Free/Libre and Open Source Software Metrics

Sponsored through Framework Programme Sixth (Call 5) by

European Commission

Information Society Technologies

Document Information

Version: 1.0
Date : Oct. 21. 2009
revision: 0

Owning Partner: UM

Author(s):
Kirsten Haaland
Sulayman K. Sowe

Reviewer(s):
Ruediger Glott
Rishab A. Ghosh
Santiago Dueñas

To:
Public

Purpose of distribution:

The FLOSSMetrics Consortium consists of: Universidad Rey Juan Carlos, University of Maastrich, Wirtschaftsuniversitaet Wien, Aristotle University of Thessaloniki, Conecta s.r.l., Zea Partners and Philips Medical Systems PMS Nederland B.V.

Printed on at

Status:		**Confidentiality:**	
[ ]	Draft	[X] Public	- Intended for public use
[ ]	To be reviewed	[ ] Restricted	- Intended for FLOSSMETRICS consortium only
[ ]	Proposal	[ ] Confidential	- Intended for individual partner only
[ ]	Final/Released		
X			

Deliverable ID: D11.1

Title:

Metrics Dictionary

Deliverable: D11.1

Title: Metrics Dictionary

Executive Summary:

This report translates the metrics identified by software engineering tools, detailed in deliverable D.4.1. “Classification Report”, and translates them into socio-economic terms. Since the benefit of research that the FLOSSMETRICS enables in interdisciplinary in nature, it is important to have a mapping like this to ensure definitions are clear and to point out possible research opportunities, and ensure quality. Already, the software engineering metrics have been utilized and translated within this project, in the deliverable D8.1 “FLOSS Guide to SMEs” and in the various high level studies in WP5 and WP11.

CHANGE LOG

Ver.	Date	Author	Description
0.1	10/09/09	Sulayman K. Sowe	Initial proposal for structure
0.2	28/09/09	Kirsten Haaland	Edit structure and content
0.3	03/10/09	Kirsten Haaland	Add combined metrics
0.4	07/10/09	Kirsten Haaland	Complex variables
0.5	09/10/09	Kirsten Haaland	First draft
1.0	21/10/09	Santiago Dueñas	Review and release

APPLICABLE DOCUMENT LIST

Ref.	Title, author, source, date, status	Deliverable Identification

TABLE OF CONTENTS

1. Introduction

Some of the most contemporary questions software engineers, in general, and Libre software participants, in particular face are “What percent of your time do you spend on libre projects at work and outside of work”? “What percent of your development time that do you spend on libre projects”? “How much time do you actually spend coding”? These questions are imperative but have proven difficult to answer independently from either the engineering and economic perspective. The inability to provide full or partial answers to some of these questions can largely be attributed to the lack of appropriate metrics to provide us the benchmark needed to measure individuals participation in Libre software projects. The FLOSSMETRICS dictionary bridges the gap between software engineering and economics, explaining the software engineering metrics collected by the FLOSSMETRICS database in socio-economic terms wherever possible.

The characteristic of the dictionary is to answer questions as outlined in the description of work, such as : is a CVS-“committer” the original producer of a piece of software? Its owner? A value-adder (editor) rather than original producer (author)? If source lines of code (SLOC) for a particular project represents net production, what is the equivalent to gross production? The word “metric” itself which has made it into the FLOSSMETRICS project acronym and the title of this deliverable might be worth to clarify at this stage, as it is a popular word in software engineering indicating a standard of measurement, while in economics this vocabulary is not often used, and generally one talks about variables instead of metrics.

The metrics dictionary draws on deliverable D.4.1. “Classification Report” that identifies three main groups of variables, namely raw variables, combined variables and complex variables. Further, other software engineering metrics that have been utilized and translated within this project in the deliverable D8.1 “FLOSS Guide to SMEs” and in the various high level studies in WP5 and WP11 are also included.

The structure of the deliverable follows the structure outlined in the “Classification Report” D 4.1. First the simple first layer variables are presented in section 2, section 3 translates the combined variables, followed by the complex variables in section 4.

2.Simple first layer variables

It is important to understand the basic building blocks, and document them well when doing analysis. The simple variables, raw variables or first layer variables are data directly as it is retrieved from software repositories and stored in the corresponding database. As they are the building blocks for combined variables and indicators, and complex variables, it is important to have a thorough understanding of these. The simple first layer variables come from three sources, the Source Code Management System (2.1) which encompasses both SCM own variables and file variables, from the Mailing lists (2.2) and from the Tracking systems (2.3).

2.1Source Code Management Systems

This table maps the variables from the Source Code Management Systems, first simple layer.

Metric	***Metric Description***	***Socio-Economic Translation***
SCM own variables		
Repository URL	Address of the analysed repository	Enables replication of the study, thus it is important to specifically know where the data comes from.
Author	Developer of the changes in the source code. The components of this variable are: Author ID., Author name and Author email. (The author name and author email is not provided in the anonymous version of the database or in public results, due to privacy.)	The author is the original producer of a piece of code. This is a value-adding activity, and represents primary production. The author does not necessarily have commit rights to the repository, but may he have so. Thus, not all authors are committers. The author is not

Metric	*Metric Description*	*Socio-Economic Translation*
		always the the copyright holder of the code he produces. His employer/school could be, depending on his employment/school contract. Note that identified authors are identified per repository, and does not guarantee uniqueness across repositories as people can use the same nick names and real names.
Committer	Person responsible for applying the set of modifications or changes (a commit), submitted by the author or the developer, on the repository. This person has write access to the repository and might not be the author of the changes.	Having committing rights does not necessarily mean one is the producer of the code, a commiter can commit code written by other authors, as well as own code. Having commit rights generally is a gained right, and reflects that the relevant community places trust in this committer. The value added of committing code written by other authors is in quality control, integration and management. Being aware of this distinction between author and committer is for example important for productivity studies. “Gate keepers” (submitting much code they did not write) should possibly be filtered out or dealt with differently.
Commit date	Date when the changes were applied in the repository.	Date and time (generally) are obviously very important for any

Metric	***Metric Description***	***Socio-Economic Translation***
		socio-economic analysis related to changes, such as evolution, growth, etc.
Commit message	Message that describes the objective or the motivation of the changes.	Can possibly help to place a commit in a relevant socio-economic context.
Revision file identifier	Identifies the status of a file at a specific point in time. The status of a file is the set of name, path, branch and content that the file has at a certain instant of time.	-
Revision repository identifier	Identifies the status of the repository at a specific point in time. The status of a repository is a repository snapshot with a set of files, which their own status, at a certain instant of time.	-
Action type	Type of the changes applied to a file on a commit. Types include (but are not necessarily supported by all repositories): *Added*, *Modified*, *Deleted*, *Moved, Copied*, *Replaced.*	This is very important to carefully decide what type of action is of interest when studying a given phenomena, and make the correct sub-selection. For example, for productivity, it takes much less time (close to 0) to delete a large amount of code, while it takes much time to write/add code. Thus for productivity one could decide to not include deleted actions. Further, copied or moved does not necessarily include effort in terms of primary production. In other studies one may wish to include these actions (e.g. looking at time spent on various activities); it is just

Metric	*Metric Description*	*Socio-Economic Translation*
		important to carefully know what phenomena one wants to study and include the relevant action types.
Tag	Label for identifying a snapshot of the repository at a specific point in time.	-
Branch name	Name of a line of development. This lines are independent but can share the status of a set of files between them. The main development line is usually named 'trunk'.	-
File variables		
File name	Name of a file at a specific instant of time.	-
File path	Path of a file at a specific instant of time.	-
File branch	Branch in which a file can be found at a specific point in time.	-
File extension	Extension of a file, if any.	-
File type	Type of a file according its extension and properties. Possible types: *build*, *devel-doc, documentation*, *image, i18n, multimedia,* package, source, ui, unknown.	Important for type of activity in question, and deciding which level to analyse, so as not to compare apples and pears.
File language.	Programming language of a source code file. The possible languages are those that SLOCCcount tool[1] supports. This list contains almost 30 languages including C, C++, C#, Java, Perl, Ada, Python or Assembly.	Comparing various programming languages could be problematic since in terms of SLOC it can take a different amount of effort to achieve the same functionality. I.e. some programming languages are more 'efficient' in terms of code than others.
File LOC	Number of lines of code (LOC) of a file in a specific revision. A line of code is any line of program text, including comment lines, blank lines, etc.	To be aware of how LOC (and SLOC) is counted is very important since this is one of the most popular measures of output in research on FLOSS. Generally, SLOC is

[1] http://www.dwheeler.com/sloccount/

Metric	***Metric Description***	***Socio-Economic Translation***
		preferred to LOC, as it leaves out blank lines and comments. However for some size measure LOC might be interesting metric.
File SLOC	Number of physical lines of code (SLOC) of a file in a specific revision. The definition of SLOC is the same used by SLOCCount: "*a physical source line of code (SLOC) is a line ending in a newline or end-of-file marker, and which contains at least one non-whitespace non-comment character. Comment delimiters (characters other than newlines starting and ending a comment) are considered comment characters. Data lines only including whitespace (e.g., lines with only tabs and spaces in multiline strings) are not included*"[2].	This is a very important output measure of source code, as it is a measure of production. File SLOC is a stock measure of the size of a file. It is also a gross measure of code production, and it is not taking into account for parts of code being reused. (If one corrects for reuse – it would become a net measure). Another approach is to consider SLOC the net production, and overall production to include a wide variety of other important activities, such as mailing list, bug reporting, management, documentation, etc. There are obviously many activities that takes place that are not captured by SLOC measures; which basically is a size metric.
Blank lines of a file	Number of blank lines including lines with white spaces and tabs.	-
Comment lines of a file	Number of lines with comments on a file.	Indicates how well documented a file is. Has links to issues such as quality.

[2] http://www.dwheeler.com/sloccount/sloccount.html

Metric	***Metric Description***	***Socio-Economic Translation***
Comments of a file	Number of comments the file contains.	Indicates how well documented a file is. Has links to issues such as quality.
Functions of a file	The number of functions the specific file contains.	Generally, the more functions, the higher complexity.
McCabe's cyclomatic complexity of a file.	The definition of this metric is the given by McCabe[3]. McCabe's complexity is defined at function level but the database only stores information of files. Therefore, instead of store cyclomatic complexity values per function, other variables were defined minimizing information loss.	Generally, the higher complexity, the higher the risk, and possibly the higher management and maintenance costs.
Halstead's software science metrics of a file[4]	The definitions of these metrics are the given by Kan, namely Halstead length: Variable that includes the total number of operator occurrences and total number of operand occurrences in a file. Halstead volume: This is a Halstead metric that contains the minimum number of bits required for coding the file. Halstead level. Level at which the source code of a file can be understood. Halstead difficulty. Level of difficulty in the file.	These are technical software engineering metrics measures, that could give an indicator of the difficulty to manage a file, and the value added in terms of coordination costs.

[3] Thomas J. McCabe. A complexity measure. *IEEE Translations of Software Engineering*, SE-2(4):308-320, 1976

[4] Stephen H. Kan. *Metrics and Models in Software Engineering (2nd Edition).* Addison-Wesley Professional , September 2003

2.2 Mailing List Metrics

The following variables are available for mailing lists repositories:

Metric	*Metric Description*	*Socio-Economic Interpretation*
Mailing list address	Address of the mailing list.	To be able to replicate the study and specifically know where the data comes from.
User	Person which her email address appears, as a sender or as one of the receivers, in any of the emails sent to the mailing list. The components defined for users are the following: ○ User ID. Identifier of the user. ○ User name. This name is the user's nick or her real name, if was given. ○ User email address.	For the public database and public results, user name and address is unavailable due to privacy reasons. It is further important to note that the same nick name or real name in two mailing lists does not enable to uniquely link individuals across mailing lists.
Message identifier	Hash that identifies the message as unique from other messages.	-
Message subject	The textual subject of the email message.	Could be useful for placing a message in a topic space and in a socio-economic context.
Message body	The body or content of the email message.	Could be useful for placing a message in a topic space and in a socio-economic context.
Message date	Local time and date when the client sent the message to the mailing list. From the date of the message two components are defined: *Message date*. The local time and the date of the message itself. *Message date time-zone*. The	For example looking at activity levels, participation etc, it is important to take account of time zones.

Metric	*Metric Description*	*Socio-Economic Interpretation*
	region where the client was sending the message.	
Mailing list date	This date refers to the local time and date when the server, that manages the mailing list, deliver the message to its users. Like the message date, two components are defined: *Mailing list date.* The local time and the date of the deliver itself. *Mailing list time-zone.* The region where the server was delivering the message.	As for message date, mailing date and correct time zone correction is important for mailing list activity levels.
Message poster	Identifier of the person who sent the message.	Identifying relevant agents for analysis.
Message receiver	Identifier of the entity (mailing list or person) who received the message.	-
Message response	Identifier of the message that is a response of other one.	Providing networks/links of connections and correspondence.

2.3 Tracking Systems metrics

This section describes all the variables defined for tracker systems. The list is the following:

Metric	*Metric Description*	*Socio-Economic Interpretation*
Tracker URL	Address of the analysed tracker.	Enables replication of study, and as for SMC and Mailing list data, it is important to know exactly where the data comes from.
Contributor identifier	Unique identifier of a person in the system. The user can have different roles such as reporter or fixer and so on.	A unique ID identifying agents, but does not classify or identify the roles the agent is performing.

Metric	***Metric Description***	***Socio-Economic Interpretation***
Submitter	Contributor that sent the report to system.	A specific role by a contributor that initiates one activity/record.
Summary of the report	Brief description of the report	Information that can help set the socio-economic context.
Report description	Textual description of the report.	Information that can help set the socio-economic context.
Report date	Date when the report was notified to the system.	
Report category	Type of the report.	
Report status	Status of the report at a certain instant of time. It can take one of the following values: ○ *open* : new report without assignation ○ *closed* : fixed report ○ *deleted* : invalid report ○ *pending* : assigned report but not fixed	The report status will typically change in the life cycle of the report. Further, different trackers and corresponding communities may have different policies regarding when to closing a report, etc. The current report status of a tracker is a stock measure, however with all the information how it has changed, one also has the flow of changes available.
Report priority	Urgency of the report at a specific point in time. Usually this field is modified by the tracker-master as users do not have sufficient knowledge on the software to know the correct value. It can take following values (from high priority to lower one): “immediate”, “urgent”, “high”, “normal” and “low”.	Report priority can also change over time. Different type of trackers (and projects) may have different levels/requirements of priority. Report priority may vary among contributors as perception of agents can be influenced by experience.
Report severity	Severity of the report at specific point in time. How this report affects the use and development of the software. Possible values are (from high severity to lower one): “blocker”, “critical”, “major”, “normal”, “minor”, “trivial” and “enhancement”.	As report priority, report severity can change over time, and may vary across trackers, and possibly more importantly, across perception of experienced and inexperienced contributors.
Assignation	Developer in charge of fixing the report at a	A role assigned. A report can be re-

Metric	***Metric Description***	***Socio-Economic Interpretation***
	specific point in time.	assigned to a different developer, so the point in time is also important when looking at the assignation.
Date of the comment	Date when a report comment was posted.	-
Body of the comment	Textual description of a report comment.	May provide contextual information.
Change type	Value change of any of the variables of a report at a certain point in time.	This keep track of changes, i.e. the flow or dynamics of the tracker; and at any given time the tracker has a stock of values for all variables.
Date of the change	Date when the change of a variable was performed.	-
Change description	Textual description of a change.	Possibly provide contextual information.
Name of the attachment	Name of the element attached to the report.	-
Description of the attachment	Textual description of the element attached to the report.	-
URL of the attachment	Address of the element attached to the report.	-

3. Combined Metrics

The combined metrics/variables are second layer variables derived from the simple variables above, by combining or aggregating them. They focus on combining and aggregating them, such as mean, the minimum and maximum values, values for some given period of time, or the aggregated value for a population of items. Generally, anything to do with minimum and maximum deals with variance, and if there is a time element it is useful for growth and evolution. Some of these statements for the relevant queries can be found in Melquiades documentation pages[5].

The combined metrics and variables are combined for the three sources: Source Code Management System (3.1) , Mailing lists (3.2) and from the Tracking systems (3.3).

3.1 Combined metrics based on Source Code Management Systems

Metric based on	*Metric Description*	*Socio-Economic Translation*
Commits		
	Total number of commits	The overall amount of instances of commiting by all committers. Number of commits says something about production and activity. However, it does not take into account the size of a commit, thus a 1 line commit is counted as 1, and a 100 line commit is counted as 1.
	Number of commits per unit of time	The number of commits per unit of specified time says something about growth (or decline) in activity level. It does not specify who

[5] http://melquiades.flossmetrics.org/wiki/

Metric based on	***Metric Description***	***Socio-Economic Translation***
		commits, nor the size of the commits.
	Aggregated number of commits over time	This gives the integral activity level in terms of number of commits up to a specified point in time.
	Maximum and minimum number of commits per unit of time	Maximum and minimum numbers of commits per unit of time provides insight into variance in activity level.
	Mean and median of commits per unit of time	Mean and median of commits per unit of time provides insight into variance in activity level overall for the SCM.
Actions		
	Total number of actions	Actions is about type of activity and activity level. Total number of actions Sum(*Added, Modified, Deleted, Moved, Copied, Replaced*) – or a sub-selection of relevant actions in question.
	Total number of actions per type	Sum of actions per type (*Added, Modified, Deleted, Moved, Copied, Replaced*).
	Number of actions per unit of time	The sum of number of action per specified unit of time. Useful for studying changes, such as growth and productivity.
	Aggregated number of actions over time	The integral of sum of actions up to a specified time.
	Maximum and minimum number of actions per unit of time	Maximum and minimum numbers of actions per unit of time provides insight into variance in activity level.
	Mean and median of actions per unit of time	Mean and median of actions per unit of time provides insight into

Metric based on	***Metric Description***	***Socio-Economic Translation***
		variance in activity level overall for the SCM.
	Number of actions per unit of time and per type	The sum of number of action per type of action, per specified unit of time. Useful for studying changes, such as growth and productivity.
	Aggregated number of actions per type overt time	The integral of the sum of actions per time up to a specified time.
	Maximum and minimum number of actions per unit of time and per type	Maximum and minimum numbers of actions per type per unit of time provides insight into variance in activity level.
	Mean and median of actions per unit of type and per type	Mean and median of actions per type per unit of time provides insight into variance in activity level overall for the SCM.
Committers / Authors		
	Total number of committers/authors	The total number of committers and authors in the SCM. The agents.
	Number of distinct committers/authors per unit of time	Agents per unit of time.
	Aggregated number of distinct committers/authors over time	The integral number participants/community members of SCM up to a specific time.
	Maximum and minimum number of distinct committers/authors per unit of time	Sheds light on variation in SCM community membership.
	Mean and median of distinct committers/authors per unit of time	Sheds light on variation in SCM community membership.
	Number of new committers/authors per unit of time	Growth in participation.
	Aggregated number of new committers/authors over time	Overall size of participation up to a given time.

Metric based on	***Metric Description***	***Socio-Economic Translation***
	Maximum and minimum number of new committers/authors per unit of time	Fluctuations in SCM community growth.
	Mean and median of new committers/authors per unit of time	The dynamics of new generations of community members entering the project.
Commits – Committers/ Authors		
	Number of commits per committer/author and per unit of time	An individual output measure, individual productivity and activity in terms of commits (not size adjusted).
	Aggregated number of commits per committer/author over time	The overall output and contribution of an individual up to a given point in time.
	Maximum and minimum number of commits per committer/author and per unit of time	Variations in individual output and productivity (not size adjusted) over time.
	Mean and median of commits per committer/author and per unit of time	Mean and median output in terms of commits over time.
	Maximum and minimum number of commits	Identifies the committer in the population with the largest number of commits, as well as the one with the smallest number of commits, helps to define the scale of production in the population.
	Mean and median of commits	The mean and median of commits for the population, helps to identify the production variance in the population.

Metric based on	***Metric Description***	***Socio-Economic Translation***
Actions – Committers/ Authors		
	Number of actions per committer/author and per unit of time	Individual activity level
	Aggregated number of actions per committer/author over time	Aggregate activity level in the history of a project up to a given time.
	Maximum and minimum number of actions per committer/author and per unit of time	Variance in activity level, and changes here in.
	Mean and median of actions per committer/author and per unit of time	-
	Maximum and minimum number of actions	Identifies the committer in the population with the largest number of actions, as well as the one with the smallest number of actions. Helps to define the range of production in the population.
	Mean and median of actions	The mean and median of actions for the population. Helps to define the variance of production in the population.
	Number of actions per committer/author, type and unit of time	-
	Aggregated number of actions per committer/author, type and over time	-
	Maximum and minimum number of actions per committer/author, type and unit of time	-
	Mean and median of actions per committer/author, type and unit of time	-

Metric based on	***Metric Description***	***Socio-Economic Translation***
	Maximum and minimum number of actions per type	Identifies the committers in the population with the largest and smallest number of actions per type.
	Mean and median of actions per type	The mean and median of commits for the population per type.
Files		
	Total number of files in the whole history of the project	Sum of all files.
	Number of files per unit of time	Projects evolve over time, functionality can be divided across file or in one larger (often more complex) file. The growth and evolution of the number of files is given by this metric.
	Maximum and minimum number of active (not deleted) files	The maximum and minimum of active files for the whole project.
	• Mean and median of active (not deleted) files per unit of time	A concentration measure of how much of the files are active.
	• Total number of files per type in the whole history of the project	-
	Number of files per type and per unit of time	Growth and evolution of file type
	Maximum and minimum number of active (not deleted) files per type	Per file type.
	Mean and median of active (not deleted) files per type	Mean and median of active files per type – for the whole project.
	Number of files per language	The overall distribution of files per language
	Aggregated number of files per language	Aggregated number of files per language for the whole project.
Commits – Files		
	Total number of commits per file	Overall effort and output (in terms of commits).

Metric based on	***Metric Description***	***Socio-Economic Translation***
	Number of commits per file and per unit of time	Evolution of commits per file.
	Aggregated number of commits per file over time	The sum of effort and output up to a given time.
	Maximum and minimum number of commits over files per unit of time	Variation in effort and output (in terms of commits).
	Mean and median of commits over files per unit of time	Variation in effort and output (in terms of commits).
Actions – Files		
	Total number of actions per file	Overall number of actions (i.e. added, modified, deleted, moved, copied, replaced) in project history.
	Number of actions per file and per unit of time	Changes/growth/evolution of actions over time.
	Aggregated number of actions per file over time	The sum of activity and actions up to a given time.
	Maximum and minimum number of actions over files per unit of time	Variation in the actions per unit of time.
	Mean and median of actions over files per unit of time	Variation in the actions per unit of time.
	Total number of actions per type and per file	The sum of various actions per file and per type.
	Number of actions per type, file and unit of time	The growth and evolution of the various actions.
	Aggregated number of actions per type, file over time	Sum of the number of overall actions per time up to a specific point in time.
	Maximum and minimum number of actions over files per type and per unit of time	Variation in actions per file and per type.
	Mean and median of actions over files per type and per unit of time	Variation in actions per file and per type.

Metric based on	***Metric Description***	***Socio-Economic Translation***
Size variables (LOC / SLOC)[6]		
	Total LOC/SLOC of the project in the whole history of the project	LOC and SLOC are size measures, and it represents the primary production and output. In the whole history of the project them can be taken to measure the overall effort
	Project size in LOC/SLOC per unit of time	The growth of the project size (and effort) over time.
	Maximum and minimum size in LOC/SLOC per unit of time	The size of the project changes as commits are made, files are added, deleted, etc. The maximum and minimum refers to the largest and smallest size as a stock variable.
	Mean and median of size in LOC/SLOC per unit of time	-
	LOC/SLOC added per unit of time	A measure of rate of change related to evolution and growth.
	LOC/SLOC deleted per unit of time	A measure of rate of change related to evolution and growth.
	Maximum and minimum LOC/SLOC added per unit of time	The growth of the project as a flow variable. A measure of gross productivity.
	Mean and median LOC/SLOC added per unit of time	A measure of gross productivity.
	Maximum and minimum LOC/SLOC deleted per unit of time	The decline of the project as a flow variable. Helps determine net productivity.
	Mean and median LOC/SLOC deleted per unit of time	Helps determine net productivity.

[6] Note that the notation of the metric indicate either LOC or SLOC, not the one divided by the other.

Metric based on	***Metric Description***	***Socio-Economic Translation***
Files – Size variables		Same as size variables for the overall project above, but per file.
	Maximum and minimum size in LOC/SLOC per file	
	Mean and median of size in LOC/SLOC per file	
	Maximum and minimum of size in LOC/SLOC per file	
	Mean and median of size in LOC/SLOC per unit of time	
	Maximum and minimum file sizes in LOC/SLOC per unit of time	
	Mean and median of file sizes in LOC/SLOC per unit of time	
	LOC/SLOC added per file and per unit of time	
	LOC/SLOC deleted per file and per unit of time	
	Maximum and minimum file sizes in LOC/SLOC per unit of time	
	Mean and median sizes in LOC/SLOC per unit of time	
Files – Committers/ Authors		
	Number of distinct committers/authors per file	The overall number of individuals ever participating in the development of one specific file.
	Maximum and minimum of committs by one committer/author per file	Identifying the committers/authors with the largest and smallest number of commits for a file in the history of the project.
	Mean and median of committers/authors per file	The mean and median number of committers/authors per file in the history of the project.

Metric based on	*Metric Description*	*Socio-Economic Translation*
	Number of files per committers/authors and per file type	How many files a committer/author has been participated in per file type in the history of the project.

3.2 Combined metrics based on Mailing Lists

Metric based on	*Metric Description*	*Socio-Economic Translation*
Messages or posts		
	Total number of messages	Sum of all messages, an output measure. It can be considered an indicator of (secondary) production, and of community activity. Activity on the mailing list can shed light on the health of the project.
	Number of messages per unit of time	How the activity level changes over time in terms of growth and evolution.
	Aggregated number of messages over time	The total value/overview over activity up to a certain point in time.
	Maximum and minimum number of messages per unit of time	Fluctuations in activity level – this identifies the individual posting the most and the fewest messages.
	Mean and median of messages per unit of time	The mean ad median of messages of the population of posters.
	Maximum and minimum of messages size	The message size could be

Metric based on	*Metric Description*	*Socio-Economic Translation*
		measured in kb, or the number of lines of message, the number of words in the body of the message.
	Mean and median of messages size	
Threads		
	Total number of threads	The larger the number of threads, t he more active the community.
	Size of the thread per unit of time (defining size as the total number of replies to an initial posting)	The faster and more active replies, the more healthy community.
	Aggregated size per thread over time	Reflects the stock of knowledge and information built up by the threads.
	Maximum and minimum size of threads	-
	Mean and median size of threads	-
	Maximum and minimum message thread depth, (defined as the number of replies to a reply, to a reply, etc, to an initial posting.)	Can give an indication for how active the community is, and how in-depth the knowledge and information is, also whether it is a discussion going on or more of a quick Q&A.
	Mean and median of message thread depth	-
Authors		
	Total number of distinct authors	Size of the mailing list community.
	Number of distinct authors per unit of time	Fluctuations/evolution in the size of the community over time.
	Aggregated number of distinct authors over time	The number of authors who have contributed to the stock of knowledge.
	Maximum and minimum number of authors per unit of time	For example finding the month with the largest number of author compared to the one with the fewest number of author.

Metric based on	***Metric Description***	***Socio-Economic Translation***
	Mean and median of authors per unit of time	-
	Number of new authors per unit of time	Reflects recruitment.
	Aggregated number of new authors over time	Evolution of the mailing list community. Generation changes, and allows to also investigate how long one author remains part of the community, and how the entry/exit is.
	Maximum and minimum number of new authors per unit of time	Regarding the inflow of new people, the growth, rather than the stock of members.
	Mean and median of new authors per unit of time	-
Messages – Authors		
	Total number of messages per author	The total contribution/effort by each other in the project life time. Can indicate his knowledge level, as well as the depth of involvement in the community, and the centrality of his position.
	Number of messages per author and per unit of time	-
	Aggregated number of messages per authors over time	-
	Maximum and minimum number of messages per author over all authors	Identify which author has the largest contribution, and thus a central role in terms of knowledge.
	Mean and median of messages per author over all authors	-
	Total of number of replies per author	Activity level, and the number of replies he made to questions posted in the list.

Metric based on	***Metric Description***	***Socio-Economic Translation***
	Number of replies per author and per unit of time	Evolution of author contribution and involvement. Could reflect the extent to which he is involved in knowledge transfer.
	Aggregated number of replies per author over time	The stock of input and knowledge contributed up to a given point.
	Maximum and minimum number of replies per author over all authors	Could indicate the level of discussion generated, and the the knowledge transfer potential.
	Mean and median of replies per author over all authors	-
Authors – Threads		
	Total number of distinct authors per thread	Shows the number of people engaged in the discussion; the more people actively participating, the larger the knowledge spread/learning potential.
	Maximum and minimum number of distinct authors per thread	-
	Mean and median of distinct authors per thread	-
	Number of messages per author and per thread	The depth of involvement in the discussion.
	Maximum and minimum number of messages per author and per thread	-
	Mean and median of messages per author and per thread	-

3.3 Combined metrics based on Tracking Systems

Metric based on	*Metric Description*	*Socio-Economic Translation*
Reports		
	Total number of reports	Gives an indication of health of software and community. Many reports does not necessarily indicate a problem, possibly just that many issues have been discovered and solved.
	Number of reports per unit of time	Evolution of reports may give an indication of quality of the software, also how active the community is, and how fast issues are being resolved.
	Aggregated number of reports over time	The stock of issues having been addressed.
	Maximum and minimum number of reports per unit of time	-
	Mean and median of reports per unit of time	-
	Total number of reports per category	Indicates which category is most active and important.
	Number of reports per category and per unit of time	How activity levels evolve over time, the dynamics of the process.
	Aggregated number of reports per category over time	Stock of aggregated knowledge, total effort spent.
	Maximum and minimum number of reports per category and per unit of time	-
	Mean and median of reports per category and per unit of time	-
	Number of reports per category and per unit of time	The evolution/growth/change of importance of the different topics.

Metric based on	*Metric Description*	*Socio-Economic Translation*
	Aggregated number of reports per category over time	-
	Maximum and minimum number of reports per category and per unit of time	-
	Mean and median of reports per category and per unit of time	-
	Number of reports per status and per unit of time	-
	Aggregated number of reports per status over time	-
	Maximum and minimum number of reports per status and per unit of time	-
	Mean and median of reports per status and per unit of time	-
	Maximum, minimum, mean and median defect resolution time, defined as the time difference between the date a bug was opened and the date it was closed (or confirmed).[7]	-
Contributors		
	Total number of contributors	The size of the community active in the tracking system.
	Number of contributors per unit of time	Evolution of contributors, community health.
	Aggregated number of contributors over time	Stock of knowledge potential.
	Maximum and minimum number of contributors per unit of time	-
	Mean and median of contributors per unit of time	-
	Number of new contributors per unit of time	Inflow/growth of new members.
	Aggregated number of new contributors over time	-
	Maximum and minimum number of new contributors per unit of time	-
	Mean and median of new contributors per unit of	-

[7] Several alternative definitions are possible here. It should also be noted that bugs that were opened and closed and then re-opened should be treated separately.

Metric based on	***Metric Description***	***Socio-Economic Translation***
	time	
Reports – Contributors		
	Number of reports per contributor per unit of time	Activity and effort by each contributor, and the evolution of this. He may go in and out of the core team for example. Life-cycle of involvement.
	Aggregated number of reports per contributor over time	The depth of involvement for each contributor.
	Maximum and minimum number of reports per contributor and per unit of time	Changes in activity levels for community members.
	Mean and median of reports per contributor and per unit of time	-
	Number of reports per contributor, status and unit of time	-
	Aggregated number of reports per contributor, status over time	-
	Maximum and minimum number of reports per contributor, status and unit of time	-
	Mean and median of reports per contributor, status and unit of time	-

4. Complex Variables

Complex variables are third layer variables, and more complex computation is required to get to obtain them, as well as frequently information from multiple repositories. The variables and indicators below draws on feedback from the QUALOSS project, where the indicators were identified to measure quality. It also includes variables and indicators from the high level studies from WP5 and WP11 of FLOSSMETRICS.

Metric	***Metric Description/Methodology***	***Socio-Economic Translation***
Evolution of first issue (bug) reports submitted by registered users	Measure monthly the first bug submitted by registered people, taking into account the slope of the resultant line (y=mx+b) while measuring the aggregated number and periods of one year.	Retrieving the date of the first bug for each member of the community, we are able to know if the number of new members reporting bugs remains stable.
Evolution of first commits submitted by registered users	First commit of each detected committer in the SCM is retrieved and its monthly rate calculated, taking into account the slope of the resultant line (y=mx+b) while measuring the aggregated number and periods of one year:	Retrieving the date of the first commit for each member of the community, it is possible to determine whether or not the number of new members committing remains stable
Evolution of new core members	Checking who forms the core team of developers (those with the 80% of the commits) and then analysing the first commit and monthly results of each new member who starts working on the core group, we can get the regeneration of core developers. Note: the results are obtained taking into account the slope of the resultant line (y=mx+b) while measuring the aggregated number and periods of one year:	Retrieving this information gives an estimator of how the core of contributors is evolving. Thus, it can be seen whether there is a natural regeneration of core developers.
Evolution of core members leaving the	First who forms the core team of developers (those with the 80% of the commits) is	Taking into account this variable we can estimate if there is a dramatic

Metric	***Metric Description/Methodology***	***Socio-Economic Translation***
core team	calculaded. After this, any number of people who disappear, does not commit in the next months, from this core team is counted as one.	decrease in the number of core developers, and so, a risk in the regeneration.
Evolution of the balance in the core team	Calculate the evolution of new core members and the evolution of member that leaves the core. Then, calculate the difference between these two variables.	Number of people who left the core team minus number of new members of the core team monthly.
Average of committers longevity	Measure the number of months in which a committer appears. This is her age in the project. The average of all of them is the longevity of a committer in the community.	Average age of people working on a project. This metric is focused on the average of years worked by each developer. With this approximation, we are able to know if members are approaching this limit and we can estimate future effort needs.
Evolution of code contributors who submitted patches and changes	Count the number of people commiting changes in a major release, Taking into account the slope of the resultant line (y=mx+b) while measuring the aggregated number and periods of one year:	Evolution of people who contribute to the source code and reporting bugs. A way to retrieve this data is to analyse those commiters and reporters with the same nickname.
Total code contributors who submitted patches and changes	Same metric than above but this is the sum of all of them and not the evolution.	Number of people involved in the project being a committer and also reporting bugs. With this variable we can measure the size of a community.
Evolution in number of events	Total number of commits, posts and bug reports. The indicator will be measured taking into account the slope of the resultant line (y=mx+b) while measuring the aggregated number and periods of one year.	An event is defined as any kind of activity measurable from a community. Generally speaking, posts, commits or bug reports. Monthly analysis will provide a general view of the project and its tendency.
Evolution in number	Total number of commits per month taking into	Monthly analysis will provide a

Metric	***Metric Description/Methodology***	***Socio-Economic Translation***
of commits	account the slope of the resultant line (y=mx+b) while measuring the aggregated number and periods of one year.	general view of the project. In this way an increase or decrease in the number of commits will show the tendency of the community.
Percentage of people working in old releases	Number of people working on old releases divided by the total amount of people active people working nowadays on the project.	Number of people working on old releases out of total work on the project. We can determine how supported are the old releases for maintenance purposes.
Territoriality	Percentage of files worked just for one committer. If this number if really high, it means that there are no big sub-groups of people.	This metric shows the territoriality in a project. Generally speaking, most of the files are touched or handled by just one committer. It means that high levels of orphaning may be seen as a risk situation. If a developer leaves the project, her knowledge will disappear and all her files are totally unknown by the rest of the developers team.
Activity per committer	Measured counting the number of people working on the project, out of number of people working on the whole project and taking into account the whole set of activities to carry on.	High number of SLOC, e-mails or bugs to be fixed per active developer may mean that they are overworked. In this case, the community is clearly busy and they need more people to help on it.
Number of lines per committer	Number of lines per committer, taking into account the number of active committer, and the total size in SLOC of the project.	Relationship between committers and total number of lines or files. With this absolute number, we are able to check the number of lines per committer. Thus, just regarding to the source code, we can say if they need more resources on it.