then the quality characteristics of each version are calculated, based on the normalized prices. This shows the progress on each characteristic, between the different versions of the software (Picture 14).

If you want to compare  $x$  different software applications, for each design property (metric),  $x$  corresponds to the application with the highest value, followed by the remaining numbers numbered in descending order. If a number ( $m$ ) of applications have the same value ( $k$ ), then they receive the same numbering ( $k$ ), but the next one will be ranked at  $k-m$ . For example, if 3 applications are ranked together in position 4, the next one will be ranked in position 1 (Picture 13).

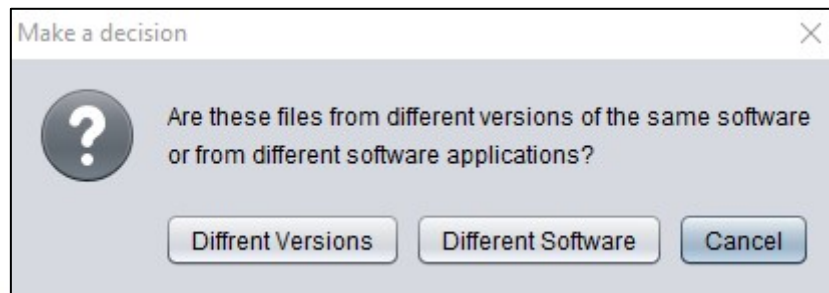
The warning message that appears prompts us to make sure that the .csv files we select have been created by the same version of the SQMetrics application and with the same options selected. Also, if we choose to compare different versions of the same software, we must select the .csv file that corresponds to the older version, first. This is because the application normalizes the values, based on the first selected file.



Picture 11: Input .csv files

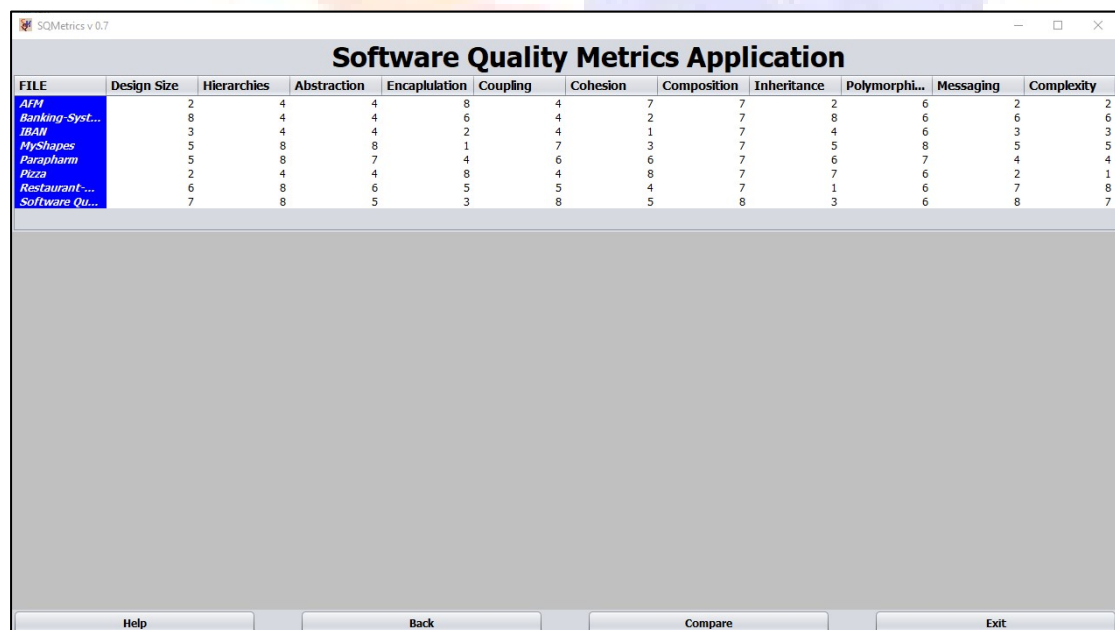
As you select .csv files, they are added on the screen (Picture 11).

When you finished and press “Compare”, a message appears prompting you to select method of comparison (Picture 12).



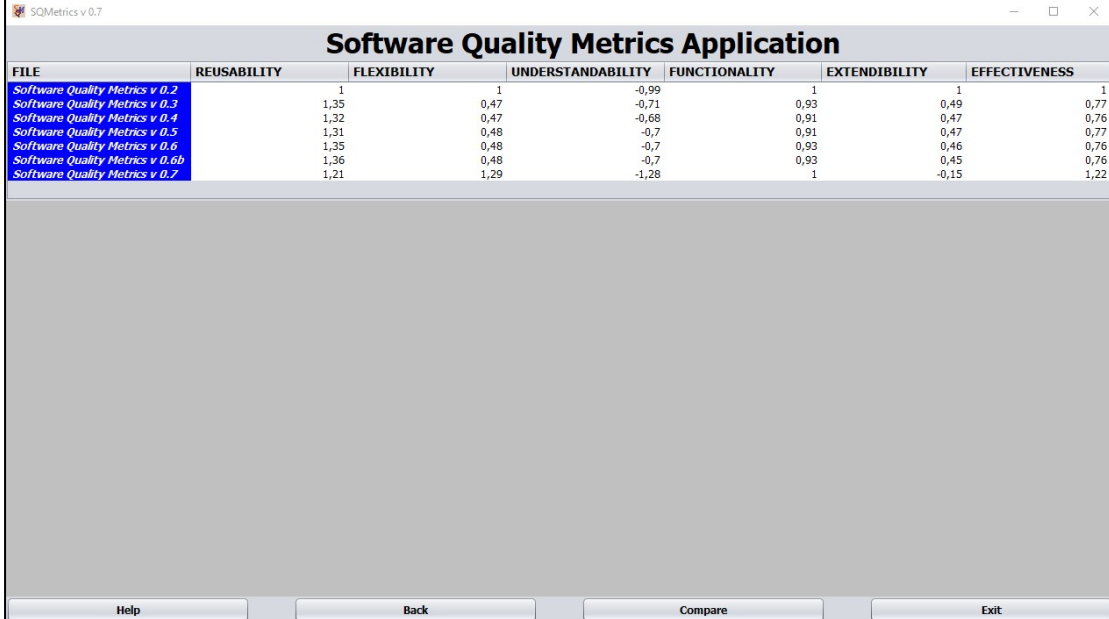
Picture 12: Select method of comparison

Then, depending on the comparison method you have selected, the corresponding results screen is been displayed (Picture 13 or Picture 14).



FILE	Design Size	Hierarchies	Abstraction	Encapsulation	Coupling	Cohesion	Composition	Inheritance	Polymorphi...	Messaging	Complexity
AFM	2	4	4	8	4	7	7	2	6	2	2
Banking-Syst...	8	4	4	6	4	2	7	8	6	6	6
IBAN	3	4	4	2	4	1	7	4	6	3	3
Myslapes	5	8	8	1	7	3	7	5	8	5	5
Parapharm	5	8	7	4	6	6	7	6	7	4	4
Pizza	2	4	4	8	4	8	7	7	6	2	1
Restaurant-...	6	8	6	5	5	4	7	1	6	7	8
Software Qu...	7	8	5	3	8	5	8	3	6	8	7

Picture 13: Comparison results for different software applications



FILE	REUSABILITY	FLEXIBILITY	UNDERSTANDABILITY	FUNCTIONALITY	EXTENDIBILITY	EFFECTIVENESS
Software Quality Metrics v 0.2	1	1	-0,99	1	1	1
Software Quality Metrics v 0.3	1,35	0,47	-0,71	0,93	0,49	0,77
Software Quality Metrics v 0.4	1,32	0,47	-0,68	0,91	0,47	0,76
Software Quality Metrics v 0.5	1,31	0,48	-0,7	0,91	0,47	0,77
Software Quality Metrics v 0.6	1,35	0,48	-0,7	0,93	0,46	0,76
Software Quality Metrics v 0.6b	1,36	0,48	-0,7	0,93	0,45	0,76
Software Quality Metrics v 0.7	1,21	1,29	-1,28	1	-0,15	1,22

Picture 14: Comparison results for different version of the same application

## DISCRIPTION OF THE METRICS

- The application supports 3 size metrics, the 6 Chidamber & Kemerer metrics and the 11 QMOOD metrics. It also calculates the 6 quality characteristics of the QMOOD model.

### Size Metrics

- *Physical Line of Code (PLOC)*: It counts the "CR" or "ENTER" characters. All the lines are counted, included the comment and the blank lines.
- *Logical Lines of Code (LLOC)*: All lines that contain code are counted. If a line contains both code and comments it will be counted. The lines with only 1 bracket are not counted.
- *Lines of Comments (LC)*: Only the lines that contain comments are counted. If a line contains both code and comments it will be counted.

### Chidamber & Kemerer Metrics

- *Number of Children (NOC)*: The NOC of a class is the number of the classes that inherit from this class.

- *Depth of Inheritance Tree (DIT)*: It calculates the depth of the inheritance tree of a class, into the project. If the class inherits from an external class out of the project, it's DIT is "1".
- *Coupling Between Objects (CBO)*: Two classes are coupled when one of them uses methods, fields or objects of the other class. There is an option to count the inheritance.
- *Weight Method per Class (WMC)*: It counts the sum of the McCabe's (1976) cyclomatic complexity of all the methods of the class. Constructors are excluded.
- *Lack of Cohesion of Methods (LCOM)*: There are 3 types. LCOM1, LCOM2 and LCOM3. The LCOM1 is the oldest and is an integer. It adds "1" for every pair of methods that use at least 1 common instance field and abstracts "1" for every pair which doesn't. If it is below "0", then it is "0". The LCOM2 is calculated by the following formula:  $LCOM2 = 1 - \frac{\sum(nA)}{(m * a)}$ , where m = number of methods and a = number of fields, nA = number of methods that access the field "A". LCOM3 is a little bit different.  $LCOM3 = (m - \frac{\sum(nA)}{a}) / (m - 1)$ . LCOM2 takes values between "0" and "1" and LCOM3 between "0" and "2". For all 3 metrics, the closer to "0" the better. Getters and setters such as constructors are excluded.
- *Response For a Class (RFC)*: It is the number of methods that could be called to answer a message from an object of another class. It is calculated by the formula:  $RFC = M + R$ , where M is the number of the methods of the class and R is the number of methods of other classes, calling directly some method of the class. Constructors are included. There is an option to add the methods of the other classes, twich call methods of the current class.

### **QMOOD Metrics**

- *Design Size in Classes (DSC)*: It is the number of the classes of the project.
- *Number Of Hierarchies (NOH)*: It is the number of the subtrees of the project, under the System.Object class, which means the number of the classes that have at least 1 child and DIT = 1.
- *Average Number of Ancestors (ANA)*: It is the average of the classes from which a class inherits. It is given by the formula  $ANA = DIT / DSC$ .
- *Data Access Metric (DAM)*: It is the division between the number of "private" and "protected" fields by the total number of fields. The closer to "1" the better.
- *Direct Class Coupling (DCC)*: It is the number of the other classes that are directly coupled with this class. A class is directly coupled with another class when it is parent or child of the other class, it has a field with the type of the

other class, or its methods return a value or have a parameter with a type of the other class.

- *Cohesion Among Methods (CAM)*: It is calculated by the formula  $CAM = (\sum Mi) / (n * (T + 1))$ . "Mi" is the number of the distinct parameters of the method "i", "sum Mi" is the sum for all methods of the class, "n" is the number of the methods of the class and "T" is the number distinct parameter type of all the methods of the class. In "Mi", the type of the class itself is included, but in the "T", it is not included.
- *Measure of Aggregation (MOA)*: This method returns the number of variables which are declared by the user. This means the variables, that their type is another class of the project. By default, it counts the class's (and instance) variables. There are also 3 options to include the constructor's and methods' parameters, the return types of the methods and the variables that are declared inside the constructors or the methods. The 3<sup>rd</sup> option is very slow.
- *Measure of Functional Abstraction (MFA)*: It is the "Measure of Functional Abstraction" (MFA) metric, which is the division between the inherited methods, by all the methods of the class (inherited and declared).
- *Number of Polymorphic Methods (NOP)*: It is the number of the methods of a class, which may have polymorphic behavior.
- *Class Interface Size (CIS)*: It is the number of the "public" methods of the class.
- *Number of Methods (NOM)*: It is the number of the methods that are declared in a class, which means the number of the public, protected, default (package) access, and private methods of the class. Constructors are excluded.

### QMOOD Quality Characteristics

Each of the following characteristics is been calculated, only if all of its parameters have been checked for calculation.

- $REUSABILITY = (-0.25) * DCC + 0.25 * CAM + 0.5 * CIS + 0.5 * DSC$
- $FLEXIBILITY = 0.25 * DAM - 0.25 * DCC + 0.5 * MOA + 0.5 * NOP$
- $UNDERSTANDABILITY = (-0.33) * ANA + 0.33 * DAM - 0.33 * DCC + 0.33 * CAM - 0.33 * NOP - 0.33 * NOM - 0.33 * DSC$
- $FUNCTIONALITY = 0.12 * CAM + 0.22 * NOP + 0.22 * CIS + 0.22 * DSC + 0.22 * NOH$
- $EXTENDIBILITY = 0.5 * ANA - 0.5 * DCC + 0.5 * MFA + 0.5 * NOP$
- $EFFECTIVENESS = 0.2 * ANA + 0.2 * DAM + 0.2 * MOA + 0.2 * MFA + 0.2 * NOP$

## LIMITATIONS

- If, for some reason, the source code is missing or if the code has not been compiled yet (so the .class files do not exist), then the program will run but will not work properly and no results will be displayed. So, you have to compile your program first, so the \*.class files must exist and should have the same names as the \*.java files do. If a ".java" file is missing, the program will run and you'll get the results for the other classes.

- The declarations of the methods into the code, must be in 1 line, otherwise the results of some metrics will be slightly different.

- Each .java file must correspond to a single class. If there is more than one class code in the same .java file, only one will be calculated and the results will be incorrect.

- Internal classes are not counted as separate classes and they are counted as part of the code of the method they belong.

## REQUIREMENTS

- This program will run on Microsoft Windows, Linux and Mac OS X. It has been tested on Microsoft Windows 7 and 10, on Ubuntu Linux and on Mac OS X High Sierra. It should run on the other versions of these platforms too, but has not been tested.

- JRE 1.8 or JDK 1.8 (or newer) is required to be installed on your system, so this program is able to run.

## KNOWN BUGS

- On some Windows systems the runnable ".jar" file won't run. In this case you have to execute the "RunMe\_on\_Windows.vbs" script and the program will run perfectly.

- If you encounter the same problem on Linux, run the "Run\_Me\_On\_Linux.bat" file.

- Please, report any new bug at [std131050@ac.eap.gr](mailto:std131050@ac.eap.gr) or at [sofjim65@gmail.com](mailto:sofjim65@gmail.com).

## HISTORY

v 0.1:

- All 6 Chidamber & Kemerer metrics are supported, plus 3 size metrics as physical and logical lines of code and lines of comments.

- The results are displayed in the terminal.

*v 0.2:*

- A graphical environment and the ability to export data to a.csv file have been added.

*v 0.3:*

- All 11 QMOOD metrics have been added.

*v 0.4:*

- Calculation for the 6 quality characteristics, that the QMOOD model suggests, has been added.

- A bug, that led to wrong calculation of the NOM metric and, by extension, to other metrics such as LCOM1, RFC, CAM etc, has been fixed.

*v 0.5:*

- The ability to choose the separate character for the exported csv file, between coma (,) and semicolon (;), has been added.

- Tooltips' delay time increased to infinity.

*v 0.6:*

- Improved and faster algorithm for LCOM1, LCOM2 and LCOM3 metric (now only instance fields are counted).

- Names of classes and metrics, now appear on the progress bar.

*v 0.6b:*

- No changes for the environment of the application have been made, but the speed for some algorithms was increased.

*v 0.7:*

- Total new interface. More flexible, more stable.

- Faster algorithms up to 60% for many metrics.

- The problem with non utf-8 characters has been fixed. Now all sets rare recognized.

- The ability to compare different versions of the same application or different applications, as it has been introduced by Bansiya, J., & Davis, C. G., 2002 has been added.

- A project report has been added to inform the user for the classes, methods and fields of the whole project, including the cyclomatic complexity of each method



and the modifications of the methods and the fields. This report is been saved to a text file into the project directory.

- An "Options" button has been added and the user has the ability to choose some parameters for some metrics, that are not clearly defined.

