Hindawi Publishing Corporation
Advances in Software Engineering
Volume 2012, Article ID 235392, 17 pages
doi:10.1155/2012/235392

Research Article

Assessing the Open Source Development Processes Using OMM

Etiel Petrinja and Giancarlo Succi

Center for Applied Software Engineering, Free University of Bozen/Bolzano, 39100 Bolzano/Bozen, Italy

Correspondence should be addressed to Etiel Petrinja, etiel.petrinja@unibz.it

Received 14 May 2012; Revised 2 August 2012; Accepted 6 August 2012

Academic Editor: Gerardo Canfora

The assessment of development practices in Free Libre Open Source Software (FLOSS) projects can contribute to the improvement of the development process by identifying poor practices and providing a list of necessary practices. Available assessment methods (e.g., Capability Maturity Model Integration (CMMI)) do not address sufficiently FLOSS-specific aspects (e.g., geographically distributed development, importance of the contributions, reputation of the project, etc.). We present a FLOSS-focused, CMMI-like assessment/improvement model: the QualiPSo Open Source Maturity Model (OMM). OMM focuses on the development process. This makes it different from existing assessment models that are focused on the assessment of the product. We have assessed six FLOSS projects using OMM. Three projects were started and led by a software company, and three are developed by three different FLOSS communities. We identified poorly addressed development activities as the number of commit/bug reports, the external contributions, and the risk management. The results showed that FLOSS projects led by companies adopt standard project management approaches as product planning, design definition, and testing, that are less often addressed by community led FLOSS projects. The OMM is valuable for both the FLOSS community, by identifying critical development activities necessary to be improved, and for potential users that can better decide which product to adopt.

1. Introduction

Free/Libre Open Source Software (FLOSS) development approaches differ from the traditional software development approaches [1] such as the waterfall or the spiral. The FLOSS approaches have specific characteristics as the geographical distribution of the development team [1]. The developers usually do not know personally each other, there are no budget constraints, and so forth. However, some traditional software development issues as [2]: faults insertion, continuous change of requirements, and growing complexity, are present also in agile and FLOSS projects with additional critical aspects that have to be addressed. Some of these, are for example, issues related to a strongly distributed development process, and absence of formal responsibility of developers for meeting deadlines.

The software development process is increasingly being defined and standardized [3]. Assessment models have been defined for evaluating the quality of the software development process. Only by assessing it, it is possible to identify poorly implemented practices, identify missing practices, and improve the development process. Only by assessing the quality of the development process it might be possible to optimise the use of resources and reduce the development time. For successfully assessing the process, it is possible to use an assessment approach that addresses key aspects of the development process. For this reason, it is important to modify an assessment approach or use different approaches when assessing different types of software processes. One standardized approach to assess the quality of the software development process is the Capability Maturity Model Integration (CMM/CMMI) [4]. It is both an assessment and an improvement model for software development. CMMI is increasingly being adopted by the software industry and the number of experts that are knowledgeable in its structure and usage is already large.

This paper presents a FLOSS-development-process assessment and improvement model and shows its application on three case studies. The Open Source Maturity Model (OMM) [5, 6] was designed with the aim of increasing the perceived quality of the FLOSS development process [7]. The trustworthiness of FLOSS is an important criteria when potential

users decide to download and try to use the FLOSS product [8]. A study conducted on a large group of stakeholders, both from the software industry, and members of FLOSS communities [8], unveiled this. Specially stakeholders from the software industry stressed the importance of quantitative measurements of the quality of FLOSS. Without quality metrics potential users have problems when selecting and adopting FLOSS products. The main beneficiaries of the OMM are potential users of FLOSS products, especially integrators of FLOSS products. For example, software companies that adopt FLOSS products and integrate them into composed software products that can be either FLOSS or proprietary. FLOSS communities can also benefit from a large number of users attracted by a good OMM assessment, especially integrators from the software industry that can contribute new code to the project. The results of the conducted stakeholders survey was an important source of information for the design of a new assessment model. The large majority of metrics listed in OMM are addressing quantitative measurements of the FLOSS development process suggested by the stakeholders included in the study.

The existent FLOSS assessment models have not yet been largely adopted. However, when they are used, they provide indications about the quality of FLOSS projects. The OMM maintains some similarities with existent models. However the advantages of OMM, if we compare it with other FLOSS-oriented assessment models, is its particular focus on the development process. The OMM prescribes practices as: time-related implementation and management of the testing process; support of good maintenance practices; measure response time of the community to improvement suggestions from the community, the performing of configuration audits; inclusion of the plan for process quality; and so forth. Available FLOSS models sometimes address similar aspects, however at a coarser level of details. Aspects that distinguish the OMM from other models are often related to the management and the improvement of adopted software development practices.

The assessment of the development process can bring benefits not just to the further improvement of the process itself but also to an increase of the quality of the FLOSS product. By omitting the analysis of process characteristics as: the quality of the testing plan, the project, and the process-planning characteristics, and others, we could miss aspects related to the survivability [9] and the quality of the development process. These aspects are useful indicators, not just of the current quality of the project but also, of its potential future evolution. The OMM helps when deciding to adopt a FLOSS product, specially if the product will be integrated in a composed product created by a software integrator. Compared to CMMI, OMM addresses FLOSS-specific aspects that are not addressed by CMMI. A CMMI assessment is usually an extensive process that is not used for assessing small software companies or single-software projects. On contrary, OMM was designed specifically for assessing single FLOSS projects, and the assessment process is considerably shorter than the CMMI assessment. The OMM should cover all key aspects of the FLOSS, however it should be flexible and allow the assessment of just some parts of the development process.

Despite OMM diverges in several aspects from CMMI, there are similarities. The structure, the content of the model, and the assessment process share common elements. Part of the motivation for designing a similar model was based on the fact that CMMI is currently the largely used software development process assessment methodology. Although there are still several problems adopting CMMI in the software industry, there is already a considerable expertise in the software industry that we wanted to benefit from. Experts from the software industry that know CMMI can easily become users of the OMM model.

Three key contributions of this paper are:

(1) A description of the QualiPSo OMM, a FLOSS-oriented development process assessment model.

(2) A report of the application of the QualiPSo OMM on six-case studies; assessing FLOSS projects started and led by a software company (two SMEs and one large software integrator company) and projects developed by FLOSS communities.

(3) The presentation of improvement opportunities for the assessed FLOSS projects.

Section 2 presents the related work. The third section describes key characteristics of the used assessment model. In section four, we present how the OMM model should be used. In section five, we present six case studies. We present the results of the assessment of the FLOSS projects in the sixth section. The seventh section contains the discussion of the most important results. In the section eight, we present the threats to validity and finally we draw the conclusions and propose the future work.

2. Related Work

The adoption of software assessment methodologies has been studied extensively in the last decade [10–13]. Roberts et al. [12] studied factors influencing the adoption of software methodologies. They identified factors as: the organisational system development methodology (SDM) transition, the functional management involvement/support the use of models, and the external support. The study provided useful indications to researches by identifying a list of measures to be checked, and to developers by providing a guide to aspects that should be considered when implementing a system. The study conducted by Misra et al. [14] provides a list of best practices and factors that influence the quality of Agile software development. Most of those factors are also part of the OMM model. Khalifa and Verner considered human behaviour when analysing the adoption of the waterfall and the prototyping methodologies [10]. By studying the behaviour of developers and other stakeholders they identified two factors that influence the perceived quality of the development method: the facilitating conditions, and the process quality. The product quality was not identified as a statistically significant factor for explaining the usage. Their work sheds some light on the perceived quality of the development process and the impact of the methodology. Their results show the importance of the development

process adopted by software developers which was one of our motivation factors when designing OMM. Matook and Indulska conducted a study of the development and measurement of the quality of process models by using the quality function deployment approach and proposed a tool that can be used for evaluating process models [11].

The quality of reference models can influence the final quality of the developed software and reduce development costs and development time. von Wangeheim et al. conducted a study on the methodological support of creation and modification of the software process capability/maturity models (SPCMM) [13]. An important aspect they have identified is that "SPCMM elements are not explicitly and systematically related to quality and performance goals." We adopted some of the approaches proposed by the presented studies. Several results are aligned with findings described in the cited literature. The key difference is our focus on the FLOSS development process.

The CMMI model has been increasingly adopted by software companies [4]. The research of the applications of CMMI showed that after a decade of the availability of the CMM half of the companies involved in the SEI study were classified in its lower maturity level [4]. Other studies reported an even higher level of companies classified as CMM level 1 [15]. Recently, discording studies of the adoption of CMMI had been published; some studies found a growth and improvement of the quality of assessed companies [16], on contrary Staples et al. [17] report that the adoption of CMMI is still difficult for several reasons. For example, one reason is the cost and the complexity of the CMMI assessment process. Companies might not be interested to adopt the CMMI model due to its complexity or because they use other assessment and improvement models.

There are several studies presenting modifications or extensions of reference models. Many of them analyse the adoption of CMMI in small and medium enterprises (SMEs). Staples et al. [17] and Guerrero and Eterovic reported that the adoption of CMM is difficult in small enterprises [18]. Methodologies are often adapted to specific assessment needs or simplified methodologies are used [17]. Assessment methodologies have been proposed also for the agile software development approach [19]. Paulk studied the issues related to the use of agile approaches and the CMM assessment methodology [20]. He concluded that both methodologies can benefit from each other. von Wangeheim et al. [21] have studied the adoption of ISO/IEC 15504 in SMEs and found that the adoption level of Reference models is much lower in SMEs than in large companies. Dybå [22] presented the adoption of process improvement methodologies in the Scandinavian Context focusing on the quality management aspects which we considered important when we designed OMM. Contrary to other studies, he found that SMEs implement process improvement elements efficiently as large companies. Our paper describes the use of a reference model dedicated to FLOSS projects. During the study we have encountered many issues identified by the cited studies that were all conducted on non-FLOSS software projects. Their findings were important for understanding the issues that we found during our research.

The Open Source Maturity Model (OSMM) assessment methodology for FLOSS projects was presented by Cap Gemini in year 2003 [23]. Afterwards several new methodologies were proposed, some of them are: the Open Source Maturity Model (OSMM) from Navica Inc. [24], the Methodology of Qualification and Selection of Open Source software (QSOS) [25], the Open Business Readiness Rating (OpenBRR) [26], the Open Business Quality Rating (Open BQR) [27], the Qualoss methodology [28], and the SQO-OSS quality model [29]. To our knowledge there are only few studies analysing and validating previously listed FLOSS assessment models. A comparison study of OpenBRR and QSOS was performed by Deprez and Alexandre [30]. Their study identified positive and negative aspects of both models. A similar approach was done by Petrinja et al. [6]. The authors of the study compared the use of OpenBRR, QSOS, and OMM to assess two FLOSS projects: Firefox and Chromium. Most of these models have a repository available on the web where users can see examples of assessments of popular FLOSS projects; there are, for example, 101 available assessment results obtained using the QSOS model.

Some of these methodologies were proposed by private entities as, for example, the OSMM methodology. More often they were proposed by research centres, universities, or individuals. They share common aspects, the most important is the set of software characteristics they measure. We analysed those characteristics and reused some of them while designing OMM. The reuse of characteristics included in other methodologies was dictated by the aim of providing an as possible complete measurement of the FLOSS process.

3. Characteristics of OMM

The OMM is a FLOSS assessment and improvement model that was designed with the aim of being able to support the assessment of the quality of the FLOSS development process. FLOSS adoption is susceptible to the lack of trust in its quality, both of the development process and of the software product. The quality of the software product is related to the quality of the development process that is adopted to produce the software product. Our aim was not just to improve the stakeholders perception of the quality of FLOSS, by providing a detailed set of metrics characterising the development process, but also of its quantitatively measurable quality. The lack of trust in FLOSS is often unjustified, and therefore, it is important to understand what are the issues that hinder the trust.

3.1. The Inputs for Building the Model. The information gathered for the design of the model was based on different types of users, development approaches, and on previous studies. We have conducted personal interviews with 52 individuals in the first iteration of the research [8]. The interviewees were experts from the software industry (Siemens, Engineering Ingegneria Informatica, Bull, Atos, IBM, Mandriva, Thales, etc.) and members of FLOSS communities (Apache HTTP Server, Eclipse, Emacs, Linux Kernel, Mozilla project, GNOME, Debian, etc.). The two groups were equally represented. The majority of participants covered managerial

and development roles. Some of them were working in the industry and at the same time participating in FLOSS communities. Following a predefined questionnaire, we collected opinions and the practices interviewees use when developing, using, or adopting FLOSS. During the interviews we focused on: the quality, the FLOSS stakeholders, the technology used, and the business aspects. However, the whole questionnaire covered sixteen topics and contained 53 questions. Interviews were usually face to face and the meetings lasted between one and two hours. There were always at least two researchers participating to the meeting and annotating the answers. After the meeting, the draft of the answers collected during the interview was sent to interviewees that had to confirm its correctness. We conducted a second iteration of the information collection in the form of a survey that we conducted with the help of a web questionnaire. After analysing the results of interviews and surveys we identified the areas of the FLOSS development process that participants consider important for improving the quality of FLOSS.

Other sources of information for the design of the model were [7]: the literature review, and the study of existent FLOSS assessment methodologies and standards as: OpenBRR, QSOS, CMMI, ISO/IEC 15504, and so forth. These sources were used when the answers from the experts were lacking sufficient details about the development process they mentioned.

Interviews and surveys were the main source for topics that are important for FLOSS stakeholders. These topics are measured on different granularity levels inside OMM. The CMMI principles influenced the design of the OMM structure and the assessment process. From the literature review we gathered additional details about FLOSS practices and the whole development process. These elements were inserted into the OMM as questions on different levels of the model.

3.2. High-Level Components of OMM. The key components of OMM are elements focusing on important aspects of the FLOSS development process. We identified a list of elements that influence the perceived quality of the FLOSS development process by a large group of stakeholders in FLOSS projects. We name these components TrustWorthy Elements (TWEs). From interviews and surveys, we obtained a list of TWEs that were mentioned by a large percentage of experts. We were able to compose a ranked list of topics where some of the TWEs were considered important by a large percentage of experts. The addressing of the needs of software users and integrators, that are the first potential users of OMM, influenced the type of elements inserted into OMM. Some questions as for example: "check the availability of the requirements specification" can be considered not important for simple FLOSS projects. However, the majority of successful FLOSS projects that have many users and contributions, sometimes contributed by the software industry, take in consideration aspects related to requirements specification and other aspects important for a (FLOSS) software project. Several elements in the structure of the proposed model resemble elements in CMMI; OMM is

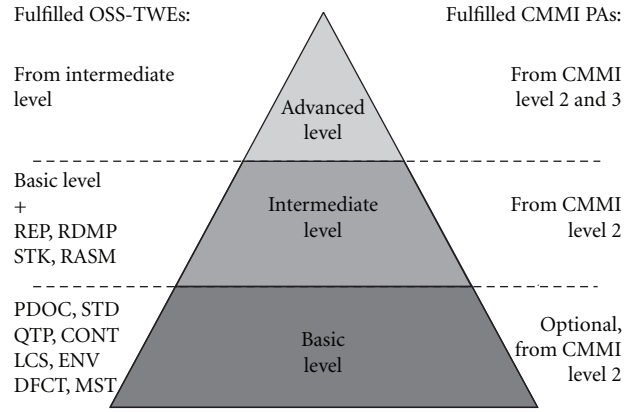

Figure 1: Three OMM maturity levels.

structured in levels, practices are important building blocks of the model, they focus on elements that are assessed also in the CMMI, and so forth. Both models aim to be usable for assessment but also for improvement of the development process. Similarly as in the CMMI model, we have proposed a level structure for the OMM model. We limit the number of maturity levels to three (in CMMI there are 5 maturity levels) (see Figure 1):

(1) the basic level with 11 TWEs,

(2) the intermediate level with 7 TWEs,

(3) the advanced level with 7 TWEs.

When we talk about maturity we used a simple definition (Dictionary.com) that says that a software product or a software process is mature when it has reached a high (or full) development. The full development in OMM is when all practices that are expected by stakeholders, quality standards, and the technological process are fulfilled.

The TWEs were distributed into three maturity levels according to their importance for interviewed and surveyed stakeholders in FLOSS projects. TWEs considered important for improving the perceived quality of the FLOSS development process were inserted into lower OMM levels. For example, license management was considered important for a large percentage of participants to the initial research, therefore we have decide to put the TWE LCS into the Basic OMM level; the level that should be first implemented. We based our classification of TWEs also on the complexity of its implementation. Reputation of a FLOSS project has also an important influence on the perceived quality of the assessed FLOSS project, however, we decided to put the REP TWE into the Advanced OMM level because it is not easy to build the reputation of a FLOSS project. Our initial distribution of TWEs in the three levels was tested during the validation of the model. After collecting the results of the assessments we identified few TWEs that had to be moved between levels.

The information about the maturity level reached by a FLOSS project provides a single figure about its maturity. Somebody can argue that the collection of TWEs and their distribution into the three maturity levels is not the only possible interpretation of the maturity of a FLOSS project.

We think that a better source of information about the quality and maturity of a FLOSS development process is the whole set of assessment values for each TWE. These values represent quantitatively measured characteristics of the FLOSS project. We identified a set of 25 key TWEs:

(i) PDOC—Project documentation.

(ii) STD—Use of established and widespread standards.

(iii) QTP—Quality of the testing process.

(iv) LCS—Licenses management.

(v) ENV—Environment.

(vi) DFCT—Number of commits and bug reports.

(vii) MST—Maintainability and stability.

(viii) CM—Configuration Management.

(ix) PP1 and PP2—Product planning, project planning.

(x) REQM—Requirements management.

(xi) RDMP and RDMP2—roadmap.

(xii) STK—Stakeholders.

(xiii) PPQA—Process and product quality assurance.

(xiv) PMC—Project monitoring and control.

(xv) TST1 and TST2—Test.

(xvi) DSN1 and DSN2—Design.

(xvii) CONT—Contributions.

(xviii) RASM—Results of third-party assessment.

(xix) REP—Reputation.

(xx) PI—Product integration.

(xxi) RSKM—Risk management.

Twelve TWEs address generic software development practices, for example project management (PM) and risk management (RSKM), the others address FLOSS specific aspects as the contribution level (CONT) and the Licenses management (LCS). The TWEs are high-level elements that we specialized into smaller components that can be assessed easier and increase the granularity level of the assessment. The identification of components that address a small aspect of the software development process limit the subjectivity of the assessment. The smaller are the assessed components of the development process, the more precise is the assessment.

Questions as: "Check the availability of FAQ documents" can be simply answered by searching through the web pages of the FLOSS project. The experience of an OMM assessment team, gained with the assessment of several FLOSS projects, help to precisely assign an assessment value to each practice. Based on experiments conducted during the validation of OMM [6] we identified that distinct assessors assign similar grades for measured characteristic. The variability of assessment values, based on the subjective perception of questions and documents evaluated, is small.

3.3. Low-Level Components of OMM. We used the Goal Question Metric approach (GQM) [31] to derive the components of the OMM model that are part of the TWE elements. In the GQM we have to identify the Goals that we wish to achieve; the Questions to which we will have to answer to be able to know if the Goal has been reached; and the lower level is composed by metrics that have to be measured to answer to the questions. The number of the goals specified varies between one and five and depends on the complexity of the TWE to which they belong. Some TWE (e.g., Design 2 (DSN2) have 5 goals and addresses aspects as requirements, components solutions, and design decisions) have several goals, others (as Roadmap 2) have just one goal. The elements were extracted by the interviews and surveys conducted, from the literature study, from other models, and from the experience of the team that designed the OMM. Questions were written in a structured form and they were circulated inside the team. A common agreement had to be reached inside the team on the elements inserted into OMM.

We have designed the create, the manage, and the improve set of questions and most TWEs contain a goal that is addressing each of these three aspects. A similar design is used in the CMMI. Goals first address aspects related to the "creation" of: the documentation (e.g., the PDOC TWE has a goal: "Create product documentation."), software components, processes, and so forth. The "manage" type of goals address the management of already created elements. The last type of goals addresses the activities related to the "improvement" of the development practices adopted, the software components created, the documentation, and so forth. It was not always possible to include the three types of goals for all TWEs but we followed this structure wherever it was possible. Each goal is composed by one or more practices. They specialize goals and characterize activities that are usually conducted inside the FLOSS project. OMM contains 122 practices for the complete assessment process. The GQM approach requires the definition of "questions" that inside the OMM are called "practices" for maintaining the similarity with the CMMI naming convention. An example of a practice from the PDOC TWE is:

> *Practice PDOC.3.2—"Improve the support for several natural languages."*

Metrics are the lowest level of the OMM model. In the current version of the model there are 630 metrics. The metrics are used to measure if a practice is fulfilled completely, partially, or not at all.

The number of metrics is large and it may be perceived as too large for a FLOSS assessment. However, the granularity level of metrics in OMM is optional; assessors can decide to assess just the practices and not consider metrics in details. An assessment without metrics will be of smaller precision and with less value for people reading the results, but it will still give a usable OMM assessment result. The 122 questions representing practices are comparable to the level of details of methodologies as QSOS and OpenBRR and it can be assessed in a time frame ranging from two to four hours depending on the size of the assessed project.

3.4. The Rating Mechanism. In the OMM model each assessed value can reach a specific threshold value. These should be defined uniformly for the whole model to limit the bias. We have decided to use a uniform threshold value for the whole model; there are four possible assessment values ranging from 1 to 4 with the possibility to assign a 0 for metrics that do not apply in specific domains. For example Practice ENV.1.3 (Select integrated management and communication tools used in the project) in the TWE assessing the Environment aspects contains several metrics that should be searched and measured, for example: Eclipse, BlueFish, Kdevelop, and others. A FLOSS project will probably adopt just one of the development environments and not all of them. During the assessment we have to measure the diffusion of the use of the adopted environment; all others will be considered not applicable (assessed with 0). The threshold value of 4 means that the implementation of the requested elements is fulfilled more than 75%, value 3 means that it is fulfilled between 50% and 75%, value 2 between 25% and 50%, and value 1 that its implementation is lower than 25% of what requested by the practice.

We have decided not to prescribe a complete list of elements that have to be implemented in a project to reach a specific threshold level as it is done in OpenBRR and QSOS. The first version of the OMM model contained the exact definition of threshold values, however the elements present in each level were quite arbitrary and based on the results of the first validations we decided to insert this information as subprocess elements that have to be considered but are not strictly prescribed for reaching a threshold level. Therefore, the decision to assign a 1, 2, 3, or 4 value depends on the decision of the assessment team based on the information available. The assessor proposes an assessment value (1, 2, 3, or 4) based on his previous domain knowledge and the assessment team can agree with the value or propose new documents that support a different assessment value. The documents showing the activity of the FLOSS project used during the assessment should be collected and archived by the assessors and made available as proof of the assessment result.

A detailed description of all 25 TWEs will not be provided in this paper (for details see the document: http://www.qualipso.org/sites/default/files/Open Maturity Model.pdf). We present here just part of the structure described in previous sections that is schematically presented in Table 1. We can see all four granularity levels of the OMM model related to documentation aspects inside the FLOSS project:

(i) the TWE: PDOC: Project Documentation;

(ii) description of the purpose of the PDOC TWE: Develop and maintain project documentation, making it readily accessible to the community;

(iii) the first goal PDOC.1: Provide high quality documentation;

(iv) the goal is specialised into three practices, the first one is PDOC.1.1: Create development documentation;

(v) the lowest granularity level (LookFors) contains the set of documents we have to search in the assessed project (for example: Check the availability of requirements specification).

We can see in the example questions related to the software product (check the availability of a user's guide) and questions related to the development process (check the availability of work flow guidelines). The assessment team must decide what is the quality and completeness of the requirements specification documentation available in the project based on the available documents and previous experience of the team.

The OMM model divides the development process into small activities that can be assessed quantitatively. The experience of the assessor and the adoption of assessment best practice guidelines guarantees a good precision of the whole assessment process. Case studies and experiments conducted [6] demonstrate that the value assessed by more individuals does not vary strongly. Therefore, in general we can be confident in the objectivity of the assessment result.

The rating (R) mechanism defines how to aggregate the assessment values for the FLOSS development process by aggregating the assessments on different granularity levels: the OMM maturity level $R(\mathrm{ML}_i)$, the TWE level $R(\mathrm{TWE}_i)$, the goal level $R(G_i)$, and the Practice $R(P_i)$ level. The rating is performed by calculating the average value of the elements aggregating the elements from the lower level (see (1) for Practice rating, (2) for Goal rating, and (3) for TWE rating).

Calculation of the Rating of a Practice

$$R(P_i) = \frac{\sum_{\text{All Metrices}} \text{Metric}_i}{\text{number of Metrices}}. \tag{1}$$

Calculation of the Rating of a Goal

$$R(G_i) = \frac{\sum_{\text{All Practices}} \text{Practice}_i}{\text{number of Practices}}. \tag{2}$$

Calculation of the Rating of a TWE

$$R(\mathrm{TWE}_i) = \frac{\sum_{\text{All Goals}} \text{Goal}_i}{\text{number of Goals}}. \tag{3}$$

The weighting of metrics (assigning different importance to some metrics) is subjected to several potential risks [32] that can distort the assessment result. We decided to do just a simple averaging of metrics. Offering to readers of the assessment results a preferred level of granularity provides a more informative source of details about the assessed FLOSS project. We are considering also additional weighting mechanisms, but for an empirically supported mechanism we need a large set of assessment data which will be collected with a larger usage of the model.

The overall rating can be calculated to:

(i) identify the OMM maturity level of the FLOSS project,

Table 1: Mobile OS project assessment results for the project documentation (PDOC) Trustworthy element.

PDOC: Project documentation Purpose: Develop and maintain project documentation, making it readily accessible to the community.		2.11
Goal/practice	Goal/practice description	
Goal PDOC 1	Provide high quality documentation	2.67
Practice PDOC-1.1	Create development documentation	3
LookFor	Check the availability of requirements specification	3
	Check the availability of high level design/product architecture	4
	Check the availability of detailed design	3
	Check the availability of technical documentation (e.g., for use in debugging)	2
	Check the availability of workflow guidelines (for checking, testing...)	4
Practice PDOC-1.2	Create user documentation	3
LookFor	Check the availability of a user's guide	3
	Check the availability of FAQ documents	3

(ii) to identify the percentage of implementation of specific TWEs or goals.

The calculation of the maturity level is not performed by aggregating the values of TWEs in that maturity level but by using (4):

Calculation of the OMM Maturity Levels

$$R(\mathrm{ML}) = \frac{\sum_{\text{All practices}} P_i}{\max \sum_{\text{All practices}} P_i}. \quad (4)$$

There are two possible outcomes:

(1) OMM Level fulfilled: $R(\mathrm{ML}) \geq 90\%$,

(2) OMM Level not fulfilled: $R(\mathrm{ML}) < 90\%$.

There can be differences between FLOSS development processes, therefore the assignment of the OMM maturity level allows flexibility. For specific FLOSS projects some practices can be of limited importance and the project is not implementing them. The project can be otherwise of high quality but because of just few not important practices it would not reach a specific maturity level. By assigning a maturity when fulfilling 90% of practices, the approach provides some flexibility. The OMM model can be used for assessing different FLOSS projects, therefore it is not convenient to have a rigid set of practices that have to be fulfilled in all types of FLOSS projects. By using (4), a FLOSS project can fulfil an OMM maturity level if it implements more than 90% of practices required for the specific level.

4. The OMM Assessment Process

There are two possible assessment approaches envisioned by OMM:

(1) a complete (internal) OMM assessment where in the assessment process participate individuals who have access to all documents of the assessed project, and

(2) a partial (external) OMM assessment where the assessors assess only documents available: on the web, in mailing lists, in forums, in the source code, and so forth.

The first approach is intended for companies or organizations that want to integrate the assessed FLOSS tool with their software products or they want to start contributing to the FLOSS project. The first approach is also appropriate for companies that have started a software project and they wish to monitor its development. The second approach is more common for individuals who are interested just in the overall quality or in few specific quality aspects of the assessed software project.

FLOSS projects usually publish their documents on the web, however it is still not always possible to find information about the project development process, as for example some architectural decisions. This is especially the case for FLOSS projects that are strongly supported by a software company. Many decisions and activities, in this type of projects, are taken according to companies' rules and these information are often not publicly available on the web.

The OMM assessment is done by using the assessment questionnaire covering all TWEs. Each element has to be assessed or identified as not applicable (for few cases). It is not necessary to rate all metrics, however they should be taken in consideration when assessing practices to which they belong. All practices have to be assessed for obtaining a complete OMM assessment.

The assessment team reviews relevant documents and the information related to the assessed practice and has to meet an agreed assessment value. After reviewing the available documents and listening to the descriptions of the representatives of the FLOSS project, the assessor proposes the assessment value based on assessment rules and assessment best practices. If the project representatives do not agree with the assessment value they can object and present missing documents or explanations. The last assessment should be agreed between assessors. All the documents used during the assessment should be collected and archived as source of information for the assessment. Documents reviewed during

the OMM assessment are numerous, ranging from: the documentation available on the web, to bug/issue reports, the concurrent versioning system logs, the mailing list archives, the forum archives, the software code itself, and so forth. Most of the information needed for assessing a FLOSS project are available online. In FLOSS projects led by companies few documents can be restricted and they cannot be published, however for a complete assessment the assessment team should be able to evaluate them.

5. Case Studies Design

We present the results of six case studies performed on six FLOSS projects. We will use the following names to refer to the projects: the Mobile OS Project, the Network Monitoring Project, the Business Intelligence Project, the Apache community Project, the SourceForge community project, and the Mozilla community Project. The first three projects were started by three software companies that still have a leading role in the assessed FLOSS projects. Their developments have been strongly influenced by decisions taken by the three companies. The contributions from the FLOSS community have grown slowly. The second three projects are typical examples of community grown projects hold in three different FLOSS projects web available repositories. The studies presented are not proper "Yin" case studies but detailed inspections of software products and software development processes. The assessments were conducted by the designers of OMM together with people who are involved in the assessed projects. The three companies might use recognised assessment schemes such as ISO9001, ISO15504, or CMMI for their development, however none of these addresses FLOSS specific characteristics. The OMM assessment of the three community grown projects were conducted by the OMM designers with the help of master students.

For the studies, we used two slightly different assessment scales; for assessing the Business Intelligence and the Network Monitoring projects we used a three-levels scale (1–3), while when assessing the other projects we decided to extend the scale to four levels (1–4). We have to take in consideration this difference when comparing the results of the six assessments. The comparisons we do between the projects are limited to a simple observation of which project obtained better grades for specific characteristics. The difference of scales prevents us to do detailed comparisons, however, this was not the purpose of this study. We just assessed the projects for demonstrating the use of OMM, and we present the results in common tables. We decided to change the scale from three levels to four to study the precision of the assessment results. The only difference is in the middle value 2 that was split with the new scale into two values: 2 and 3. In the new scale (1–4) value 1 is still 1 and value 3 became 4. By splitting the middle threshold value into two thresholds assessors can specify if a metric is above (obtaining a 3) or bellow (obtaining a 2) the average implementation of the assessed metric. For a direct comparison of assessed values from all projects we have to normalize the values. The normalized assessment values are presented in Table 2.

The OMM assessment process usually lasts from 5 to 12 hours for larger FLOSS projects. In our case, the shortest assessment time was needed for the Mobile OS project that was finished in 5 hours, and the longest time was necessary to assess the Business Intelligence project that was finished in 10 hours. The time necessary for the assessment is shortened if the participants of the assessment team coming from the assessed FLOSS project prepare for the assessment by reading the OMM questions and preparing the documents and figures required during the assessment.

During the assessment of industry based projects there were participants from the team that built OMM and one or more participants from the assessed FLOSS project. During the assessment the OMM questionnaire was used. The creators of OMM were reading the questions and the members of the FLOSS project provided answers and when available showed documents and data to confirm their answers. Based on the previous experience of assessors with the assessment of other FLOSS projects and the documentation presented, a grade was assigned for each characteristic measured by OMM. The final grade was agreed inside the assessment team. At the end of the assessment the values were aggregated and the figures on different levels of granularity were calculated. The assessment team prepared a consolidated report with analysis of the assessments and improvement suggestions for the FLOSS project. The members of the assessed industry started FLOSS project had to review the final assessment and to confirm the results.

5.1. The Mobile OS Project. The company developing the Mobile OS Project was founded in 2002. The company has grown to be an important mobile open source market player with millions of downloads. It offers solutions based on a dual license; the commercial software is used by companies in the mobile industry, including software firms, device manufacturers, service providers, system integrators, and others. Currently it employees 85 professionals.

The Mobile OS project was started in the same year as the company and it represents its key product.

5.2. The Network Monitoring Project. The company developing the Network Monitoring Project was founded in 1993. From the beginning it was focused on the integration and development of the information infrastructure for science. It is implementing innovative technologies for the national scientific network of a large European country. Currently it employees 250 specialists.

The Network Monitoring project was started in 2006 and it is one of several projects run by the company.

5.3. The Business Intelligence Project. The company developing the Business Intelligence Project was founded in 1980. It is a multinational software integrator with 6300 employees distributed in several countries worldwide. Its core business is innovation in the ICT sector specialised in activities as applied software research. The company is an international

Table 2: Normalized assessment results of OMM trustworthy elements for all six projects.

	OMM trustworthy element	Mobile OS project	Network monitoring/inventory project	Business intelligence suite project	Apache community project	SourceForge community project	Mozilla community project
Basic OMM level	PDOC	2.11	2	1.89	1.89	2.89	2.89
	STD	3.38	1.5	2.25	3	2.83	2.17
	QTP	2.6	1.9	1.3	1.8	1.6	2
	LCS	3.78	1.22	2.67	3.86	2.86	2.43
	ENV	2.83	2.42	2.33	2.67	2.33	3
	DFCT	2	1.8	1.8	1.78	3	2.22
	MST	3.17	2.17	1.5	1.8	3.4	2
	CM	3.43	2.43	2.57	2.43	3	2.14
	PP1	3.22	2.33	2.11	1.67	2.11	1.44
	REQM	3.5	2.25	1.75	1.25	2.75	1.75
	RDMP	3.33	2.67	1.67	1	3	2.67
Intermediate OMM level	RDMP2	1.33	2	1	1	4	1
	STK	3	1.4	1.3	1.7	2.4	1.6
	PPQA	2.67	1.67	1	1.5	1.67	1.5
	PMC	3.63	2	1.63	1.5	2.13	1.75
	TST 1	3	2.25	1.38	1.25	2.38	1.38
	DSN 1	3.25	1.75	2.25	1.5	2.25	2
	PP2	3.6	1.4	1.2	1.4	2	1.6
Advanced OMM level	CONT	1.67	1	1.5	1.67	2.17	1.83
	RASM	2.22	1.11	1.67	1.11	2	1.44
	REP	3.2	1.6	1.6	2	2	2.6
	PI	3.56	2.22	1.67	1.78	1.89	2.33
	DSN 2	2.86	2.14	1.71	1.43	2.21	1.79
	RSKM	1	1.29	1.57	1	1.29	1.14
	TST2	3.6	2.4	2	1.2	2.6	2

player and the third IT operator in a large European country. Its work is focused on the software value chain.

The Business Intelligence project was started in 2005 and it represents only a small segment of the work performed by the company.

5.4. The Apache Community Project. The Apache community is part of the Apache Software Foundation (ASF) that was established in 1999. The community has grown to be an important supporter of the FLOSS development and it holds currently more than 100 FLOSS projects. Some of them are key building blocks of the FLOSS software suites as the Apache http web server.

The assessed FLOSS project was started in the year 2000 and is an important component for software developers. In the last years it has reached maturity and its development is mostly dedicated to maintenance activities.

5.5. The SourceForge Community Project. The SourceForge was for several years the largest web available repository of FLOSS projects. It offers several tools for FLOSS development and it holds at the moment more than 400.000 FLOSS projects. There is not a unique SourceForge community; there are thousands of small communities that form around single projects. However, they usually use the available tools and some developers participate in more than one project. The service was established in the year 1999 and it soon attracted thousands of participants.

The assessed project is a free and open source instant messaging application. The project was started in the year 2005 and it is still evolving.

5.6. The Mozilla Community Project. Mozilla is a global nonprofit organization connecting users, contributors, and developers contributing to more than 240 projects. Some of them popular as the Firefox web browser or the Thunderbird mail client. The Mozilla was created in the year 1998 with the release of the Netscape browser source code.

The assessed project was first released in the year 1998 and since than it is a popular tool for bug/issue management used by software developers.

Table 3: Number of OMM practices reaching a specific threshold level.

	OMM trustworthy element	Mobile OS project				Network monitoring/ inventory project				Business intelligence suite project				Apache community project				SourceForge community project				Mozilla community project			
	Treshold value	1	2	3	4	1	2	3	4*	1	2	3	4*	1	2	3	4	1	2	3	4	1	2	3	4
Basic OMM level	PDOC	2	4	3	0	2	5	2	0	2	6	1	0	4	2	3	0	2	0	4	3	0	2	6	1
	STD	0	2	1	5	4	4	0	0	2	2	4	0	1	1	1	3	0	3	1	2	1	3	2	0
	QTP	0	6	2	2	2	4	3	0	7	3	0	0	3	6	1	0	6	2	2	0	1	8	1	0
	LCS	0	1	0	8	7	2	0	0	0	3	6	0	0	0	1	6	1	2	1	3	2	2	1	2
	ENV	0	1	5	0	0	3	2	0	0	4	2	0	0	3	2	1	1	3	1	1	0	1	4	1
	DFCT	0	5	0	0	2	2	1	0	2	2	1	0	4	3	2	0	0	3	3	3	1	5	3	0
	MST	0	1	3	2	1	3	2	0	3	3	0	0	1	4	0	0	0	1	1	3	0	5	0	0
	CM	0	0	4	3	1	2	4	0	1	1	5	0	1	3	2	1	1	2	0	4	1	4	2	0
	PP1	0	2	3	4	0	6	3	0	1	6	2	0	4	4	1	0	4	1	3	1	5	4	0	0
	REQM	0	0	2	2	0	3	1	0	1	3	0	0	3	1	0	0	0	2	1	1	1	3	0	0
	RDMP	0	0	2	1	0	1	2	0	1	2	0	0	3	0	0	0	1	0	0	2	0	1	2	0
Intermediate OMM level	RDMP2	2	1	0	0	0	3	0	0	3	0	0	0	3	0	0	0	0	0	0	3	3	0	0	0
	STK	0	2	6	2	4	2	2	0	7	3	0	0	4	5	1	0	1	3	3	2	4	6	0	0
	PPQA	0	1	2	0	1	2	0	0	3	0	0	0	3	3	0	0	2	4	0	0	3	3	0	0
	PMC	0	1	1	6	0	8	0	0	3	5	0	0	5	2	1	0	2	3	3	0	3	4	1	0
	TST 1	1	2	1	4	0	6	2	0	5	3	0	0	6	2	0	0	0	5	3	0	6	1	1	0
	DSN 1	0	0	3	1	1	3	0	0	1	1	2	0	2	2	0	0	1	2	0	1	0	4	0	0
	PP2	0	0	2	3	3	2	0	0	4	1	0	0	3	2	0	0	2	1	2	0	2	3	0	0
Advanced OMM level	CONT	2	4	0	0	6	0	0	0	4	1	1	0	2	2	0	1	2	1	3	0	2	3	1	0
	RASM	3	1	5	0	8	1	0	0	3	6	0	0	8	1	0	0	3	4	1	1	5	4	0	0
	REP	0	0	4	1	3	1	1	0	2	3	0	0	1	3	1	0	0	2	2	0	0	2	3	0
	PI	0	0	4	5	2	3	4	0	4	4	1	0	3	5	1	0	4	3	1	1	0	6	3	0
	DSN 2	0	4	8	2	2	8	4	0	6	6	2	0	9	4	1	0	2	7	5	0	4	9	1	0
	RSKM	7	0	0	0	5	2	0	0	3	4	0	0	7	0	0	0	5	2	0	0	6	1	0	0
	TST2	0	0	2	3	0	3	2	0	1	3	1	0	4	1	0	0	0	3	1	1	0	5	0	0

6. Results

Table 1 presents the assessment results of the first Goal (PDOC 1: Provide High-Quality documentation) of the Project documentation TWE for the Mobile OS project. We present the complete results for one TWE. We have assessed the metrics ("LookFors") and the practices and calculated the goals and the TWEs. The OMM allows both a staged and a continuous assessment approach. We used the second approach where all OMM elements are assessed. We decided to assess elements from all three maturity levels to have a good understanding of basic and advanced practices. The final value for the Project documentation (PDOC) TWE of the Mobile OS project is 2,11 which means that the aspects related to product documentation are averagely addressed by the project and improvements are possible. We see that some aspects related to the project documentation are appropriately addressed by the Mobile OS project, others needs improvement. Table 2 shows the normalized assessment results for all six projects. In Table 3 we see the assessment results of practices for the assessed projects. The threshold value 4 is valid for the Mobile OS project and for the three community based projects. The Networking Monitoring and the Business Intelligence projects have a 0 for all practices in the last column (4*). Table 3 is the aggregation of the assessment results for the six projects. Since we are validating the OMM, we grouped results of all projects to be able to compare them. Figures 5, 6, and 7, and Table 3 provides an overview of which practices have to be improved in each FLOSS project. We present detailed results for each project separately in the following sections. We describe just the elements that obtained good and bad assessments for each of the six projects.

6.1. The Mobile OS Project. Figures 2, 3, and 4 present the basic, intermediate, and advanced OMM levels for the Mobile OS project. Figures 5, 6, and 7 present the percentage of practices reaching a specific-threshold assessment value (1–4). The TWEs that have a large percentage of practices assessed with a value 3 or 4 represent good coverage of TWEs by the FLOSS project. The TWEs with a large percentage

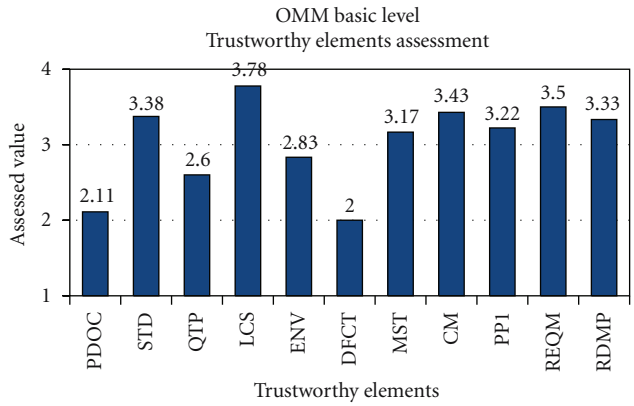

Figure 2: Mobile OS project OMM Basic level results.

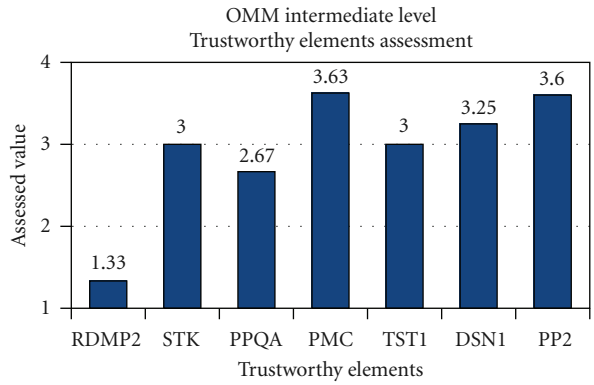

Figure 3: Mobile OS project OMM Intermediate level results.

of practices assessed with values 2 or 1 represent FLOSS development processes that should be additionally improved by the FLOSS project.

In the second column of Table 2 we see the assessment results for the mobile OS project trustworthy elements.

Basic Level. The project obtained high grades for: the aspects related to the consistent adoption and management of licenses (LCS 3,78), the management of requirements (REQM 3,5), and the configuration management (CM 3,43). Low values were obtained by: the TWEs related to commits and bug reports (DFCT 2,0), and the documentation (PDOC 2,11).

Intermediate Level. Three best assessed trustworthy elements were: the project monitoring and control (PMC 3,63), the project planing (PP2 3,6), and the design of the product (DSN1 3,25). A low value was obtained by the advanced elements of the roadmap (RDMP2 1,33).

Advanced Level. The project obtained good assessments for: the advanced testing (TST2 3,6), the product integration (PI 3,56), and the reputation of the project (REP 3,2). Poor results obtained two TWEs: the risk management (RSKM 1,0) and the contributions from external developers (CONT 1,67).

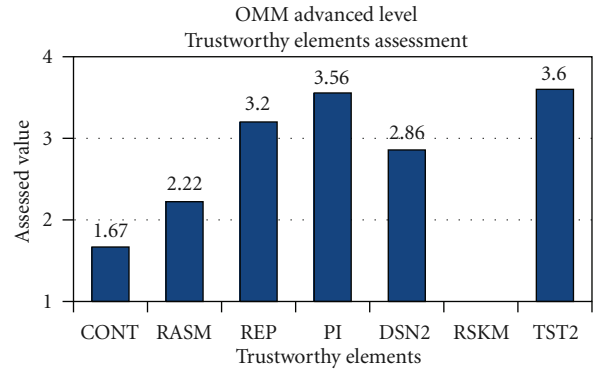

Figure 4: Mobile OS project OMM Advanced level results.

From Figures 5, 6, and 6 we see how many *practices* obtained good or poor assessments. The assessment value of a trustworthy element displayed in Table 2, and the percentage of practices obtaining a specific assessment value are related. For the Mobile OS project we see that in the:

Basic Level. There are 2 practices assessed with the value 1 only in the product documentation trustworthy element (PDOC). 6 practices in the quality of the testing plan (QTP) and 5 practices in the number of commits and bug reports (DFCT) TWE were assessed with the value 2. The highest grades were obtained for the adoption and management of aspects related to licenses (LCS) where 8/9 (8 out of 9 practices assessed in LCS) obtained the highest assessment value 4.

6.2. The Network Monitoring Project. We present the results for the other five Projects in the compact form of a table, nevertheless the graphical representation is easier to be interpreted.

In Table 2, we see the normalized results of the assessment of *trustworthy elements*. In this section, we describe the raw assessment data before the normalization.

Basic Level. Good results were obtained by: the roadmap (RDMP 2,67), the configuration management (CM 2,43), and the environment adopted (ENV 2,42). Poor results were obtained by: the standards implemented (STD 1,5) and the adoption and management of license aspects (LCS 1,22).

Intermediate Level. Only the testing trustworthy element obtained a good result (TST1 2,25) but there are three poor results: the stakeholders involvement (STK 1,4), the project planning (PP2 1,4), and the product and process quality assurance (PPQA 1,67).

Advanced Level. The advanced testing aspects were addressed appropriately (TST2 2,4) and partially also the project integration (PI 2,22). There are several aspects that are almost not addressed by the project, as: the external contributions to the project (CONT 1,0), the assessment of the project by external entities (RASM 1,11), and the risk management aspects (RSKM 1,29).

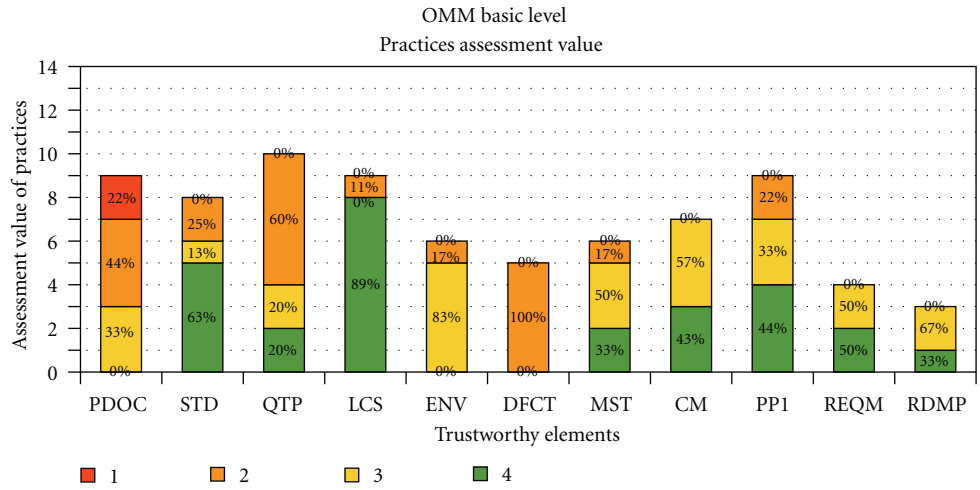

Figure 5: Mobile OS project OMM Basic level single practices threshold values.

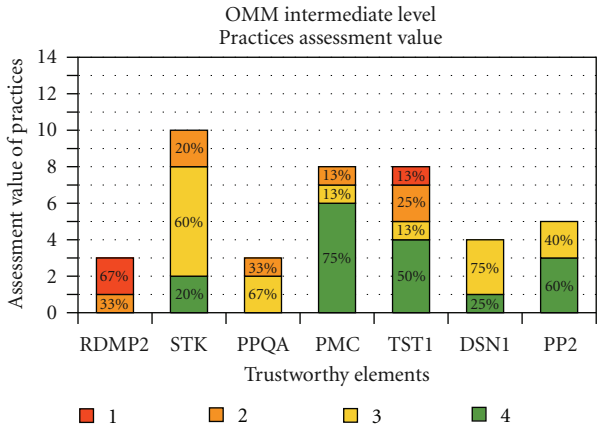

Figure 6: Mobile OS project OMM Intermediate level single practices threshold values.

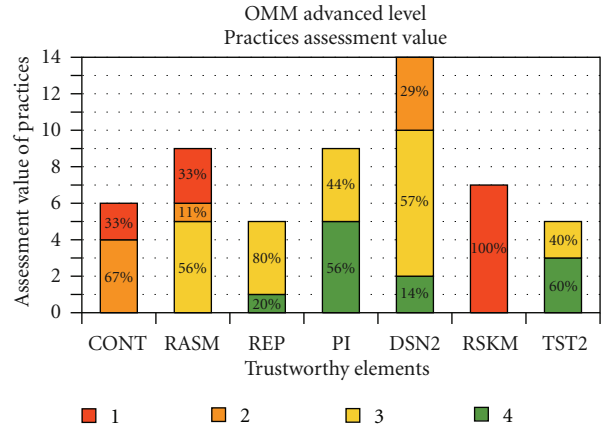

Figure 7: Mobile OS project OMM Advanced level single practices threshold values.

In Table 3, we see results related to good- and poor-assessed *practices*.

Basic Level. Configuration management (CM) with 4/7 (four out of seven) practices, and the roadmap (RDMP) with 2/3 practices obtained the highest possible value (3). There are many poor values as: the standards implemented by the project (STD) with 4/8 practices and the management of licenses (LCS) with 7/9 practices that obtained the value 1.

Intermediate Level. There is only the testing TWE (TST1) with all practices assessed with the values 2 and 3. Poor values were obtained by: the stakeholders involvement (STK) with 4/8 practices assessed with 1, the product and process quality assurance with all three practices assessed with either value 1 or 2, and the product plan TWE (PP1) with 3/5 assessed with 1.

Advanced Level. There is only the testing TWE (TST2) that fulfilled the majority of available practices. There are four TWEs that have to be improved: the contribution (CONT) practices were not addressed at all by the project 6/6 obtained 1, external assessments of the project (RASM) 8/9 practices obtained 1, for the reputation of the project (REP) 3/5 practices obtained 1, and the risk management (RSKM) was addressed only marginally with 5/7 practices with 1.

6.3. The Business Intelligence Project. Some interesting results for the Business Intelligence project are presented in the following paragraphs. First we present the results for *trustworthy elements*.

Basic Level. Aspects as management of licenses (LCS 2,67), configuration management (CM 2,57), and development environment used (ENV 2,33) obtained relatively good results. Two TWEs with low assessment values were: the quality of the testing plan defined (QTP 1,3) and maintainability and stability (MST 1,5).

Intermediate Level. The Business Intelligence project obtained only one good assessment result; the design TWE

(DSN1 2,25). Other TWEs obtained poor results; the lowest were: the advanced roadmap (RDMP2 1,0), the Product and Process quality assurance (PPQA 1,0), and the process planing (PP2 1,2).

Advanced Level. The assessment values for most of the TWEs were similar; Advanced testing (TST2 2,0) obtained the highest value, and the contribution level (CONT 1,5) obtained the lowest value.

From Table 3, we see that the Business Intelligence project reached a good fulfilment of *practices* mainly in the basic level.

Basic Level. Standards implemented (STD) 4/8 obtained the value 3, management of licenses (LCS) 6/9 with the value 3, and configuration management (CM) 5/7 with value 3. The TWEs that can be still improved are: the quality of the testing plan (QTP) 7/10 practices obtained 1, the number of commits and bug reports (DFCT) with the 4/5 values with the value 1 or 2, and the maintainability and stability (MST) with 3/6 practices assessed with 1.

Intermediate Level. Only design (DSN1) obtained a larger part 2/4 of practices assessed with 3. Other TWEs obtained poor values; 3/3 for the advanced roadmap (RDMP2), 7/10 for the stakeholders involvement (STK), and 3/3 for the product and process quality assurance (PPQA), practices were assessed with 1.

Advanced Level. Only the testing TWE (TST2) has a relatively large percentage of practices 4/5 assessed with values 2 or 3. The majority of the practices were assessed with value 1.

6.4. The Apache Community Project. The results for the Apache community project are presented in the following paragraphs. The results for *trustworthy elements* are the following:

Basic Level. The management of licenses (LCS 3,86) is almost optimal, the configuration management (CM 2,43), the development environment used (ENV 2,67), and the standards implemented (STD 3,0) obtained good results. Two TWEs with low-fourassessment values were: the requirements management (REQM 1,25) and the roadmap availability (RDMP 1,0).

Intermediate level. The project obtained only intermediate results in the second OMM level. Only the stakeholders involvement (STK 1,7) obtained some positive practices assessments. Other TWEs obtained poor results; the lowest were: the advanced roadmap (RDMP2 1,0), and the basic testing aspects (TST1 1,25).

Advanced Level. The assessment values for most of the TWEs were similar; The project's reputation obtained the best value (REP 2,0). Advanced testing (TST2 1,2), the results of third-party assessment (RASM 1,11), and the risk management (RSKM 1,0) obtained low assessment results.

From Table 3 we see that the Apache community project reached a good fulfilment of practices only in the basic level.

Basic Level. Standards implemented (STD) 3/6 obtained the value 4, and the management of licenses (LCS) 6/7 with the value 4. The TWEs that can be still improved are: the number of commits and bug reports (DFCT) with the 4/9 values with the value 1, the product documentation (PDOC) with 4/9 with the value 1, and the project planning 1 (PP 1) with 4/9 obtaining the value 1. Several other practices can be addressed to increase the basic OMM quality of the Apache community project.

Intermediate Level. Only the stakeholders involvement (STK) and the project monitoring and control have obtained a 3 for one practice. The rest of practices have been assessed either with a 1 or 2. This means that the project should still work on the improvement of the intermediate level practices.

Advanced Level. The contributions TWE (CONT) have obtained one practice assessed with the highest grade. The rest of the practices obtained poor assessments.

6.5. The SourceForge Community Project. The results for the SourceForge community project are.

Basic Level. Four TWEs obtained good results in the basic level; the number of commits and bug reports (DFCT 3,0), the maintainability and stability (MST 3,4), the configuration management (CM 3,0), and the basic roadmap (RDMP 3,0). Several other TWEs obtained relatively good assessments. A problematic aspect was only the quality of the testing process (QTP 1,6) that was not addressed sufficiently inside the assessed SourceForge project.

Intermediate Level. The advanced aspects of the roadmap (RDMP2 4,0) obtained an excellent result. A problematic aspect of the intermediate OMM level was the process and product quality assurance (PPQA 1,67).

Advanced Level. The advanced testing aspects (TST2 2,6) obtained the highest grade in the advanced level. The worst result was obtained by practices of the risk management TWE (RSKM 1,29).

From Table 3, we see that the SourceForge community project reached a good fulfilment of *practices* in the basic level and just few in the intermediate level.

Basic Level. Many practices obtained the highest grade in the basic OMM level; the Project Documentation (PDOC) 3/9, the Licenses management (LCS) 3/7, the Number of commits and bug reports (DFCT) 3/9, the Maintainability and Stability (MST) 3/5, and the Configuration Management (CM) 4/7 aspects were addressed optimally by the assessed SourceForge project. More than half of the practices of the Quality of the testing plan (QTE) 6/10 TWE obtained, on contrary, the lowest grade and should be better addressed by the project's community.

Intermediate Level. The advanced aspects of the roadmap (RDMP2) 3/3 were optimally addressed. Apart the Stakeholders involvement (STK) all other TWEs have a below-the-average assessment value of the practices and can therefore be additionally improved.

Advanced Level. In the advanced level the practices obtained poor assessment results. Just the Contributions (CONT) 3/6 obtained the assessment value of 3, and the Reputation (REP) 2/4 practices obtained the assessment value of 3 were assessed with average assessment values. The other TWEs were assessed below the average and should be further improved.

6.6. The Mozilla Community Project. Some significant results for the Mozilla community project are presented in the following paragraphs.

Basic Level. Only the Product documentation (PDOC 2,89), the environment (ENV 3,0) and the basic roadmap TWE (RDMP 2,67) obtained an above-the-average grade. All the other TWEs were not well addressed by the project. The basic aspects of product planning, project planning (PP1 1,44) obtained the lowest assessment result.

Intermediate Level. The intermediate level TWEs obtained poor assessment result. The worst assessed TWEs were the advanced roadmap aspects (RDMP2 1,0), and the basic testing (TST1 1,38).

Advanced Level. The advanced TWEs were assessed just slightly better than the intermediate TWEs. The best result was achieved by the reputation TWE (REP 2,6). Two TWEs that should be improved by the project are the third-party assessment results available on the web (RASM 1,44) and the risk management (RSKM 1,14) by the community developing the project.

From Table 3, we see that the Mozilla community project reached an above-average fulfilment of *practices* just in the basic level.

Basic Level. The product documentation (PDOC) 6/9 obtained the value 3, the Environment (ENV) 4/6 obtained the value 3, and the basic Roadmap practices (RDMP) 2/3 obtained the value 3, were assessed above the average. The other practices obtained the assessment value 1 or 2. The basic aspects of Product Planning, Project Planning (PP1) can be still much improved since 5/9 practices obtained the assessment value 1.

Intermediate Level. Just two practices of all assessed in the intermediate level obtained the value 3 and none obtained a 4. The advanced roadmap (RDMP2) 3/3 obtained the value 1, and the basic-testing practices (TST1) 6/8 obtained the value 1, should be improved by the developers.

Advanced Level. Three practices of the reputation (REP) and the Product Integration (PI) TWEs were assessed with the value 3. No practice of the advanced OMM level was assessed with a 4. The community should focus on the third-party assessment (RASM) practices where 5/9 obtained the lowest assessment value and the risk management (RSKM) TWE's practices where 6/7 obtained the assessment value 1.

7. Discussion

Based on the results we can identify aspects of the FLOSS development process that were not addressed sufficiently by the assessed projects. By aggregating all practices values for each project using (4); the Mobile OS project obtained the total value 2,88 (72%), the Networking Monitoring project obtained 1,88 (63%), the Business Intelligence project obtained 1,73 (58%), the Apache community project obtained 1,73 (43%), the SourceForge community project obtained 2,43 (61%), and the Mozilla community project obtained the aggregated assessment value of 1,95 (49%). The values in brackets present the percentage of the reached assessment value normalized by the highest possible assessment value; the percentages normalize the results for the assessments removing the differences related to the two-assessment scales used.

Critical TWEs in the Mobile OS project are the product documentation (2,11), the development environment (2,83), the number of commits and bug reports (2,00), and the testing (2,60). Those TWEs should be first improved since they are part of the Basic OMM level and as described in Section 2 they are perceived as important by a large percentage of potential FLOSS stakeholders. There are other aspects that the Mobile OS project should improve, but, since they are not part of the Basic OMM level, their improvement can be done after the Basic-level TWEs are addressed.

Comparing the six assessments we see that the Mobile OS project obtained the highest overall rating; the assessment results (taking in consideration different scale levels) are considerably higher than for the other industry led projects started later (2005 and 2006) but also higher than for the three community led projects that have started few years before the Mobile OS project. An explanation can be the importance of the assessed FLOSS project for the company. While the Mobile OS project is the key product of the first company and the main source of revenues, the other two company-led projects are just one of many projects developed by the two companies and are not critical for their business. This argument holds also for the community-led projects. The communities dedicate effort for the project but they are usually not constrained by strict deadlines and quality controls as an industry-led FLOSS project. Another reason is that the company involved in the Mobile OS project is more aware of FLOSS aspects addressed in OMM (as the reputation and the use of open standards) than the other two companies.

In all six cases there are several TWEs that obtained low grades. The lowest assessment values were obtained by the following TWEs (in brackets are the values for all the projects): External contributions to the project (1,67; 1,0; 1,5; 1,67; 2,17; 1,83), the advanced Roadmap (1,33; 2,0; 1,0; 1,0; 4,0; 1,0), and the Risk management (1,0; 1,29; 1,57; 1,0; 1,29; 1,14), and so forth. The only exception is

the advanced roadmap that was optimally addressed by the assessed SourceForge community project. Some trustworthy elements were graded better in the Mobile OS project and partially in the three community-led projects than in the other two industry projects; as for example the reputation of the project (3,2; 1,6; 1,6; 2,0; 2,0; 2,6). This result does not surprise since the reputation usually grows with a longer existence of a FLOSS project.

Differently from FLOSS focused TWEs, all three industry projects cover well aspects addressed by traditional software development TWEs as: the configuration management (3,43; 2,43; 2,57; 2,43, 3,0; 2,14), the Product Planing (3,22; 2,33; 2,11; 1,67; 2,11; 1,44), and the testing (3,6; 2,4; 2,0; 1,2; 2,6; 2,0). We can see that the community-led projects obtained lower assessment values for these TWEs. Other TWEs obtained also high values in one or two projects as: the licenses management that is appropriately addressed by the Mobile OS (3,78) and the Business intelligence (2,25) projects, and obtained a low value in the Networking Monitoring project (1,22).

It is evident from the results that standard software development process aspects regularly used in the software industry are addressed appropriately by the three industry-led FLOSS projects. These software aspects are characterized by the following TWEs: the configuration management (CM), the product plan (PP1), the basic testing (TST1), the basic design (DSN1), the advanced testing (TST2), and the advanced design (DSN2). The software development processes used in the three companies leading a FLOSS project are used also in this kind of projects. Some FLOSS specific aspects as: the external contributions to the project (CONT), the reputation of the project (REP), and the stakeholders involvement (STK) on contrary are not addressed sufficiently by the industry-led projects but are in general better addressed by the three community-led projects.

Table 3 presents the level of fulfilment of all practices for each Trustworthy element. One result of the OMM assessment process conducted is a set of improvement suggestions for the assessed projects. If we mention just the TWEs that have to be improved first by each project we can say that:

(i) the Mobile OS project should focus first on the quality of the documentation (PDOC);

(ii) the Networking Monitoring project on the management of Licenses (LCS); the Business intelligence project on the quality of the testing plan (QTP);

(iii) the Apache community project on the practices of the basic roadmap (RDMP);

(iv) the SourceForge community project on the quality of the testing plan (QTP);

(v) the Mozilla community project on the basic Product Planning, Project Planning (PP1).

According to OMM only after improving the basic level practices the projects should focus on higher level practices.

8. Threats to Validity

The main type of threats faced in our research are internal. The OMM model itself is subjected to construct validity threats: are we really measuring what the metrics in the OMM are supposed to measure. The threats described in this section are of these two types. Due to the limitation of our research, to three industry-led and to three community initiated FLOSS projects, we cannot generalize the results to all FLOSS projects, therefore we did not consider particularly the external threats to validity.

An important threat to validity of our study is the subjectivity of the assessment process. Different assessors can interpret questions in a slightly different way and therefore provide a different assessment result. This problem is intrinsic to other software assessment methodologies as the OpenBRR, the QSOS, but also to the CMMI. There are however a series of measures adopted to minimise the subjectivity of the assessment process. The most important are: the standard and homogeneous formulation of questions throughout the whole model; the extent of the development process or documentation material that have to be analysed to evaluate each question (questions are precise and limited to one or few information sources); in the assessment process participate two or more experts that should meet a common assessment result. These are three elements mitigating the subjectivity of the assessment. From experiments conducted by comparing assessment results of several individuals assessing the same process activities, it is evident that the variability of assessment results is small [6].

An additional approach to mitigate the threat to validity related to subjectivity of the assessment is the expertise of assessors. The OMM model is new and therefore none of the participating assessors have a vast experience with its usage, however they were knowledgeable with CMMI assessments that inspired many design decisions related to the OMM model. Participants of the assessments were also people involved in the development of the assessed FLOSS project. They provided support when searching the needed documentation and indicating sources of information.

The time constraints of the assessment may have limited the precise assessment of few practices that would benefit from a tool support. The OMM model is improved once per year; the major improvement efforts are focused on the creation of tools that can automatically extract information as for example: the number of license files, the number of contributors, the type and the number of documentation, and so forth. At the time of assessment of the six projects we did not use assessment tools and we assessed all metrics manually, this could present a threat to the completeness of some metrics that require a tedious measurement process. However the number of these metrics is limited and it does not considerably influence the final assessment result.

9. Conclusions

The OMM assessment results can bring benefits to FLOSS users and to FLOSS communities. The assessments can be beneficial also to potential software integrators when they are

deciding to use a FLOSS product. The final goal of the OMM is an improved quality of the FLOSS development process and therefore also the quality of the FLOSS products.

In this paper we present the results of six development process assessments of FLOSS projects that are led by three different software companies and by three different FLOSS communities. The study was performed using the OMM model analysing 25 process characteristics. The result of the assessment is the collection of values related to the FLOSS development process. The assessment results were aggregated on four levels of granularity; in this paper we present only two of them: the level of OMM Trustworthy elements presented in Table 2, and the level of OMM Practices presented in Table 3, visualizing the number of practices that were fulfilled in each threshold level.

The main result of the OMM assessment is the creation of a detailed picture of 25 aspects of the FLOSS development process. Tables and figures present a clear overview on aspects that have to be improved to arise the quality of the development process and subsequently the quality of the FLOSS product. From the figures we see which TWEs were insufficiently addressed by the assessed projects. Those TWEs were:

(i) the number of commits and bug reports (DFCT),

(ii) the external contribution to the project (CONT),

(iii) the risk management (RSKM).

Based on the results, the assessed FLOSS projects should check which practices, or more specifically, which documents are not available inside the project (in the Mobile OS project for example, a road map document should be created, and the technical documentation must be improved—see Table 1).

From the comparison of the results for the assessed projects we observed that specific FLOSS aspects obtained lower assessment results than standard software development practices. We can also see that the project that exists longer, both the industry-led and the FLOSS communities led, obtained better results related to the aspect of reputation. The Mobile OS project generically obtained better results from the other assessed projects; it appropriately implemented 72% of the 122 practices.

The results for the six studies demonstrate the applicability of the OMM and describe the benefits that FLOSS projects can have from the assessment. The validity of the research results is limited to the six studies; however the assessment process of a FLOSS project, either industry or pure FLOSS communities supported, would be conducted following a similar approach. The results of the study will bring concrete benefits to the quality of the three industry-led FLOSS development processes and it can suggest improvement activities necessary for the three community-led projects. The results of the first assessment are good indicators of which practices should be improved.

We plan to conduct a second set of assessments of the six projects after one year from the first assessment and compare the improvements of the development process. In the following months we plan to perform other assessments of well-known FLOSS projects. Our aim is to use the model to assess a larger number of FLOSS projects where the leading role is maintained by the FLOSS community. We plan to compare those results with the one presented in this paper and further confirm the validity of the OMM model.

Acknowledgments

The research was conducted in the scope of the QualiPSo project (FP-IST-034763). The authors are grateful to all QualiPSo partners that contributed to the creation of the assessment model and the participants from the assessed FLOSS projects.

References

[1] E. S. Raymond, *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, O'Reilly & Associates, 2001.

[2] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: a systematic review," *Information and Software Technology*, vol. 50, no. 9-10, pp. 833–859, 2008.

[3] A. Fuggetta, "Software process: a roadmap," in *Proceedings of the Conference on the Future of Software Engineering (ICSE '00)*, pp. 25–34, ACM, Limerick, Ireland, June 2000.

[4] *Process Maturity Profile of the Software Community 1999 Year End Update*, Software Engineering Institute, 2000.

[5] E. Petrinja, R. Nambakam, and A. Sillitti, "Introducing the opensource maturity model," in *Proceedings of the ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development (FLOSS '09) collocated with 31st International Conference on Software Engineering*, pp. 37–41, Vancouver, Canada, May 2009.

[6] E. Petrinja, A. Sillitti, and G. Succi, "Comparing OpenBRR, QSOS, and OMM assessment models," in *Proceedings of the 6th International Conference on Open Source Systems (OSS '10)*, pp. 224–238, Notre Dame, Ind, USA, May 2010.

[7] Qualipso Consortium: QualiPSo—Quality Platform for Open Source Software, http://www.qualipso.org/index.php.

[8] E. Petrinja, A. Sillitti, and G. Succi, "Overview on trust in large FLOSS communities," in *Proceedings of the 4th International Conference on Open Source Systems (OSS '08)*, pp. 47–56, Milan, Italy, 2008.

[9] U. Raja and M. J. Tretter, "Defining and evaluating a measure of open source project survivability," *IEEE Transactions on Software Engineering*, vol. 38, no. 1, pp. 163–174, 2012.

[10] M. Khalifa and J. M. Verner, "Drivers for software development method usage," *IEEE Transactions on Engineering Management*, vol. 47, no. 3, pp. 360–369, 2000.

[11] S. Matook and M. Indulska, "Improving the quality of process reference models: a quality function deployment-based approach," *Decision Support Systems*, vol. 47, no. 1, pp. 60–71, 2009.

[12] T. L. Roberts, M. L. Gibson, K. T. Fields, and R. Kelly Rainer, "Factors that impact implementing a system development methodology," *IEEE Transactions on Software Engineering*, vol. 24, no. 8, pp. 640–649, 1998.

[13] C. G. Von Wangenheim, J. C. R. Hauck, A. Zoucas, C. F. Salviano, F. McCaffery, and F. Shull, "Creating software process capability/maturity models," *IEEE Software*, vol. 27, no. 4, pp. 92–94, 2010.

[14] S. C. Misra, V. Kumar, and U. Kumar, "Identifying some important success factors in adopting agile software development practices," *Journal of Systems and Software*, vol. 82, no. 11, pp. 1869–1890, 2009.

[15] E. Yourdon, "Where's the basis for year 2000 optimism?" *Computerworld*, vol. 32, no. 7, p. 68, 1998.

[16] M. Agrawal and K. Chari, "Software effort, quality, and cycle time: a study of CMM level 5 projects," *IEEE Transactions on Software Engineering*, vol. 33, no. 3, pp. 145–156, 2007.

[17] M. Staples, M. Niazi, R. Jeffery, A. Abrahams, P. Byatt, and R. Murphy, "An exploratory study of why organizations do not adopt CMMI," *Journal of Systems and Software*, vol. 80, no. 6, pp. 883–895, 2007.

[18] F. Guerrero and Y. Eterovic, "Adopting the SW-CMM in a small IT organization," *IEEE Software*, vol. 21, no. 4, pp. 29–35, 2004.

[19] A. Qumer and B. Henderson-Sellers, "A framework to support the evaluation, adoption and improvement of agile methods in practice," *Journal of Systems and Software*, vol. 81, no. 11, pp. 1899–1919, 2008.

[20] M. C. Paulk, "Extreme programming from a CMM perspective," *IEEE Software*, vol. 18, no. 6, pp. 19–26, 2001.

[21] C. G. von Wangeheim, A. Anacleto, and C. F. Salviano, "Helping small companies assess software processes," *IEEE Software*, vol. 23, no. 1, pp. 91–98, 2006.

[22] T. Dybå, "Factors of software process improvement success in small and large organizations: An empirical study in the scandinavian context," in *Proceedings of the 9th European Software Engineering Conference Held Jointly with 11th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pp. 148–157, ACM Press, September 2003.

[23] G. B. Dietrich, D. B. Walz, and J. L. Wynekoop, "The failure of SDT diffusion: a case for mass customization," *IEEE Transactions on Engineering Management*, vol. 44, no. 4, pp. 390–398, 1997.

[24] Navica Inc., The Open Source Maturity Model is a vital tool for planning open source success, http://www.oss-watch.ac.uk/resources/osmm.xml#body.1_div.2.

[25] Atos Origin, Method for Qualification and Selection of Open Source Software (QSOS), 2009, http://www.qsos.org/.

[26] A. Wasserman, M. Pal, and C. Chan, *Business Readiness Rating Project*, BRR Whitepaper 2005 RFC1, http://www.openbrr.org/.

[27] D. Taibi, L. Lavazza, and S. Morasca, *OpenBQR: A Framework for the Assessment of OSS*, Open Source Software 2007, Limerick, Ireland, 2007.

[28] D. Izquierdo-Cortazar, G. Robles, J. M. González-Barahona, and J.-C. Deprez, "Assessing FLOSS communities: an experience report from the QualOSS project," *Open Source Ecosystems: Diverse Communities Interacting*, vol. 299, p. 364, 2009.

[29] I. Samoladas, G. Gousios, D. Spinellis, and I. Stamelos, "The SQO-OSS quality model: Measurement based open source software evaluation," *IFIP International Federation for Information Processing*, vol. 275, pp. 237–248, 2008.

[30] J.-C. Deprez and S. Alexandre, *Comparing Assessment Methodologies for Free/Open Source Software: OpenBRR and QSOS*, Book chapter in Lecture Notes in Computer Science, Springer, Berlin, Germany, 2008.

[31] V. R. Basili, "Software modelling and measurement: the Goal/Question/Metric paradigm," Computer Science Technical Report Series CS-TR-2956 (UMIACS-TR-92-96), University of Maryland, College Park, Md, USA, 1992.

[32] S. Morasca, "On the use of weighted sums in the definition of measures," in *Proceedings of the 2010 ICSE Workshop on Emerging Trends in Software Metrics (WETSoM '10)*, pp. 8–15, ACM Press, May 2010.