Sponsored through Framework Programme Sixth (Call 5) by the European Commission

Document Information

Version: 1.1
Date : Jan 21, 10
Pages : 27
revision: 355

Owning Partner:
FUNDP

Author(s):
Flora Kamseu (FUNDP),
Raimundas Matulevičius (FUNDP),
Daniel Izquierdo (URJC),
Naji Habra (FUNDP)

Reviewer(s):
Martin Soto (IESE)
Daniel Izquierdo (URJC)

To:
European Commission

Purpose of distribution:

Submission of deliverable D4.4 to the European Commission

The QUALOSS Consortium consists of: CETIC (BE), Facultés Notre Dame de la Paix à Namur (BE), Universidad Rey Juan Carlos (ES), Fraunhofer IESE (DE), ZEA Partners (BE), MERIT (NL), AdaCore (FR), PEPITe (BE)

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

Deliverable ID: D4.4

Title:

Official Title Validation of Quality Models and User Manual

New Title: Validating the standard QualOSS assessment method

Deliverable: D4.4

Title: Validating the Standard QualOSS Assessment Method

Executive Summary:

Standard QualOSS Assessment Method provides a systematic way to assess the FlOSS endeavour along different dimensions of quality represented in a quality model tree. In this deliverable, we investigate the validity of the Standard QualOSS Assessment Method. Together QualOSS partners performed a number of different validity tests on the different versions of the standard QualOSS assessment method, in particular, its initial version and then version 1.0. The two versions were applied on case study assessments and on various test and benchmark assessments of different FlOSS endeavours. This validation effort resulted in a number of suggestions to improve version 1.0 of the Standard QualOSS assessment method. The improvements are suggested for different method artefacts, namely (*i*) the FlOSS assessment process, (*ii*) quality model, (*iii*) metrics and indicators, and (*iv*) documentation. Some of these recommendations will be implemented in version 1.1 of the standard QualOSS assessment method to be delivered in D4.5. Others are left for future work when developing new method versions. At the end, we conclude that our findings contribute positively to the validity of the the Standard QualOSS Assessment Method based on a number of performance and correctness test.

CHANGE LOG

Ver.	Date	Author	Description
v0.1	02/11/09	Prepared: Flora Kamseu (FUNDP), Raimundas Matulevičius (FUNDP) Reviewed: Naji Habra (FUNDP)	A draft prepared and submitted for the FUNDP internal review. Review is given back.
v0.2	03/11/09	Flora Kamseu (FUNDP), Raimundas Matulevičius (FUNDP), Daniel Izquierdo (URJC), Naji Habra (FUNDP)	A draft submitted for the QualOSS review.
V0.3	12/11/09	Martin Soto (IESE)	Reviewed of the first draft
V0.4	19/11/09	Flora Kamseu (FUNDP)	Implements comments from the first review
V0.5	28/11/09	Raimundas Matulevičius (FUNDP),	Implementation of some comments from the second review

APPLICABLE DOCUMENT LIST

Ref.	Title, author, source, date, status	Deliverable Identification
D4.1	Deprez J.-C., Haaland K., Kamseu F., QualOSS Methodology & QualOSS Assessment Method, Deliverable D4.1 of the QualOSS project, 2008, Final	D4.1
D4.2	Soto M., Ciolkowski M., Deprez J.-C., Ruiz J., Herraiz I., Campos C. G., Matulevičius R., Metrics and Indicators of the Standard QualOSS Assessment Method, Deliverable 4.2 of the QualOSS project, 2009, Submitted	D4.2
D5.1	Ruiz J., Ghosh R., Glott R., Haaland K., Deprez J.-C., Flamand J., Case Study Design and Pilot Projects, Deliverable D5.1 of the QualOSS project, 2009 Submitted	D.5.1
D3.2	Cortázar D. I., Measurement Tools Deployment and Data Collection, Deliverable D3.2 of the QualOSS project, 2009, in progress	D3.2
D4.3	Auvray V., Deliverable D4.3 of the QualOSS project, 2009, in progress	D.4.3

TABLE OF CONTENTS

1. Introduction

The strategic objective of the QualOSS project is to enhance the competitive position of the European software industry by providing a method and tools for improving productivity and the quality of Open Source Software (OSS) products. The QualOSS project is subdivided in six work packages. The Work Package 4 (WP4) is the core of the QualOSS project. It builds the complete version of the QualOSS method and specifies how to apply it. WP4 is divided in 5 tasks.

Task 4.1 has already defined the advanced QualOSS quality model [1]. Task 4.2 also calibrates this model by testing it on actual F/OSS endeavours [2]. Task 4.3 is more of an explanatory nature and attempts to apply various artificial intelligence and machine learning techniques to discover new indicators. Task 4.4 applies the validation strategy proposed during Task 4.1 to the FlOSS endeavours selected and processed during WP3. Task 4.5 is a short task that updates the QualOSS platform. In particular, it implements the new indicators discovered and validated during WP4 which were not yet implemented during WP2.

1.1 Scope

The scope of this document is to present the validation activities performed for the Standard QualOSS assessment method. In particular, this document focusses on the *initial* version and version 1.0 produced in the QualOSS project.

1.2 Purpose and target

The main purpose of this work is to make the validation of the Standard QualOSS assessment method [1, 2]. This goal is subdivided into four sub-goals:
•SG.1: Validate the quality model [1] of the Standard QualOSS assessment method;
•SG.2: Validate the assessment process [1] of the Standard QualOSS assessment method;
•SG.3: Validate metrics and indicators [2] of the Standard QualOSS assessment method;
•SG.4: Validate documentation [1, 2] of the Standard QualOSS assessment method.

The target of SG.1 is the quality model developed in the initial version and advanced in version 1.0. Similarly is about the SG.2 target, which is the validation process developed and advanced in the QualOSS standard assessment method in the initial and 1.0 versions. In version 1.0 metrics and indicators were defined. These are the target of the SG.3. Finally target of SG.4 is the documentation that guides the application of the Standard QualOSS assessment method to assess the FlOSS endeavors.

1.3 Validation Method and Contributions

A systematic method used to validate the Standard QualOSS assessment method is presented in Figure 1. It consists of the following activities performed in different Work Packages of the QualOSS project:

• **Definition and Review.** This activity involves defining the initial version of the Standard QualOSS assessment method. It was executed as Task 4.2 in WP4. The major steps include definition, review and update of metrics, and definition, review and update of the indicators, The major *validity contribution* is

received from the *review* steps which contributed to the revision of the metrics and indicators by both the authors and reviewers of the metrics and indicators.

• **Assessment of *k3b* and *Findbugs***. This activity involves defining the initial version of the Standard QualOSS assessment method. It was executed as Task 4.4 in WP4. ***Findbugs*** and ***k3b*** are the two FlOSS endeavors that were used to validate the initial version of the QualOSS standard assessment method. These validated the method dimension values like {*Full FlOSS collaboration, Entire FlOSS Project, Product Comparison, Integration in Product}.*

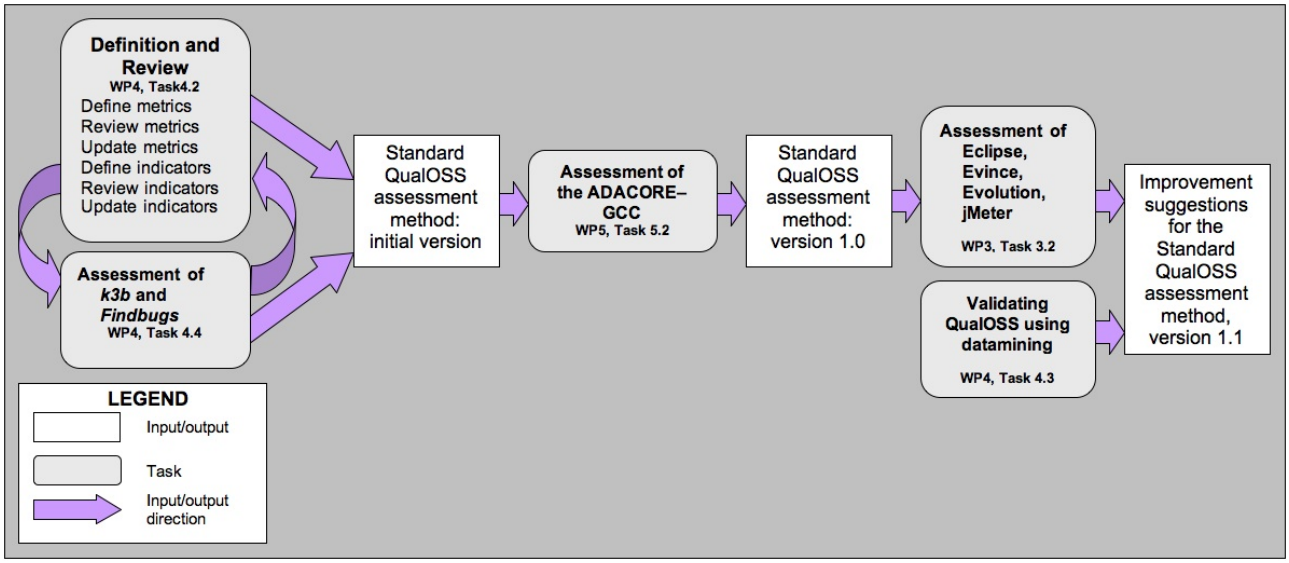

Figure 1: Validation method for the Standard QualOSS assessment method

• **Assessment of the AdaCore GCC**. This activity involves validating the first stable/initial version of the Standard QualOSS assessment method and contributes to the definition of version 1.0. This case study involves evaluator's perceptual understanding of the Standard QualOSS assessment method. The AdaCore GCC back-end case study explored different dimension values: like {*full FlOSS collaboration, part of a FlOSS project, version comparison,* and *integration in product}.*

• **Assessment of Eclipse, Evince, Evolution and jMeter**. This activity involves extensive application of the Standard QualOSS assessment method to assess quality of different FlOSS endeavors. For the version 1.0 validation, four projects were selected: Eclipse, Evince, Evolution and jMeter. These projects were used as (*i*) the benchmarking projects (e.g., ***Evolution***, ***Evince*** and ***jMeter***) to calibrate the performance of the metrics and indicators (this addressed QualOSS dimension values, like {*full FlOSS collaboration, entire FlOSS project, product comparison,* and *integration in product*}), and as (*ii*) the feasibility test to investigate standard QualOSS assessment (v1.0) dimensions of {*full FlOSS collaboration, set of FlOSS projects, product comparison, and integration in product*}. In the later case the ***Eclipse*** project was investigated. In both cases the method resulted in the assessment reports on the findings of the project quality. In the rest of this work we will consider these results as an input to produce the improvement recommendations for the v1.1 of the standard QualOSS assessment method.

• **Validating QualOSS using datamining**. This activity involves definition of the metrics and indicators of the Standard QualOSS assessment method using datamining techniques. This contributes to the validity of the metrics and indicators since this allows for comparing two different metric and indicator definitions (manual used in Task 4.2 and datamining applied in Task 4.3).

The validation activities described in this deliverable contribute, together with the set of guidelines and improvement suggestions, to produce version 1.1 of the Standard QualOSS assessment method.

1.4 Structure

The structure of the document is as follow: Section 2 describes how the version 1.0 of the standard QualOSS assessment method was produced and validated. This section includes validation activities such as: review of the QualOSS of the metrics and indicators, application of the quality model, metrics and indicators at the AdaCore GCC back-end case study. In Section 3 we discuss validation of the Standard QualOSS assessment method used to assess different FlOSS endeavours. Section 4 described validation of the QualOSS assessment method through the datamining. Finally we conclude this deliverable in Section 5, where we present a list of improvements to produce version 1.1 of the Standard QualOSS assessment method.

2.QualOSS v.1.0 Development

In this section we will describe how version 1.0 of the Standard QualOSS assessment was developed. In Subsection 2.1, we will illustrate the first attempts to validate the standard QualOSS assessment method through the review activities and the assessment of two open source projects: ***k3b*** and ***Findbugs***. Next in Subsection 2.2, we will describe how the QualOSS standard assessment method was applied in a case study to measure the AdaCore ***GCC*** project. This case study resulted in a list of suggestions to improve the initial version. Implementation of these improvements resulted in version 1.0 of the standard QualOSS assessment method. As mentioned in Section 1 these projectes were used to validated the method dimension values like {*Full FlOSS collaboration, Entire FlOSS Project, Product Comparison, Integration in Product}.*

2.1 Review and Application [1]

This section describes validation activities done when defining the initial version of the Quality model, that are also interrelated with the method development operations. Section 2.1.1. focuses on the *review* operation performed during the metric and indicator definition. Section 2.1.2 emphasizes the first application of the quality model to evaluate the ***k3b*** and ***Findbugs*** projects. As shown in Figure 2, these two activities are interrelated. They both were executed in parallel and they resulted in the *initial* version of the Standard QualOSS assessment method [2].

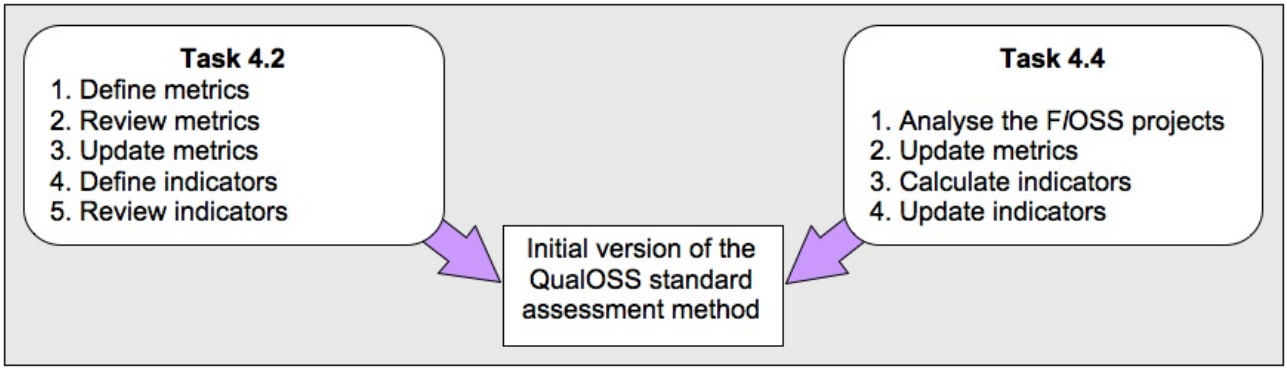

Figure 2: Definition of the initial version of the Standard QualOSS assessment method

2.1.1 Metrics and Indicators definition and review

Quality model. In deliverable D4.1 [1], we have described the quality model. It was created using questionnaires in order to identify the context and the business needs behind a FlOSS assessment. These questionnaires contain different measurement questions that help to identify different viewpoints for a given quality characteristic. More specifically, our questionnaire followed the GQM [3] approach. We have interviewed several stakeholders; these interviews resulted in their business needs where assessments based on the QualOSS methodology could be useful. Our initial proposal is to divide the assessment of robustness and evolvability of the FlOSS endeavour into four quality concerns: work products, community members, software processes and tools and dependencies.

[1]This activity was carried out in WP4, task 4.2

Process. To define metrics and indicators for the quality concerns, we followed the method proposed in Figure 3.

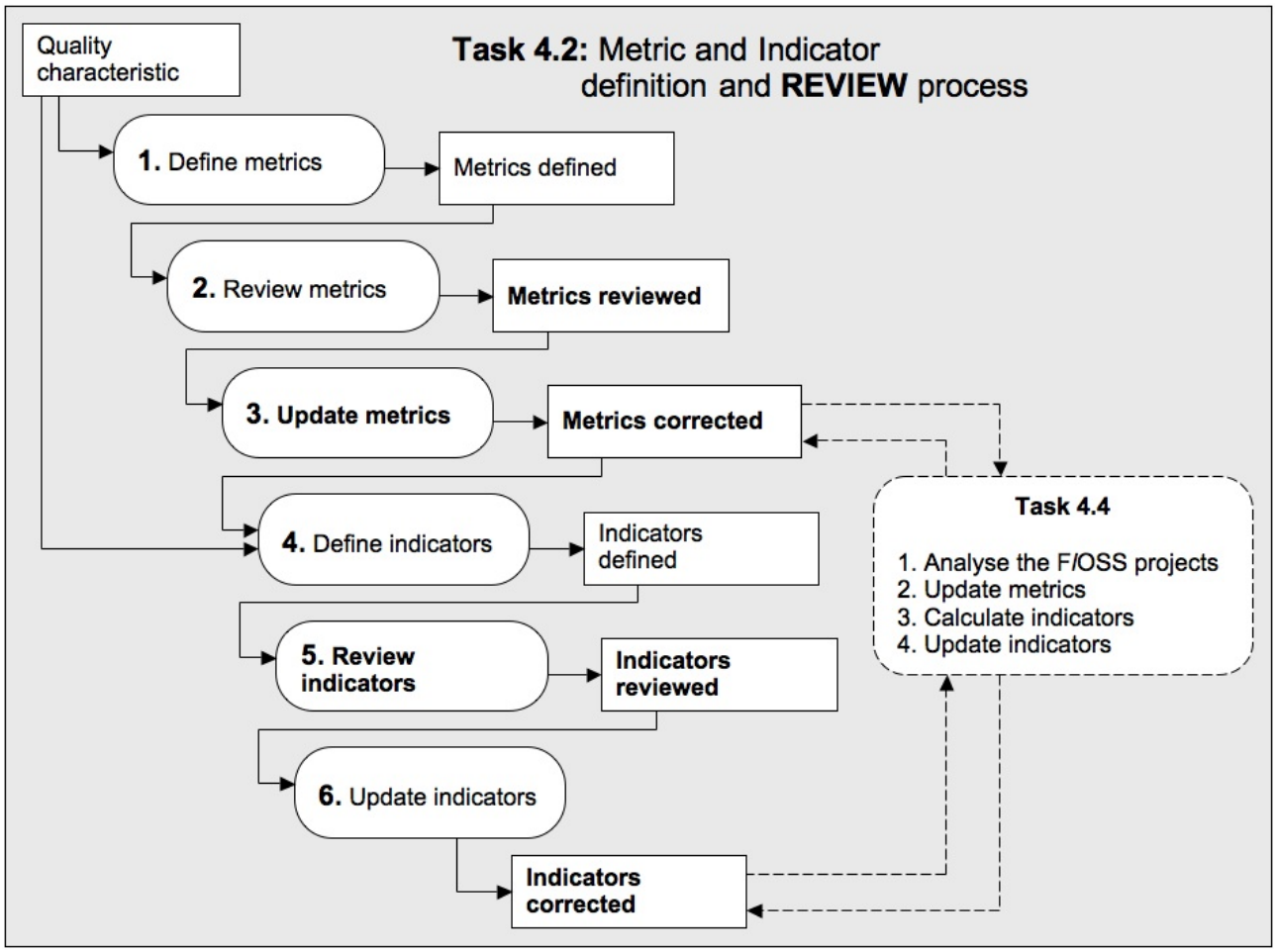

Figure 3: Process to define metrics and indicators

1. *Define metrics.* The input for the first step is the quality concerns/characteristics defined in [1]. This step results in a set of defined metrics for each input characteristic. The metrics were defined by experts from the QualOSS consortium. Metric definition resulted in a template that includes both the documentation on how to take a metric and the metrics values. This definition was submitted to review (step 2).

2. *Review metrics.* In order to review the defined metrics, their definition was exchanged between the QualOSS partners. The reviewers checked for consistency, correctness, and completeness of the metrics. They also studied if the metrics were built on a sound theoretical foundation (units, scales, sound definitions...), and if possible, tried to see if the metrics could be validated on a large enough set of (real-world) data. One review action was to see if the metric definition was in accordance with the provided template. At the end, the reviewers provided comments and suggestions to improve the defined metrics.

3. *Update metrics.* The review comments and suggestions for improvement were addressed to the partner who initially defined metrics during the first step. This step resulted in the metrics being updated for each quality characteristic. The partner who initially defined the metrics provided a new version if necessary of the metric definition.

4. *Define indicators.* This activity has two inputs: the first is the characteristics of the quality model, the second is the metrics set for this quality characteristic. In practice, an indicator is created by the QualOSS partners. The step results is the definition of the indicators for the quality characteristic.

5. *Review indicators.* The inputs for this task are the indicators defined in the previous step. For the review, the indicators were exchanged between the QualOSS partners (similar as for the metric review)). The main outcome of this review is comments and suggestions to improve indicators. The indicators were reviewed for their consistency, correctness and correspondence to the predefined template.

6. Update indicators. The partner who defined indicators addressed the review comments. The step resulted in the updated indicators and improvement suggestions for the quality characteristic.

Result. Our definition and review resulted in the metrics and indicators and their application process documented in the spreadsheet templates. Table 1 presents the numbers of metrics and indicators defined for each characteristic [2]. For some quality characteristics, the definition of metrics and indicators was postponed/cancelled because of the lack of information and complexity of the assessment domain (e.g *Software processes/Capability support and release management* and *Runtime and compile/Tool dependencies/FlOSS endeavors compatibility*).

Table 1: Number of metrics and indicators defined for the quality model

Area	Metrics		Indicators		Effort (hours)
	Manual	Automatic	Manual	Automatic	
Work Product / Product / Reliability		9		4	6
Work Product / Product / Maintainability	22	11		15	4
Work Product / Product / Security		38		9	1
Work Product / Product / Test	40			9	1
Work Product / Documentation	41[(a)]			6	8
Community members	16	7			8
Software process	69			17	6
TOTAL	**147**	**65**	**0**	**60**	**34**

[(a)] A total of 572 individual document checks will be performed in order to calculate these metrics

Threats to Validity. Despite of the fact that the definition and the review of metrics and indicators was planned as a relative short activity, the review took quite a long time. It started in January and ended in May 2009. The reviewers of metrics and indicators are the members of the QualOSS consortium and due to the wide variation in human ability and capabilities, this review can be qualified as subjective. However, we see it as a nice first validation since it helped us to create the initial version of the metrics and indicators for the quality model and documentation.

Because of time limitations and giving the time taken by the first review, the metrics and indicators review was performed only once. It would be interesting to have another peer review that may contribute to the strength of the metrics and indicators definition. On the other hand, we chose to test the metrics and indicators quality model and their documentation during the performance test. These activities are presented in Section 2.1.2 and 3.

2.1.2 Assessment of *k3b* and *Findbugs*

The purpose of this section is to test the *applicability* of the QualOSS assessment method to assess FlOSS endeavours. We have applied the method to assess ***k3b*** and ***Findbugs***.

FIOSS description. ***k3b*** (abbreviation for KDE Burn Baby Burn) is a CD and DVD burning application for Linux systems. ***Findbugs*** is a static analysis tool to find bugs in Java programs. ***Findbugs*** and ***k3b*** are distributed under the terms of the Lesser GNU Public License and are free software. Thus, they are open source software and could be used by us as a first candidate to apply the QualOSS quality model. The main objective of this assessment was to see if we can apply the method [1, 2] to assess software like ***k3b*** and ***Findbugs***, as if we can obtain some reasonable results within a reasonable time.

Process and results. The process method used for assess ***k3b*** and ***Findbugs*** is described in Figure 4.

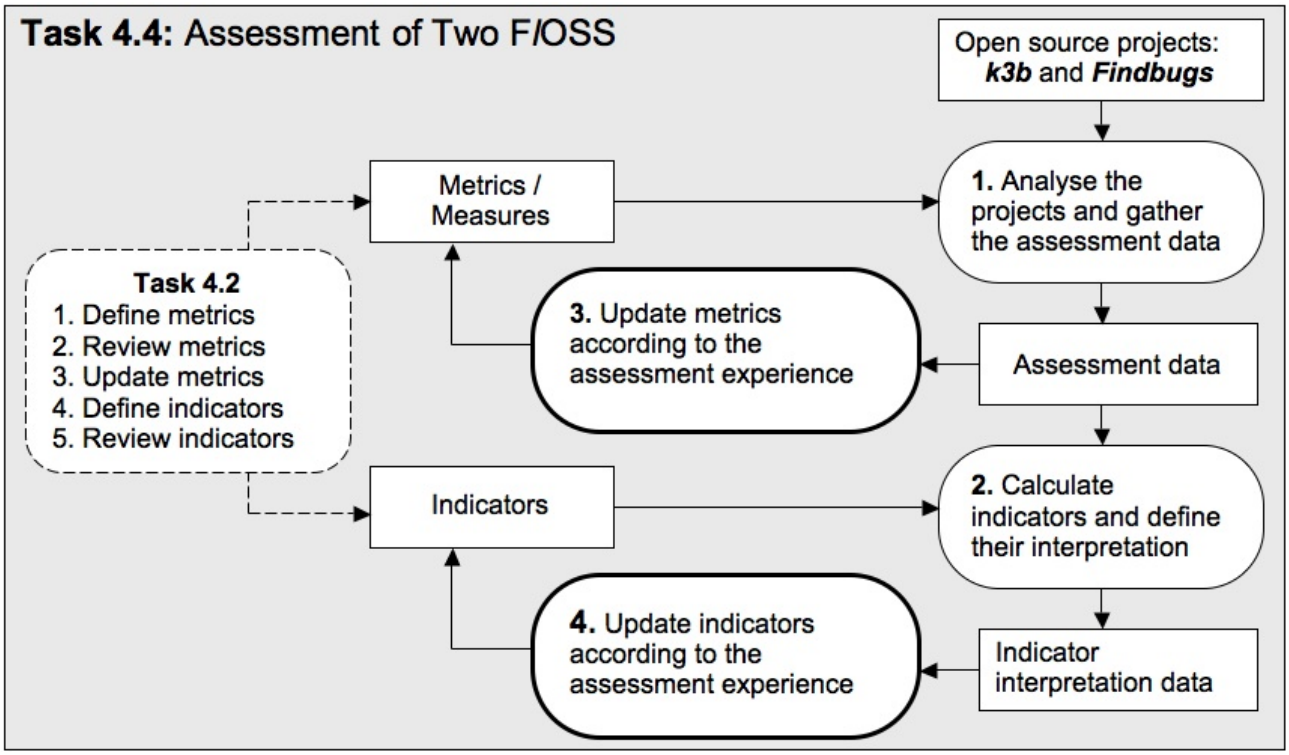

Figure 4: Testing the metrics and indicators in the project assessment

It consisted of the following steps :

1. *Analyse the projects and gather the assessment data*. This step has two inputs. The first input is the ***k3b*** / ***Findbugs*** projects data. The second input is the metrics defined during the metric and indicator definition

process (Figure 3). During this step, the evaluators (the same who defined the metrics and indicators) were gathering data and evaluation experience about ***k3b*** and ***Findbugs*** project according to the metrics. The step results in the assessment data (*i.e.*, metric values) on the ***k3b*** and ***Findbugs*** project and evaluator's experience on the assessment.

2. *Calculate indicators and define their interpretation*. This step also has two inputs. The first input is the ***k3b*** and ***Findbugs*** assessment data. The second input is the indicators defined during the metric and indicator definition process (Figure 1). During this step, the evaluators were calculating indicator values and defining their interpretation (green, yellow, red or black, see [2] for details). The step results in the indicator interpretation models and evaluators experience on the assessment.

3. *Update metrics according to the assessment experience*. This step implements the feedback on the metrics received during the ***k3b*** and ***Findbugs*** assessment. Based on the evaluator's experience and knowledge acquired during the assessment process, the metric definition for each quality characteristic were updated, re-evaluate and/or corrected. This task is one of the validation task since the assessor can redefine or update metrics definition because of some experience knowledge acquired during the measurement process. The output of this step is the updated metrics in the quality model.

4. *Update indicators according to the assessment experience*. This step implements the feedback on the indicators used during the ***k3b*** and ***Findbugs*** measurement. Based on the evaluator's experience and knowledge about indicators acquired during the assessment process, the indicator definition are updated, re-evaluate and corrected. The output is the updated indicators.

Validity threats: This assessment is the first one done by the QualOSS partners. Although the evaluators were the ones who defined metrics and indicators, there might not be easy to apply our approach for the people who were not involved in the method definition process.

Another threat is the number of projects used to apply the method. We applied the method on two FlOSS endeavours This number is not significant enough to talk about the complete validation.

Results from this measurement can be considered as subjective because they depend on the evaluators personal interpretation, experience and knowledge. This results might be influenced by the background of the person who did the measurement.

Another threat is the fact that after applying the model on ***k3b*** and ***Findbugs***, there was no review operation. This mean that the result might contain assessment errors.

Finally, there were some quality characteristics (e.g. security) that were not applied to ***k3b*** and ***Findbugs*** [2].

The above issues indicate the need to continue validation of the standard QualOSS assessment method. The most important aspect of assessing ***k3b*** and ***Findbugs*** was to show that we can perform the FlOSS endeavours assessment using our model. We succeeded to obtain the results for the ***k3b*** and ***Findbugs***. In this respect, our validation reached its initial objectives. The lessons learnt weree implemented in the standard QualOSS assessment method. In the next subsection, we will describe how the standard QualOSS assessment method was applied in the AdaCore GCC.

2.2 Applying QualOSS in Case Study[2]

In order to have some human perception on the standard QualOSS assessment method [1] we have applied this method to a concrete case study. The case study is called AdaCore GCC back-end. As mentioned in

[2] This was the activity of WP5.

Section 1, the AdaCore GCC back-end case study explored different QualOSS dimension values:, like {*full FlOSS collaboration, part of a FlOSS project, version comparison,* and *integration in product}.*

Case study description. AdaCore is a company that provides commercial, open software solutions for Ada to build robust and reliable software. AdaCore needs to use the most recent version of the GCC back-end in order to take advantage of recent fixes and improvements, and new targets made available in each new release. The main goal of this case study is to use the QualOSS method to get information about the GCC back-end status in terms of robustness. The version that will be the input of this case study is version 4.3 of the GCC back-end to assess whether it is suitable or not in terms of robustness. On the other hand, AdaCore skipped version 4.2 because it did not provide many new features and it was considered not robust enough. Therefore, there are several reasons to do the AdaCore case study [4]:

•AdaCore needs to assess whether the application of the QualOSS method could give some indication of GCC back-end version 4.2 about the lack of new features and robustness

•The AdaCore Quality Assurance team could know in advance if a new release may be problematic

•AdaCore's customers could be interested in the quality of the back-end code generator

•The GCC community could be interested in the feedback about the evolution of GCC in terms of robustness.

In this deliverable we are interested mainly in the application of the standard assessment method. Other goals of the case study are reported in [4].

Case study design. The person who did the assessment case study is an expert in AdaCore and a partner in the QualOSS project. The design of the case study is provided in Figure 5. It consists of two steps. The first step is *Application of the QualOSS standard assessment method (called QualOSS in this section) to assess the AdaCore GCC back-end.* We have applied the version of QualOSS which resulted from the Task 4.2 (see description in Section 2.1). The evaluator investigated the AdaCore GCC back-end and filled in the templates that provide guidance for the assessment of different QualOSS quality characteristics. The step outcome is twofold. Firstly, it includes the assessment results, reported in the deliverable D5.1 [4]. Secondly it provides the list of improvements for the QualOSS standard model. These improvements are the evaluations perceptions based on the QualOSS application. The second step is *Implementation of the QualOSS improvements*. The input for this step is the list of improvements resulting from the first step. These improvements were submitted to the project partners who defined the quality characteristics, metrics and indicators. The implementation of these improvements resulted in the Standard QualOSS assessment method, version 1.0, which is described in detail in deliverable D4.2.

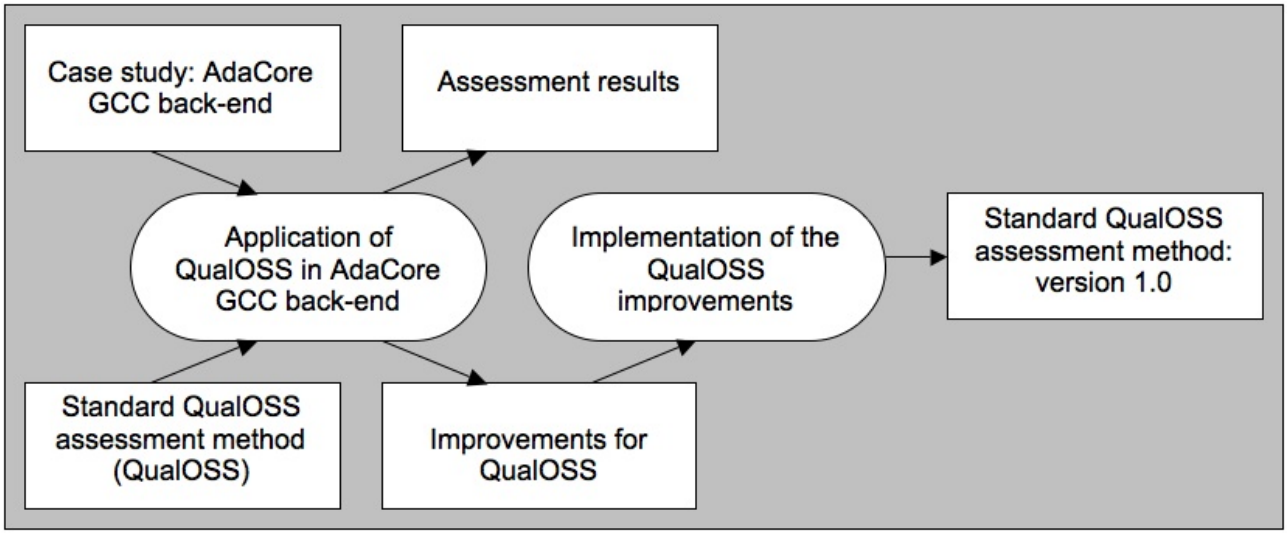

Figure 5: Case study design

Improvements: The assessor said that it was not very difficult to perform the assessment. But he also suggested a number of improvements for the standard QualOSS assessment method.

In general the assessor should have an introduction for each of the quality characteristics describing how metrics and indicators are computed and where the results must be filled in the template. Basically, it requires the definition of the concrete cells in the characteristic templates that need to be filled, also describing the input required for them. This introduction should also point to the documents where to find the detailed definition of the metrics and indicators.

The templates should be common for all quality characteristics. Having a common template would make everything much faster because in the current situation each time you need to study the different sheets (layout, information to be filled, etc.). It would also be very helpful to have the indicators computed automatically from the metrics. It would avoid the need to go to the documentation to search for the place where this is defined, and it would also ensure that indicators are derived from metrics in a consistent way across different assessments.

Other feedback and improvements are specific to the individual quality characteristics, their metrics and indicators:

- ***Work products - products – maintainability*:**
  - It seems that the metric *its__issue__number_of_corrected_issues* is not used for any indicator. The assessor thinks that it is an interesting metric that should be used (for an indicator such as percentage of issues corrected).
  - There are not many metrics that can be measured for C code. Not many tools work well with C code contrary to Java code.
- ***Work products - products – reliability*:**
  - The definitions are not clear enough. Which is the document that describe them?
  - How to compute reliability indicators from metrics? It is not possible to find the document that defines it.
  - Indicator *Importance_of_corrections* should use percentage of files instead of absolute number of files. 3 files out of 5 is a bigger change than 20 out of 5000.
- ***Work products - documentation availability***
  - It would be good to be able to assign weights for the different metrics and documents, and describe how to do it in the user manual.
  - In the case of an assessor who might be more interested in documentation availability completeness than documentation availability organization, he would like to be able to assign weights to reflect this fact.
- ***Work products - tests***
  - The person who did case study measurement liked the detailed description of the process to take measurements.
- ***Community members***
  - It was not clear how to derive indicators from metrics. The assessor did not find where that was described.
- ***Software processes***
  - The person who did case study measurement liked the automatic computation of indicators from metrics.

The person who did the measurement of the AdaCore case study was not involved enough in the metrics and indicators definition. So results could not be influence by the way metrics and indicators were defined. Results could be influenced by his background, knowledge and experience.

Based on the above feedback, people who had defined the metrics and indicators, had corrected and updated the metrics and indicators definition; they had also unified the format of the template for each quality characteristic. Implementation of the feedback received from the AdaCore GCC back-end case study resulted in the Standard QualOSS assessment method, version 1.0 [2], briefly presented in Section 2.3.

2.3 Results and Threats to Validity

Results: Activities described in section 2.1 and 2.2 resulted in the **version 1.0**, of the Standard QualOSS assessment method. This version is organised as the spreadsheet template [2] for each quality characteristic. The template contains the following tabs: *Measures*, *DefinitionOfConcepts*, *Indicators_CompleteInfo* and *predefined_data*. The *Measures* and *Indicators_CompleteInfo* tabs describe how to calculate metrics and indicator for a FlOSS endeavour The *DefinitionOfConcepts sheet* is the one that defines the different concepts used in the methodology that describe how to perform the measurement of metrics and indicators and provides definitions of some important terms. This sheet helps the assessor who faces difficulties when assessing a FlOSS endeavour characteristic to find the details about how to proceed. The *predefined_data* is the one that lists all necessary data to calculate metrics and indicators measurement.

The tab *Measures* contains the structure that supports gathering of the metrics taken on the FlOSS project during assessment. Structure of this table is provided in Table 2.a. The tab *Indicators_CompleteInfo* contains the structure that supports calculating of the indicators taken on the FlOSS project during assessment. Structure of this tab is provided in Table 2.b.

Table 2: Structure of Measures and Indicators_CompleteInfo Tabs
Standard QualOSS assessment method, version 1.0

(a) *Measures* tab		(b) *Indicators_CompleteInfo*	
Column name	**Description**	**Column name**	**Description**
Question ID	Question ID from [1]	Characteristic	QualOSS model characteristic
Question	Questions related to the characteristic in [1]	Assessment goal related to the characteristics	The main goal related to this characteristic
Responsible Person	The assessor of the metric	Viewpoint	viewpoint as describe in [1]
Concept used by measure or measurement procedure	Specification of the measurement procedure to obtain the metric value	Rationale for indicator at the role level	Explanation of the indicator's purpose
Measure name	Name of the metric	Indicator name	Name of the indicator
Data source type	Specification of the datasources, data repository(ies) used as a source to calculate metric	Answer to questions	Different answers to questions related to the characteristic
Artifact type	Type of artifact used for the metric. Depending on the metric, only one of these two fields (Data source type and Artifact type) could be relevant	Measures used	Metrics used to compute this indicator
Measurement procedure for measure	The actual procedure used to obtain the metric's value. For an automated tool, we mention the tool and the version	Description/Rationale for indicator	Description of the different level of indicator
Measure scale	Measurement scale	Rule (based on measures)	Mathematical rule or procedure used to calculate the indicator's value
Measure value	Value of the metric	Value:	Indicator result

Comment	Comments that may clarify the measure	Colour	Indicator colour (that can be green, Yellow, red or black (see [2])

Threats to Validity. Although we showed that the method is usable and can help assess the FlOSS endeavours, the version 1.0 of the Standard QualOSS assessment method needs further validation.

We need to validate the standard QualOSS assessment method for many reasons:
•The current validation can also be seen as a future-proof quality check: the Standard QualOSS assessment will be maintained in the future. So, we need to go further with the applicability analyses.
•The validation of the Standard QualOSS assessment method and particularly the documentation part would help to have an easy maintenance.
•The validation activities are a sign of professionalism: they need to be more understandable and robust.

The validity of the version 1.0 of the Standard QualOSS assessment method is further investigated in WP3, Task 3.2. These validity activities are described in Section 3.

2.4 Other Case Studies

The AdaCore case study is not the only case study executed using the Standard QualOSS assessment method. Other case studies, like Oce_Yanolc (used to validate dimension values {*FlOSS fork, part of a FlOSS project, product comparison, and integration in product*}) and Freecode_Asterisk (used to validate dimension values {*full FlOSS collaboration, entire FlOSS project, product Comparison,* and *integration in service*}) [4] were still undergoing when this deliverable was in the writing mode. However to report on these other case study results and their contribution towards the validity of the standard QualOSS assessment method was too late with respect to the timing of this deliverable.

3. VALIDATING QUALOSS V.1.0

The standard QualOSS assessment method (version 1.0) consists of the assessment process [1], quality model [D4.1], metrics and indicators [2] and documentation provided together with metrics and indicators in the spreadsheet templates [2]. In this section we will describe how the standard QualOSS assessment method (version 1.0) was validated through the performance test when applying it to assess quality of four FlOSS projects. This application resulted in a number of improvements and suggestions to develop version 1.1 of the standard QualOSS assessment method. These improvement suggestions will be described in Section 5.

3.1 STUDY DESIGN[3]

Purpose. The main purpose of this section is to present the different activities related to the validation of the standard QualOSS assessment method, version 1.0. We would like to answer the following questions: How could the QualOSS standard assessment method be applied? Is it possible to apply this method to assess FlOSS of a different domain? Is the method well documented to help and support the assessment? How should the QualOSS standard assessment method be improved based on the feedback received from our performance test?

To answer these questions, four open source software were selected for assessment using the QualOSS standard assessment method. These projects are Eclipse, Evolution, Evince and JMeter. The study design is presented in Figure 6. This study is carried out as the task 3.2 in WP3 [5] (there are more specified in WP3.2).

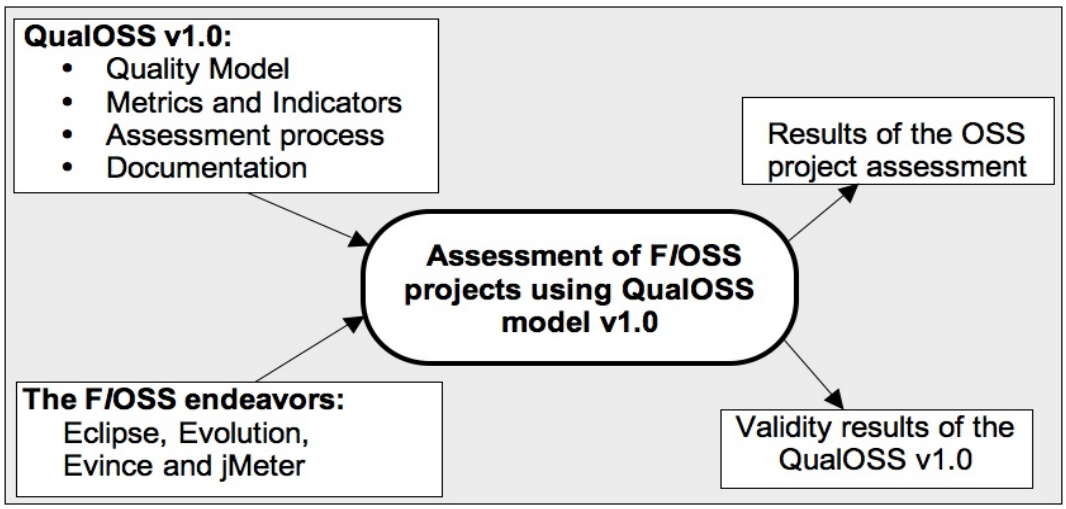

Figure 6: Assessment of the FlOSS

Inputs of the project assessment. For each of four cases, we have two inputs. The first is a FlOSS endeavour under assessment. The selection of these FlOSS endeavors included two purposes. The first is the benchmarking (also dimension values {*full FlOSS collaboration, entire FlOSS project, product comparison,* and *integration in product*}) of the indicators for the standard QualOSS assessment method. For this purpose the ***Evolution***, ***Evince***, and ***jMeter*** projects are selected. The second one includes the

[3]This study is carried out as the task 3.2 in WP3.

feasibility test over the QualOSS dimension values like {*full FlOSS collaboration, set of FlOSS projects, product comparison, and integration in product*}. This is fulfilled by selecting the Eclipse project, that is seen as an umbrella project of many sub-projects.

• ***Eclipse Platform*** is a multi-language software development environment. It supports an open source community, whose projects are focused on building an open development platform comprised of extensible frameworks, tools and runtimes for building, deploying and managing software across the life cycle. comprising an IDE and a plug-in system to extend it More detail about the Eclipse assessment can be found in the Eclipse Analysis report [6].

• ***Evolution*** is an open source software, it is the official personal information manager and workgroup information management tool for GNOME. It combines e-mail, calendar, address book and task list management functions. It assessment report can be found in [7] and resume the detail tasks for this assessment

• ***Evince*** is a document viewer for multiple document formats. It currently supports pdf, postscript, djvu, tiff and dvi. The goal of evince is to replace the multiple document viewers that exist on the GNOME Desktop with a single simple application. See [8] for the assessment details.

• ***JMeter*** is open source software, a Java desktop application designed to load test functional behaviour and measure performance. It was originally designed for testing Web Applications but has since expanded to other test functions in [9]. The assessment process and results of this evaluation are provided.

The second input into the assessment operation is the standard QualOSS assessment method, *version 1.0*, that includes the quality model, metrics and indicators, the FlOSS assessment process and documentation:

•The ***quality model*** is the one described in D4.1 [1] and it contains different groups of characteristics related to a FlOSS endeavour that should be assessed. These are the work product characteristics (reliability, test, security, maintainability and documentation), the community members characteristics, the software process characteristics, and tool and dependency characteristics. From these characteristics, many questions were derived and the evaluation will consist to define metrics and indicators that will answers to these questions from different point of view.

•The FlOSS endeavours ***assessment process*** consists of 15 operations described in [1]. These operations describe a systematic guidance what needs to be done to assess the FlOSS endeavour

•Thus, ***metrics and indicators*** are defined and applied by the QualOSS partners. Metrics and indicators definition are done through a process that validate this definition. They are used on that open source software projects; that shows us it is possible to apply the QualOSS model on OSS. To perform these assessments, documentation related to the metrics and indicator definition are also used as input to perform the evaluation.

•The documentation is one of an important input used to assess these four FlOSS endeavours In some case, this assessment was done by the partner who defined metrics and indicator (Eclipse, Evolution and Evince); it was not the case for JMeter where the assessment was achieved by a partner different from the one who defined metrics and indicators. Thus, it is very important to understand what are the process, the artefacts, or the data that will be useful for this assessment

Assessment of FlOSS projects using QualOSS method v1.0 is the evaluation operation itself of each of these four open source software. From these assessment, different results are obtained and are reported in different files (each file for each project).

Results of the OSS project assessment are presented in the reports [6, 7, 8, 9] for each FlOSS endeavour These results show different aspects useful for having feedback from assessment and knowing if some assessment errors occurred. It shows the evaluation process and how the assessment itself was perform. It also contains details of inputs used to perform these FlOSS assessments and the resulting indicators.

In the following subsection we will highlight how the validation of different standard QualOSS assessment method was executed. in particular we will separate the quality model, the FlOSS endeavour assessment process, metrics and indicators, and documentation.

3.2 Validation Activities

Assessment process. The FlOSS endeavour assessment process is described as the work-flow of operations imposed by the middleweight assessment process in [1]. It was executed to assess all four selected FlOSS endeavours For each endeavour we have recorded different notes on the operation execution. These validity notes are taken into account when preparing the list of process improvements, discussed in Section 5.

Quality model. Application of the quality model is the part of the FlOSS assessment process. We have applied this model and recorded all the observations of this application for each evaluated project [6, 7, 8, 9]. The list of improvements to the quality model are given in Section 5.

Metrics and Indicators. One of the major tasks of the version 1.0 validation is the investigation of the performance of the metrics and indicators and their calibration. To do so, the four FlOSS endeavours were selected as the benchmarking projects (there are some more). However, it was not possible to calibrate all indicators according to the benchmarking results, since these results were simple not available. Nevertheless we were able to record metric and indicator performance. The list of improvement suggestions are provided in Section 5.

Documentation. Documentation of version 1.0 of the standard QualOSS assessment method is the part of the spreadsheet template presented in Section 2.3. During this testing it was validated through the human perception on how good and bad the documentation helped to perform the overall assessment operations. The list of improvements for the documentation is provided in Section 5.

3.3 Other FLOSS Assessments

Four FlOSS projects endeavours were not the only ones analysed under Task 3.2. Other analysis include [5]: CVSanalY, JetSpeed, Nautilus (all contributing to the benchmarking), and Apache Httpd 1.3 (used to test dimension values {*full FlOSS collaboration, entire FlOSS project, version comparison,* and *integration in product* }). However to report on these other FlOSS assessments and their contribution towards the validity of the standard QualOSS assessment method was too late with respect to the timing of this deliverable.[4]

[4] The description of these assessments could be found in D3.2 [5]

4. Datamining Input

4.1 Motivation

The standard QualOSS assessment method [2] involves of several quality attributes. Each of those quality attributes is calculated by means of the results of one or more indicators which provide a general overview.

Each indicator could have several colours and corresponding values that give quick glimpse about a quality attribute. Following this idea, community indicators were defined in a first initial phase directly asking community members and personal experience from QualOSS' members. However, that approach seemed to show a lack of confidence in the results since projects with a high performance showed really low results. Thus, there were found some difficulties in the thresholds definition process.

Thanks to to the relationship with the FLOSSMetrics project, which provide a huge quantity of data from hundreds of FlOSS projects, we considered the statistical analysis of that databases in order to create more reliable thresholds.

4.2 Method

As a brief introduction to the methodology followed to create new thresholds, around 1400 projects from the FLOSSMetrics database were retrieved. Specifically, those with data from the source code management systems. Thus, even when there are other metrics which results have to be measured from other data sources, the low quantity of other data sources in the FLOSSMetrics database made statistically impossible to have enough population to create new thresholds in other data sources more than the SCM.

For more technical details, please refer to deliverable D4.3 [10].

4.3 Results

Next there is a comparison between thresholds defined in version 1.0 and version 1.1. If a "-" is found, it means that the thresholds are the same as the previous version.

Table 3: Comparison of the results

Metrics	*V1.0*	*Intermediate*	*V1.1*
cm-sra1	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	-	-
cm-sra2	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	Black:]-∞, -0.3167] Red:]-0.3167, -0.1469] Yellow:]-0.1469, -0.0527] Green:]-0.0527, +∞[	Black:]–∞, –3] Red: [–2, –1] Yellow: [0, 1] Green: [2, +∞]
cm-sra3	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	Black:]-∞, -0.7] Red:]-0.7, -0.2647] Yellow:]-0.2647, -0.1135] Green:]-0.1135, +∞[	Black:]–∞, –3] Red: [–2, –1] Yellow: [0, 1] Green: [2, +∞]
cm-sra4	Black:]-∞, -5] Red:]-5,0[	Black:]–∞, –0.5] Red:]–0.5, –0.2]	Black:]–∞, –2] Red: –1

	Yellow: 0 Green:]0,+∞[	Yellow:]–0.2, –0.0357] Green:]–0.0357, +∞[	Yellow: 0 Green: [1, +∞]
cm-sra5	Black: [3, +∞[Red:]1, 3] Yellow:]0, 1] Green: 0	Black:]0.4, +∞[Red:]0.1, 0.4] Yellow:]0, 0.1] Green:]–∞, 0]	Black: [1, +∞] Red: 0 Yellow: -1 Green:]–∞, –2]
cm-sra6	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	Black:]–∞, –0.7636] Red:]–0.7636, –0.3] Yellow:]–0.3, –0.1429] Green:]–0.1429, +∞[	Black:]–∞, –2] Red: –1 Yellow: 0 Green: [1, +∞]
cm-sra7	Black: [0, 12[Red: [12, 24[Yellow: [24, 36[Green: [36, +∞[	Black: [1, 5.53555] Red:]5.53555, 8.5789] Yellow:]8.5789, 12.25] Green:]12.25, +∞]	-
cm-sra8	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	-	-
cm-sra9	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	Black:]–∞, –0.2364] Red:]–0.2364, –0.0593] Yellow:]–0.0593, 0.0901] Green:]0.0901, +∞[	Black:]–∞, –3] Red: [–2, –1] Yellow: [0, 1] Green: [2, +∞]
cm-iwa1	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	Black:]–∞, –13.4047] Red:]–13.4047, –1.5654] Yellow:]–1.5654, 4.1337] Green:]4.1337, +∞[	Black:]–∞, –39] Red:]–39, –1] Yellow:]–1, 23.7] Green:]23.7, +∞]
cm-iwa2	Black:]-∞, -5] Red:]-5,0[Yellow: 0 Green:]0,+∞[	Black:]–∞, –19.4] Red:]–19.4, –3.6696] Yellow:]–3.6696, 0.8735] Green:]0.8735, +∞[	Black:]–∞, –37.2] Red:]–37.2, –2] Yellow:]–2, 18.5] Green:]18.5, +∞]
cm-iwa3	Black: 0 Red:]0, 0.05[Yellow: [0.05, 0.1[Green: [0.1, 1]	-	-
cm-iwa4	Black: [0.9, 1] Red: [0.7, 0.9[Yellow: [0.5, 0.7[Green: [0, 0.5[	Black:]0.87245, +∞] Red:]0.7124, 0.87245] Yellow:]0.5522, 0.7124] Green: [0, 0.5522]	-
cm-iwa5	Black: [100000, +∞[Red: [50000,100000[Yellow: [30000, 50000[Green: [0, 30000[	Black: [0, 2130.7] ∪]72295, +∞] Red:]2130.7, 5293.5] ∪]46029.1, 72295] ∪]29543.5, 46029.1] Yellow:]5293.5, 9791.7] Green:]9791.7, 29543.5]	-
cm-iwa6	Black: [100000, +∞[Red: [50000,100000[Yellow: [30000, 50000[Green: [0, 30000[	Black: [0, 2130.7] ∪]72295, +∞] Red:]2130.7, 5293.5] ∪]46029.1, 72295] ∪]29543.5, 46029.1] Yellow:]5293.5, 9791.7] Green:]9791.7, 29543.5]	-
cm-iwa7	Black: [500, +∞[Red: [200,500[Yellow: [50,200[	Black: [0, 0.012875] ∪]0.364425, 1] Red:]0.012875, 0.0443] ∪]	-

	Green: [0,50[	0.2555, 0.364425] Yellow:]0.0443, 0.076625] ∪] 0.188075, 0.2555] Green:]0.076625, 0.188075]	

4.4 Threats to Validity

First of all, FLOSSMetrics does not contain the whole population of FlOSS projects, and even more, during the statistical approach, some of the projects had to be removed due to the fact that they were too small in terms of community, too young in terms of age or they did not have any line of source code. Thus, the FLOSSMetrics database is far from being the perfect set of projects to make comparison, but those outliers were filtered in order to improve accuracy.

Also, it necessary to remark the no existence statistical approach for mailing lists or bug tracking systems since FLOSSMetrics database did not provide that datasets and the time to retrieve all of them manually is too high to be done in the QualOSS framework.

Finally, the statistical approach to calculate differences or tendencies in a FlOSS project may not be the best approach, however, the new definition of indicators solves this problem basing these data in just the tendency and not making it dependable from the statistical approach.

5. Improvements and Conclusions

In this deliverable we report on the validation activities applied to the standard QualOSS assessment method [1, 2]. We were investigating different artefacts of this method, including FlOSS assessment process, quality model, metrics and indicators, and documentation. Our work concludes in the list of improvement suggestions, provided in this section. These improvements might be implemented in the later version of the standard QualOSS assessment process. The priority is indicated at the end of each suggestion. The most important improvements (e.g., ***crucial***) will be implemented in the version 1.1 of the standard QualOSS assessment process.

Improvement suggestions for the FlOSS assessment process. These improvements can be classified into two groups. The first group of improvement (e.g., AP1 and AP2) describe how the assessment process support might be improved. The second group (e.g., AP3-8) concerns the concrete operations of the FlOSS assessment process. The list of improvements for the FlOSS assessment process:

- **AP1: Log-files**. The deviation/log file should be a part of the assessment process. The file should have a predefined structure. It is a good idea to have a general structure, but then provide the specific structures for each separate quality characteristics. Priority: ***crucial*.**

- **AP2**: (***Timing*)**. We should define procedure what needs to be done if evaluators do not contribute on time. This should shorten the factual assessment time. Priority: ***crucial*.**

- **AP3. *Plan Supervision*:** The planing of the actual assessment should also be performed. Priority: ***normal*.**

- **AP4**: ***Operation 4*:** Define a procedure to achieve a unified scope for different quality characteristics. Some suggestion (but not the final) might be that (i) responsible person gathers the scope from different evaluators; (ii) responsible person defines one scope according to the gathered information; (iii) responsible person sends this scope to all the evaluators. Another way for this scope definition: Responsible person needs to create the history table including different versions of the project. The evaluators choose from this table the versions depending on assessment information availability. The rule - to select as higher version as possible. The limitation - overall assessment still might have be done having different scope. For some quality characteristics (e.g., maintainability, reliability) the very precise scope can be defined and agreed between evaluators. For other quality characteristics (e.g., documentation availability, software processes) the broad scope can be defined. But this broad scope needs to be as close to the selected project version as possible. In both cases it is very important to document the current? scoping information in the log file. Priority: **crucial**.

- **AP5. *Operation 5*:** Define additional rules. Like: "Rules – Operation 9 cannot take place in Benchmarking assessment" Priority: ***normal*.**

- **AP7. Operations 7 and 8:** These operations were not executed during the performance test (in Task3.2). In addition, it seems that these operations are not useful for the overall FlOSS assessment process. We recommend remove them from the process. Priority: ***crucial*.**

- **AP8: *Operation 13*:** Establish and adapt the procedure to prepare the debriefing. Gathering the agenda points and providing this agenda before the debriefing is a good idea. Priority: ***crucial*.**

•**AP9. *Operation 14*:** Evaluator needs to participate (either as observer, or as discussant) in the interview process. Priority: ***normal*.**

Improvement suggestions for the quality model. There are three improvements for the quality model:
•**QM1. Adjustment of the Community member characteristic:** The case study (see Section 2.3) and the FlOSS endeavour assessment (see Section 3) showed that characteristic *Community member/Size and regeneration adequacy* were difficult to assess. We recommend not to include this characteristic in the version 1.0. For the later versions of the Standard QualOSS assessment method, this characteristic need to be more developed. Priority: ***crucial*.**

•**QM2. Adjustment of the Software processes:** The quality characteristic *Software processes/Capability of Support and Community Management* was not developed during any of the model definition/validation activities. It appeared to be too complex. We recommend removing it from the Quality model. Priority: ***crucial*.**

•**QM3. Adjustment of the Tools and Dependencies:** None of the quality characteristics were defined for the *Tool and dependencies* dimension. It appeared to be too complex and time consuming. We recommend removing this dimension from the quality model. Priority: ***crucial*.**

Improvement suggestions for the quality characteristics (metrics and indicators). We suggest improvements for each of the quality characteristics. The suggested improvements are:
•**QC1*. Maintainability*:** Refine and adjust the indicator thresholds according to the benchmarking results. Priority: ***crucial*.**

•**QC2*. Maintainability*:** Define/improve the scoping procedure. Main differences between releases in source code, and also embedded code. It provided “odd” results since there were found strong changes among different releases. We should care about the third parties source code. Priority: ***normal*.**

•**QC3*. Maintainability*:** There exist some problems with measurement process [9]. We suggest to correct the following mistakes. Firstly, problem in Normalizing the source code to measurement (some code outside src/ so what should be done. Secondly, some problem using the given script (sqlite vs mysql). Problem with % of changes between releases. It returned a large negative value. Priority: ***normal*.**

•**QC4. *Reliability*:** Refine and adjust the indicator thresholds according to the benchmarking results. Priority: ***crucial*.**

•**QC5. *Documentation availability*:** Define different thresholds based on the results of project assessment. Priority: ***crucial*.**

•**QC6. *Documentation availability*:** Correct the error of language-versioned spreadsheet. Priority: ***crucial*.**

•**QC7. *Test*:** Refine and adjust the indicator thresholds according to the benchmarking results. Priority: ***crucial*.**

•**QC8. *Test*:** Add more Automation in the spreadsheet (eg add urls in spreadsheet and measures are taken automatically) Priority: ***crucial*.**

•**QC9. *Community members*:** Refine and adjust the indicator thresholds according to the benchmarking results Thresholds should be updated (in order to better fit with Eclipse (supposed to be good enough) Platform project.) Priority: ***crucial*.**

•**QC10. *Community members*:** Recreate queries for the SF. Add some metrics in order to have more indicators and better vision of the project. Priority: ***normal*.**

•**QC11. *Software process*:** Refine and adjust the indicator thresholds according to the benchmarking results. Priority: ***crucial*.**

•**QC12. *Software process*:** Define what needs to be done when project is found under umbrella of a bigger project. Sometimes it is hard to retrieve data from projects that are found under the "umbrella" of a bigger foundation. This is the case of Evince where a significant portion of the information refers to general GNOME resources and infrastructure. Priority: ***normal*.**

•**QC10**: (***Software processes***) Define a name for each metric. Priority: ***crucial*.**

Improvement suggestions for the documentation. Some suggestions are specific for individual characteristic documentation, some are broad and general. We put them all under the same group, since they all consider improvements for the QualOSS documentation. The improvements are:

•**DC1. *Purify and document*:** One improvement related to all quality characteristics is to purify these characteristics and provide the fully documented description for the version 1.1. Priority: ***crucial*.**

•**DC2. *Document tools*:** One improvement is related to some quality characteristics (e.g., ***Maintainability***, ***Reliability*** and ***Community members***) and it documents how different tools need to be configured for assessment. Priority: ***crucial*.**

•**DC3. *Better scoping*:** Improve and define better scoping procedure (***Reliability***). Define the way how to collect scoping information (e.g., for ***Software processes*** currently it is add-hoc and not clear). Scope information can be defined either in a separate document or the template have to to adapted for this. Maybe Selenium tool (or other tools) might help. Improve the measurement procedure with the scope review step (e.g., ***Documentation availability***). Establish the common or close procedure to define similar scoping with other characteristics (e.g., ***Community Members***). The scoping was made in a totally different way among reliability, maintainability and community metrics. In the source code management system third source code parties were found and they should not be included in the assessment process. Following this idea, they were removed in order to fit with a similar directories hierarchy structure in releases and source code management system. Priority: ***crucial*.**

•**DC4. *Coding conventions*:** Adapt a coding convention tool to the assessment procedure (e.g., ***Reliability***). Priority: ***crucial*.**

•**DC5. *References between tabs***: Provide the cross references between different spreadsheet tabs (e.g., ***Documentation availability***). Priority: ***crucial*.**

•**DC6. *Time-based releases*:** Define the procedures/rules to check what are the time-based releases (e.g., ***Test***) Priority: ***crucial*.**

To conclude this deliverable, we must note that we have performed a number of validation activities. We showed its validity in the performance test - we have applied the standard QualOSS assessment method both in the case study and to assess the FlOSS endeavours Also we executed some correctness test, like comparing the results of manual definition of metrics and indicators against definition using the datamining technique. We conclude that the current version of the standard QualOSS assessment method is valid with

respect to its performance and at same level with correctness. The future work includes validating the standard QualOSS assessment method with respect to the usability. This means that the method could be applied by different organisations that are outside the QualOSS consortium Certainly the future work also includes the implementation of the suggested improvements and performing validation activities of the new releases of the method.

REFERENCES

[1] Deprez J.-C., Haaland K., Kamseu F., QualOSS Methodology & QualOSS Assessment Method, Deliverable D4.1 of the QualOSS project, 2008

[2] Soto M., Ciolkowski M., Deprez J.-C., Ruiz J., Herraiz I., Campos C. G., Matulevičius R., Metrics and Indicators of the Standard QualOSS Assessment Method, Deliverable 4.2 of the QualOSS project, 2009, Final

[3] Basili, Victor R., "Software Modeling and Measurement. The Goal/Question/Metric Paradigm", Computer Science Technical Report Series NR: CS-TR-2956 / NR: UMIACS-TR-92-96, 1992.

[4] Ruiz J., Ghosh R., Glott R., Haaland K., Deprez J.-C., Flamand J., Case Study Design and Pilot Projects, Deliverable D5.1 of the QualOSS project, 2009

[5] Cortázar D. I., Measurement Tools Deployment and Data Collection, Deliverable D3.2 of the QualOSS project, 2009

[6] Matulevičius R., Cortázar D. I., Deprez J.-C., Description of the Assessment of the Evolution project, 2009

[7] Cortázar D. I., Guilarro R. P., Description of the Assessment of the Eclipse Platform, 2009

[8] Tinte M. A., Cortázar D. I., Description of the Assessment of Evince, 2009

[9] Deprez J.-C., Description of the Assessment of the jMeter project, 2009

[10] Auvray V., Deliverable D4.3 of the QualOSS project, 2009