UNIVERSIDADE DE SÃO PAULO

Qualipso Project: Quality Recommendations for FLOSS development processes

A perspective based on trustworthy elements

Viviane Malheiros, Erika Höhn, José Carlos Maldonado

RT-335

Relatórios Técnicos do ICMC
São Carlos
Fevereiro/2009

Abstract

This technical report focuses on the proposal of Quality Recommendations for FLOSS (Free and open source software) communities. Quality Recommendations are being suggested in the context of the Qualipso Project, particularly in the context of the Open Source Maturity Model (OMM). OMM is a CMMI-like model for FLOSS that can be implemented in software organizations to enable FLOSS usage both in production and development of software products. These Quality Recommendations will set basis to establish OMM practices and are connected with OMM processes areas. Such Quality Recommendations, as the OMM, are expected to add trustworthiness to FLOSS development processes and products. The perspective based on trustworthy elements is aligned with OMM approach and may enable consistent savings and reducing the time to market.

Table of Contents

1. Introduction

This technical report focuses on the proposal of Quality Recommendations for FLOSS (Free/ Libre Open Source Software) development process. Quality Recommendations are being suggested in the context of the Qualipso Project (www.qualipso.org).

Qualipso is an integrated project that aims to define and implement technologies, procedures and policies to leverage the Open Source Software development current practices to sound and well recognized and established industrial operations [1]. Particularly, such Quality Recommendations objective is to contribute to the Open Source Maturity Model (OMM). OMM is a CMMI-like[1] model for FLOSS that can be implemented in software organizations to enable FLOSS usage both in production and development of software products.

These Quality Recommendations will set basis to establish OMM practices and are connected with OMM processes areas. The OMM model is currently under development and proposing Quality Recommendations is a major step of its construction. They are intended to be attached to the model working document. Such Quality Recommendations, as the OMM, are expected to add trustworthiness to FLOSS development processes. The perspective based on trustworthy elements is aligned with OMM approach and may enable consistent savings and reducing the time to market.

The next section briefly contextualizes Open Source Development and summarizes the Qualipso Project, particularly referring to the trustworthy elements of the trustworthy process and the actual stage of the OMM model. Section 3 describes how Quality Recommendations are inserted in the OMM

[1] Capability Maturity Model® Integration (CMMI) is a process improvement approach that provides organizations with the essential elements of effective processes. Information about the model is available at http://www.sei.cmu.edu/cmmi/general/ .

development and presents our proposed Quality Recommendations to FLOSS development process. Finally Section 4 discusses some final remarks.

2. Background

2.1 Open Source Development

Open source software (OSS) is defined as computer software for which the source code and certain other rights normally reserved for copyright holders are provided under a software license that meets the Open Source Definition or that is in the public domain. This permits users to use, change, and improve the software, and to redistribute it in modified or unmodified forms. It is very often developed in a public, collaborative manner.

Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software [6]:

- The freedom to run the program, for any purpose (freedom 0);
- The freedom to study how the program works, and adapt it to your needs (freedom 1);
- The freedom to redistribute copies so you can help your neighbor (freedom 2); and
- The freedom to improve the program, and release your improvements (and modified versions in general) to the public, so that the whole community benefits (freedom 3).

Researches on FLOSS (Free/Libre Open Source Software) are extensive. Since the sources for research are often available for free on the web, many different research studies have been done. In the document Analysis of Free/Libre Open Source Software processes [2], Qualipso Project contributors have provided an overview of major studies done in the area.

2.2 Qualipso Project

The goal of the QualiPSo integrated project is to define and implement technologies, procedures and policies to leverage the Open Source Software (OSS) development current practices to sound and well recognized and established industrial operations [1]. The project brings together software companies, application solution developers and research institutions and will be driven by the need for having for OSS software the appropriated level of trust which makes OSS development an industrial and wide accepted practice (www.qualipso.org).

The project is structured into two classes of activities: Problem activities and Project activities, as shown in Figure 1. Problem activities provide the foundation and technological content upon which the project is built. Project activities are cross-cutting activities that take the results generated by the problem activities, integrate them in a coherent framework and assess and improve their applicability using the selected application scenarios.

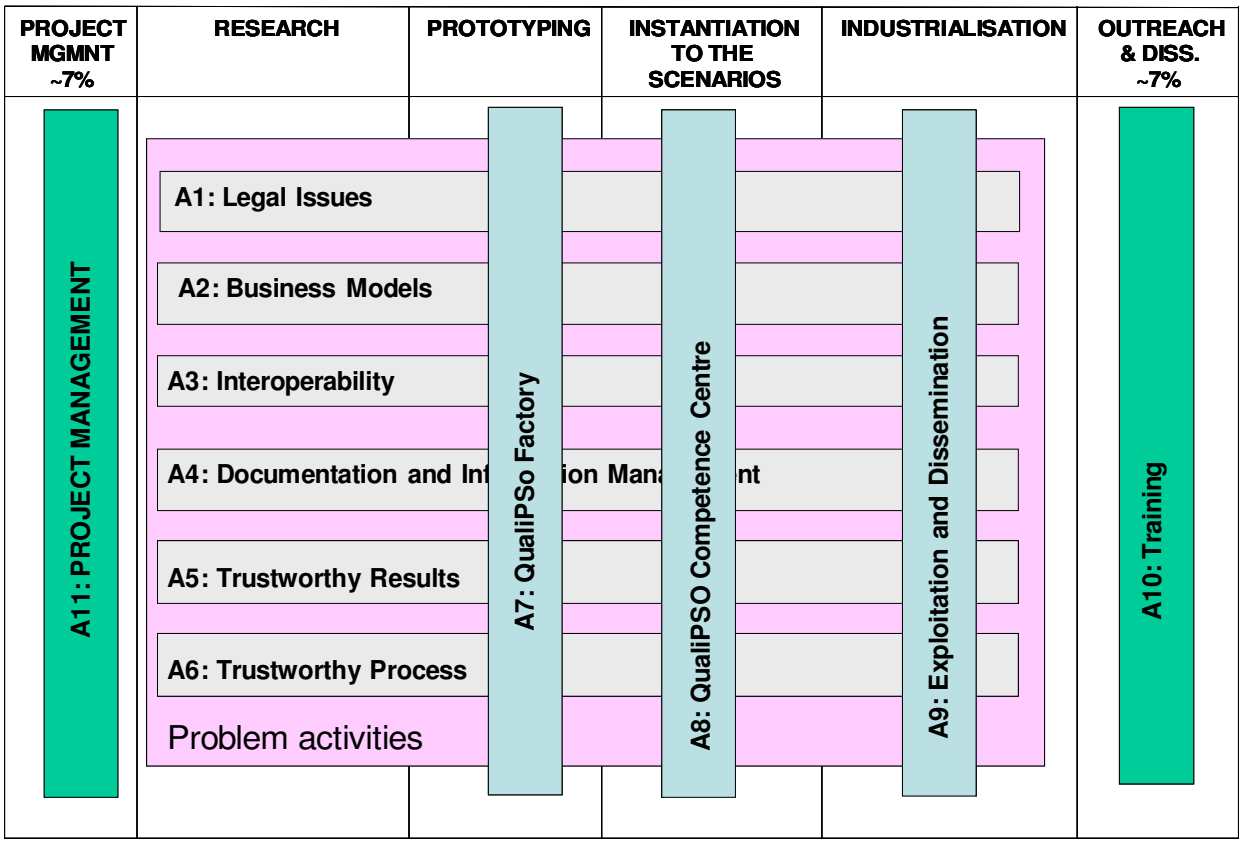

Figure 1: QualiPSo project structure [1]

The Quality Recommendations for FLOSS development processes presented here were developed in the context of the activity **A6: Trustworthy Process** (gray box in Figure 1). This activity aims to define a CMMI-like model for OSS through the identification of the factors that affect trust in Open Source development and the definition of specifications for implementing trustworthy Open Source processes in software companies through the collaborative environment developed in this project. The activity intend to: (i) research different Open Source processes adopted by the OS Community and by the industrial partners of Qualipso project; and (ii) propose a CMM-like model for the implementation of trustworthy OS processes.

Several working documents and deliverables register Qualipso researchers work and provide a detailed overview of the most important studies done. In this section we just briefly summarize the topics influencing strongly the proposed quality recommendations.

The preliminary version of the CMM-like model is resumed in Section 2.2.2. This summary was based on the 6.3 working document [7]. The trustworthy elements, identified through a survey answered by the industrial partners of Qualipso project ([3] e [4]), are available in Section 2.2.1.

2.2.1 Trustworthy Elements

In the context of the Qualipso project, an extensive research was conducted to better understand the importance of FLOSS development process and to identify key trustworthy elements that might contribute to the quality of a software product.

The Qualipso Project defines the *trustworthiness*, as a specific component or aspect of a software product that influences the belief and trust of the stakeholders in the overall quality of the software product [2]. A *trustworthy element* is as a specific factor or aspect of the software development process, or of product results that indirectly influence the perception of the trustworthiness of the FLOSS development process [4].

Two different scenarios were considered when identifying the trustworthy elements: software integrators and development communities. Surveys were applied to FLOSS communities and companies associated to the Qualipso project, which would contribute to external FLOSS projects. Details on surveys results can be obtained in QualiPSo: A6 WP6.2 wd6.2.1. In total, twelve elements were identified. The order of the elements listed above, resemble the frequency that the interviewee mentioned them ([2], [3], [4]):

1. **Product Documentation** (PDOC) – Potential integrators of FLOSS products want the product documentation to be exhaustive and easy to understand. A good documentation may also facilitate community involvement, as people would know how they can contribute. By product documentation they refer to: Product design / architecture documented (developer documentation), User documentation, and Technical documentation (for troubleshooting). Documentation must be up to date because unskilled users may encounter serious problems if the documentation is not appropriate for a specific version of the product and therefore they can stop using the FLOSS product;
2. **Popularity of the SW Product** (REP) – The more popular the software product is the more likely people will trust on it. Such popularity can be indicated by, for instance, the number of users that have downloaded the product and that are using it. Discussions in mailing lists, forums, bug reporting systems and other communication environments are also relevant to indicate the popularity of a FLOSS product;
3. **Use of Established and Widespread Standards** (STD) – Standards used are relevant for the FLOSS product. FLOSS developer communities tend to value open product standards such as http, WSDL, SOAP and process standards such as RUP. It is only natural that FLOSS integrators are positively inclined towards well established standards, which is correlated with product quality;
4. **Availability and Use of a (product) Roadmap** (RDMP) – the appearance of this element is directly related to the relevance FLOSS communities and surveyed companies attribute to process in the FLOSS context. Important aspects are, among others: responsibility for the roadmap is defined, roadmap includes plans for at least the next 2 versions, and roadmap is regularly updated. The availability and the use of a roadmap is an important trust element. It provides an insight not only in the development process followed in the past but it also

describes the improvements that are planned for the near future. The roadmap must be detailed enough and it has to be respected in order to ascertain a high quality level of the development process;

5. **Quality of Test Plan** (QTP) – Testing software appears as a relevant activity to increase the trust in a FLOSS product, particularly the quality of test plan. This includes not only the schedule for test but also planning required resources, the order in which tests will be carried out, tools to be used, test environment, testing responsibilities, how test results will be analyzed, defects corrected and open issues handled. Testing is an important part of classic software development and is often underestimated in the FLOSS process. Successful FLOSS processes and communities however conduct extensive tests on their products;
6. **Relationship between Stakeholders** (Users, Developers etc) (STK) - This trustworthy element refers more to the quality and degree of collaboration between developer communities and users than to formal sharing of responsibilities between stakeholders. It is important to know if there is good collaboration within the groups and how they communicate;
7. **Licenses** (LCS) – This trustworthy element refers to the ability of a FLOSS product to selecting and managing license properly. For instance, it is important that the product does not contain any commercial components. The decision about which license will be used is a critical decision that has to be taken in the beginning of the FLOSS development process. Licenses are even more important for companies that would like to integrate the FLOSS product with other their products;
8. **Technical Environment** (Tools, OS, Programming Language, Dev Environment.) (ENV) – Tools, operating systems, programming languages and environments used by the FLOSS products developers are important factors influencing the trust surveyed companies have in

the FLOSS process. Probably these elements have one of the most important impacts specifically on the development process. Before adopting an FLOSS product, integrators would like to know details of operating systems, tools, languages and environment in order to check if the technical environment of a FLOSS product is compatible with integrator needs;

9. **Number of Commits and Bug Reports** (DFCT) – The number of commits and the number of bug reports are also considered important for the evaluation of the FLOSS process. They are indicators of FLOSS product popularity. It also may indicate that the product is being actively developed and supported, and that further change requests and bug reports will be undertaken;
10. **Maintainability and Stability** (MST) – A potential integrator is more likely to view a FLOSS product favorably if it is proven to be maintainable and stable;
11. **Contribution to FLOSS Product from SW Companies** (CONT) – For a potential integrator, participation of reputed software or IT companies in the FLOSS development may be a positive indication of the FLOSS product. It may serve as a sign of quality for the FLOSS product; and
12. **Results of Assessment of the Product by 3rd Party Companies** (RASM) – Assessment of the product by 3rd party companies may count in favor of the product, when potential integrators evaluate FLOSS products for use in their own development.

2.2.2 Open Source Maturity Model

This section consolidates an overview of the current version of the CMMI-like model for Open Source. It's temporally name is OMM (Open Source Maturity Model). Details on it can be found at: http://www.qualipso.org/sites/default/files/A6.D1.6.3CMM-LIKEMODELFOROSS.pdf .

Like CMMI [5], OMM is organized in levels, each level building on and including the Trustworthy Elements at the lower level. OMM levels are: Basic, Intermediate and Advanced. The Trustworthy Elements included in OMM are from two different sources, 1) CMMI Process Areas and 2) FLOSS Trustworthy Elements gathered from the survey of QualiPSo work package 6.1 ([2]):

- The Trustworthy Elements at the basic level are essential for developing and delivering a trustworthy (high quality) FLOSS component. These include basic product documentation, development standards, test plan, license, technical environment, defect management (configuration management) and maintainability and stability.
- Trustworthy Elements at the intermediate level include reputation; roadmap; contributions from companies; stakeholder involvement; and results of assessment from third parties. These TWEs require that the TWEs from the basic level are fulfilled (prerequisites).
- Trustworthy Elements at the advanced level include the TWEs from the basic and intermediate levels. In addition, there are TWEs unique to the advanced level.

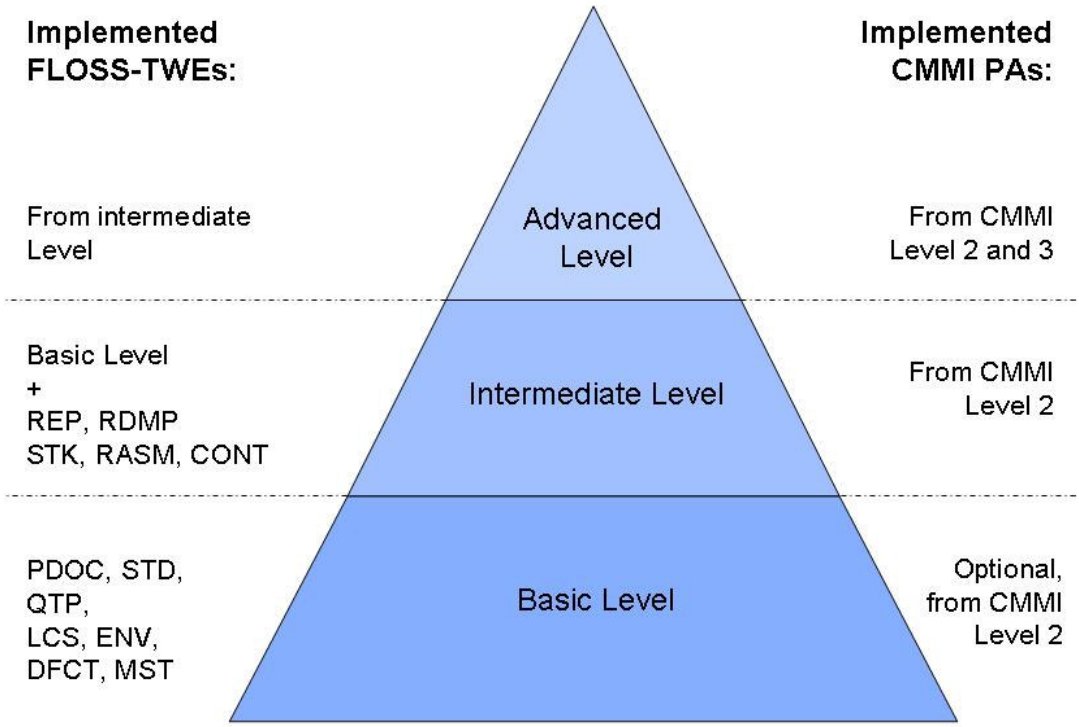

Figure 2: Trustworthy Elements in OMM [7]

3. Quality Recommendations for FLOSS development processes

Towards defining a complete OMM model, identifying and organizing Quality Recommendations for FLOSS development processes are intermediate steps, as illustrated in Figure 3 (see highlighted box).

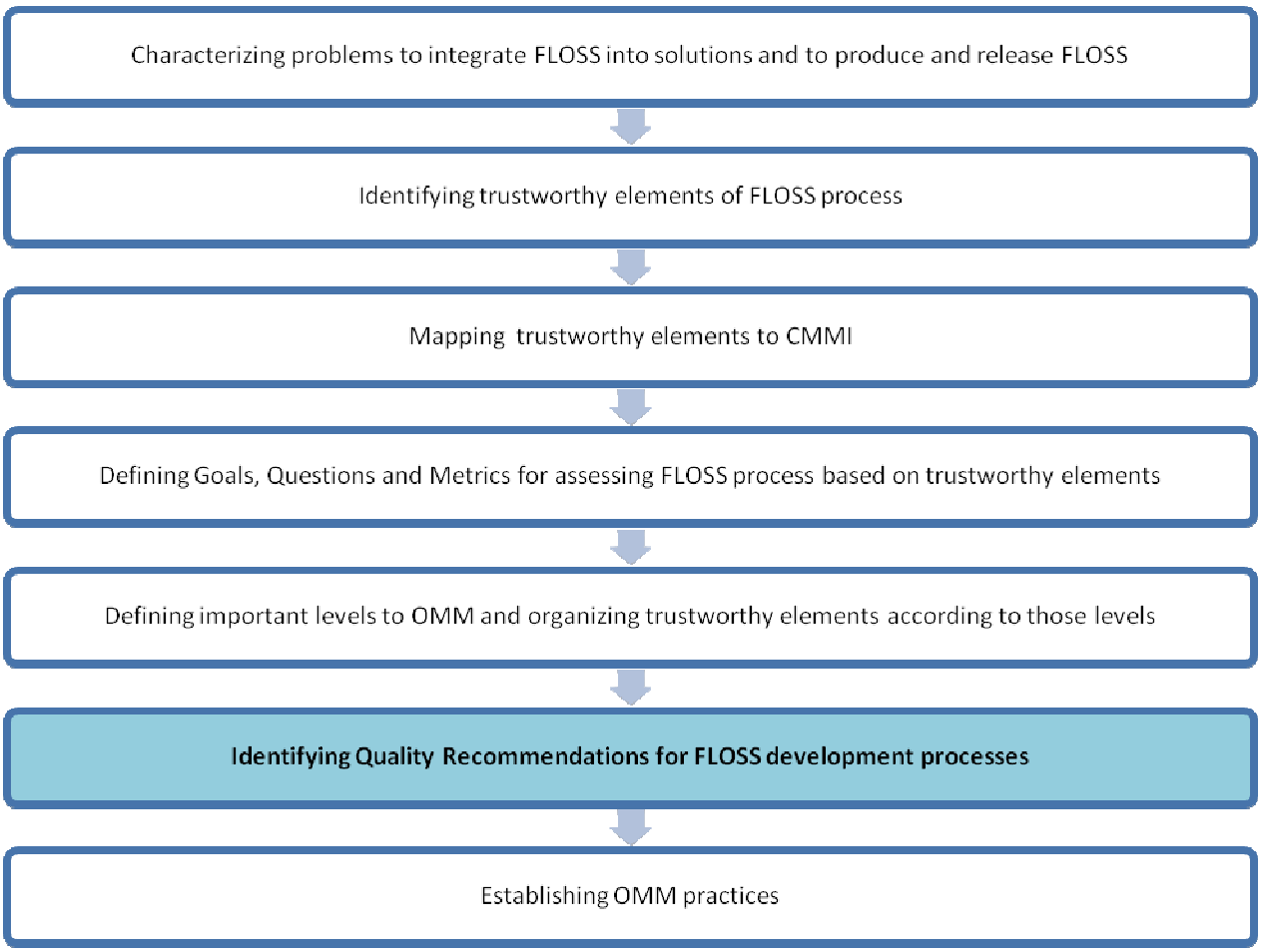

Figure 3: Steps for defining OMM

The first step was to identify typical problems faced by the Industry when integrating FLOSS into products and solutions and when producing and releasing FLOSS. Typical issues that every integrator should be aware of are described in [2]. Yet, pursing FLOSS project characterization, the second step was to identify trustworthy elements (Section 2.1.1).

Once all trustworthy elements were identified, the third step was to map trustworthy elements to CMMI, comparing such elements with CMMI practices and finding out what would be necessary to keep from CMMI model and what was missing. Trustworthy elements were compared to the various process areas in CMMI, and to Specific Practices in each case. The rationale behind such analysis was: (i)

CMMI is a well-known maturity model that can be used as a reference for software development; however (ii) FLOSS development presents peculiarities that may justify its own maturity model. This mapping intended to set basis to the OMM definition.

The next step was to describe first ideas on which process related metrics could be used to evaluate FLOSS; and to assess the adoption of the OMM. Those first ideas were structure according to GQM [8], in other to emphasize the connection between metrics as goals. Two different points of view were considered: (i) Goal, question, metric (GQM) for system integrators and (ii) Goals, Questions and Metrics applicable to FLOSS development communities. This step should end up in a well defined set of metrics suitable to support the OMM. It was also a preliminary step to defining Quality Recommendations for FLOSS development processes. The trustworthy elements are the foundation for the goals defined. The goals defined offer an improvement guideline for trustworthy elements identified.

The fifth step was to define major levels of OMM to distinguish different stages of FLOSS development projects (Section 2.2.2). OMM model aims at helping organizations on managing development processes including the usage and the development of FLOSS products components or solutions.

During the sixth step (Identifying Quality Recommendations for FLOSS development processes), every goal, question and metric was considered in order to support the abstraction of Quality Recommendations FLOSS development processes would be important in order to increase the trustworthiness of a FLOSS project.

Such Quality Recommendations may be further detailed and better structured into formal OMM practices (seventh step). The mapping between Quality Recommendations to Trustworthy Elements and

OMM is presented in Section 3.1. The description of each Quality Recommendations is presented in Section 3.2. Section 3.3 shows how one quality recommendation may influence others.

3.1 Mapping Quality Recommendations X Trustworthy Elements and OMM

The mapping between Quality Recommendations and Trustworthy elements and OMM process areas is presented in Table 1. It was an iterative definition.

Table 1: Mapping Quality Recommendations to Trustworthy Elements

	PDOC	REP	STD	RDMP	QTP	STK	LCS	ENV	DCFT	MST	CONT	RASM
Communication		X				X						
Community Involvement		X			X						X	
Development Process			X	X								
Continuous Project Improvement									X	X		
Quality of FLOSS product					X				X	X		
License							X					
Integrated Development Environment								X				
FLOSS Standards			X									
Software product and project assessment												X
Configuration Management												
Project Documentation	X											
Resource Management						X						

When detailing the Quality Recommendations according to the Goals, Questions and Metrics, we identified some improvement opportunities to the OMM. Those suggestions were sent to the A6

group for evaluation. Major suggestions are illustrated (see red items) in Figure 4. Also, the proposition of distributing quality recommendations according to OMM levels is illustrated in Figure 4.

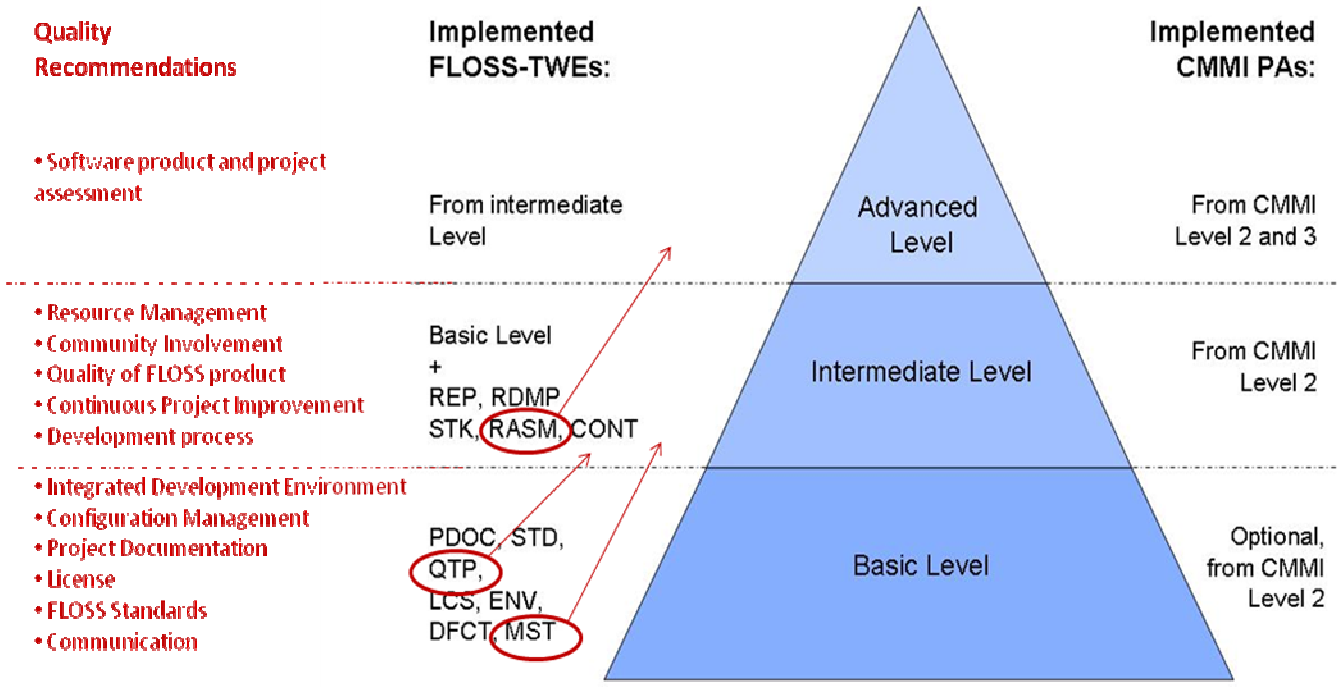

Figure 4: Mapping Quality Recommendations to OMM. Adapted from [7]

3.2 Quality Recommendations for FLOSS development processes

Each recommendation is composed by: (i) identification; (ii) recommendation main goal; (iii) description and justification on why the recommendation apply; and (iv) recommended actions that may contribute to achieving recommendation goal; (v) main trustworthy element related to the recommendation, if it is the case.

3.2.1 RECOMMENDATION 0: Communication

R0 – Communication

Communication is essential for Open Source Projects. Promote communication inside the community

- Establish and maintain a shared vision of the project. A shared vision facilitates people working together, helps those people to attain unity of purpose, and creates a common understanding of the end state the organization is aiming to achieve. It continues to evolve as more ideas are shared. Publishing summary, latest news, purpose and goals are means of sharing project vision.
- Maintain the archives of past communication, allowing users to see past communications inside the community.
- Provide support for adequate internationalization within the development process. Provide adequate support for translators, adaptors etc.
- Maintain and improve communications channels such as mailing lists, forums, trackers.
- Monitor the level of communication inside the community, which communication channels exist and what is the traffic. Monitor user satisfaction to answers in mailing lists and forums.
- Encourage suggestion/ request from the community and provide feedback about the integration of these suggestions/ requests.
- Promote the FLOSS project, aiming the adoption of the FLOSS product inside big organizations and companies.
- Evidence monitor results and rewarding system.

Related trustworthy element: Popularity of the SW product (REP), Relationship between stakeholders (users, developers etc) (STK).

3.2.2 RECOMMENDATION 1: Community involvement

R1 – Community involvement

Monitor community involvement and make evident project activity. Once the project is running the community will interact in different ways with it. The involvement must be monitored to ensure appropriate levels of popularity and support for the community, as well as project activity. Typical indicators of project activity and community involvement are downloads evolution; subscribers in mailing list evolution; numbers of contributors evolution; and number of commits in a given period. The more people perceive community interaction, the more popular a FLOSS project will be. Promoting communication inside the community will the community may favor community involvement.

- Evidence community interaction with the project. Log new features suggestions, bug reports, support requests and answers, mailing lists activities, etc.
- Review periodically community involvement and project activity to ensure the appropriate interactions are occurring.
- Promote the increasing of users and users' commitment to the community.
- Monitor if level of commitment to the project tasks continuously grow and incentive users to progressively get involved in the FLOSS development process. For instance, monitor the progress of active developers turning into core developers.
- Monitor actual performance against historical data and established objectives.
- Identify significant issues and their impact (such as significant reduction on the number of submission to the mailing list). Analyze issues to determine need for corrective action.
- Encourage community to contribute through source code, documentation, suggestion, bug reports, translation, etc.
- Monitor the project coverage on media (blogs, magazines, forums, mass media).
- Establish a rewarding system related to contributions and their usefulness. For instance promoting developers from active developer to core developer based on useful responses and contributions can be a good practice.

- Follow a well-defined strategy to attract new users.

Related trustworthy element: Popularity of the SW product (REP), Contribution to FLOSS Product from Software Companies (CONT), Relationship between stakeholders (users, developers etc) (STK).

3.2.3 RECOMMENDATION 2: Use of well-known FLOSS standards

R2 – Use of well-known FLOSS standards

Adopt, maintain and evaluate the implementation of FLOSS Standards. Based on the Open Standards, establish and maintain criteria for the evaluations. Use the criteria to evaluate performed process for adherence to standards. Identify noncompliance found during evaluation. Interoperability can have important consequences on projects successful and usage. Particularly, be aware of interoperability. It is important to assure the project capability of exchanging data via a common set of standards, to the same file formats and protocols.

- Specify and document the project's set of standards, ensuring that it adheres to well-known FLOSS standards, particularly standards for interoperability.
- Define guidelines on how to use project's set of standards and make them readily available to the community.
- Enforce the usage of the standards on project evolution.
- Conduct periodic reviews of the effectiveness and suitability of the established standards. Objectively evaluate adherence of the project against its standards, and address noncompliance.
- Keep track on changes on FLOSS standards and evolve project guidelines accordingly.
- Monitor project's interoperability of other projects and the progress of activity developers turning into core developers.
- Identify lessons learned and suggestions that could improve the Open Standards, feeding back the community.

Related trustworthy element: Use of established and widespread standards (STD)

3.2.4 RECOMMENDATION 3: Project Documentation

R3 – Project Documentation

Develop and maintain project documentation, making it readily accessible to the community. Potential integrators of FLOSS products want the product documentation to be available, easy to understand and consistent. A good documentation may also facilitate community involvement, as people would know how they can contribute. By product documentation they refer to: Product design / architecture documented (developer documentation); User documentation; and Technical documentation (for troubleshooting).

- Provide high quality documentation considering which level of information is appropriated to different stakeholders.
- Make this documentation available considering different interests, abilities, languages and skills. Ensure the availability of the documentation, for instance, providing automatic full text indexing of software documentation or a roadmap for the documentation process.
- Keep the documentation updated. Unskilled users may encounter problems if the documentation is not appropriate for a specific version of the product.
- Institutionalize an integrated documentation system to improve the process of documentation writing.
- Evaluate the quality of the documentation by checking a list of components that the documentation must have; monitoring indicators and getting community feedback.
- Provide specific support for different stakeholders, for instance, adequate support for final users/ integrators.

Related trustworthy element: Product Documentation (PDOC)

3.2.5 RECOMMENDATION 4: Integrated Development Environment

R4 –Integrated Development Environment

Establish an Integrated Development Environment. An integrated development environment help people communicating clearly and efficiently about the product, processes, people needs, and project organization. The careful selection and maintenance of integrated tools and software architecture to

develop a FLOSS project may contribute to produce better quality product and to support a good quality production process.

- Establish and implement an integrated development environment to the FLOSS project. Define in advance development strategies and development tools and programming languages. Ensure that programming languages, methodologies and tools defined are FLOSS compliant.
- Provide ongoing maintenance and operational support for the integrated development environment.
- Evaluate regularly the effectiveness of the existing environment and forecast the need for additional, upgraded, or new tools or integrated work environment components.
- Maintain awareness of current and emerging technologies, tools, and resources that are related to the integrated development environment.
- Monitor and evaluate the adequacy of the integrated development environment to satisfy user needs.

Related trustworthy element: Tech environment (tools, OS, Programming language, Dev Environment.) (ENV)

3.2.6 RECOMMENDATION 5: Quality of FLOSS product

R5 – Quality of FLOSS product

Evaluate the quality of the FLOSS product. Particularly, provide and evaluate the stability and security of the FLOSS product within the development process.

- Integrate test activities incrementally throughout the development process.
- Plan, establish and maintain the validation environment, procedures and criteria. The environments used for product integration (R4) may be considered in collaboration with the validation environment to reduce cost and improve efficiency or productivity.

- Validate product or product components. The validation methods, procedures, and criteria are used to validate the selected products and product components and any associated maintenance, training, and support services using the appropriate validation environment.
- Based on the established validation criteria, identify products and product components that do not perform suitably in their intended operating environments, or identify problems with the methods, criteria, and/or environment.
- Define integration of security and stability tests within the development process, considering the need of preparing for validating and evaluating validation.
- Record the results of the analysis and identify issues.
- Use validation results and feedback from these tests implemented back to the development process.

Related trustworthy element: Quality of Test Plan (QTP), Maintainability and Stability (MST), No. of commits and bug reports (DFCT)

3.2.7 RECOMMENDATION 6: Continuous Project Improvement

R6 – Continuous Project Improvement

Enhance the software project continuously. Enhancing a software project involves enhancing the project product and process. Improvement opportunities for the project should be identified periodically and as needed.

- Determine and evidence software project improvement opportunities, such as feature requests, bug reports or recurrently support request.
- Collect project performance indicators. Typical indicators are evolution ratio of modification requests submitted, evolution ratio of feature requests submitted by users/developers, evolution ratio bugs/issues submitted to the project.

- Monitor and enhance project performance based on indicators.
- Adopt and implement requests/ suggestions from users/ integrators and developers for both software product and process.

Related trustworthy element: No. of commits and bug reports (DFCT), Maintainability and Stability (MST)

3.2.8 RECOMMENDATION 7: Configuration Management

R7 – Configuration Management

Establish baselines and release timely. A baseline is a set of specifications or work products that has been formally reviewed and agreed upon. Afterward, the baseline serves as the basis for further development, and that can be changed only through change control procedures. Baselines provide a stable basis for continuing evolution of a software project and its items. Inclusion and identification of new code to be included in future versions may affect the quality of the FLOSS project, thus managing the source code is important.

- Establish and maintain a configuration management and change management system for controlling work products, including documentation. Place designated work products under appropriated level of configuration management, establishing and maintaining the designated work product integrity throughout their useful life.
- Establish baselines for delivery to the community. A set of feature request, design, source code files and the associated executable code, build files, and user documentation (associated entities) that have been assigned a unique identifier can be considered to be a baseline. Release of a baseline constitutes retrieval of source code files (configuration items) from the configuration management system and generating the executable files. A baseline that is delivered to a customer is typically called a “release”.
- Deliver new software versions in regular spaces of time according to project's needs considering defects corrected, improvement and new features.

- Validate the baseline integrity before delivering it.
- Plan releases of new versions in advance and deliver them timely.

Related trustworthy element: No specific one. Configuration Management of CMMI

3.2.9 RECOMMENDATION 8: Resources Management

R8 – Resources Management

Coordinate and manage development resources. Although, frequently, in the FLOSS development each member selects its role according to his or her personal preferences, as a FLOSS project grows the need of organizing different roles and responsibilities grows. A system to distribute different tasks among the project's participants will, for instance, improve the quality of the development process and support continuous improvement of the product and process.

- Define and evidence strategies to manage development recourses, ensuring the necessary resources for project's implementation during all phases. Consider when defining the strategy assignments, tasks, skills, appropriate tools, adequate funding.
- Distribute responsibilities among the people involved in the project, according to different roles in the community.
- Control attributions/tasks assigned to participants in the development process.
- Track available and used resources by controlling allocation and availability of the resources (scaling of resources). Monitor, also, the knowledge and skills of personnel when allocating them.

Related trustworthy element: Relationship between stakeholders (users, developers etc) (STK).

3.2.10 RECOMMENDATION 9: Development Process

R9 – Development Process

Define and evolve a development process (or roadmap). The development of a FLOSS project may involve many volunteers working together on a common objective. Those volunteers may be geographically distributed and working in different time. Such characteristics require special organization of work, such as: well-defined decision making process; set of development and

communication tool; and particular development strategies. Such strategies may include *a homogeneous and structured work-flow, encompassing the whole developing structure. The FLOSS project development process should also consider software maintenance peculiarities.*

- Define (or instantiate) and evolve a development process to develop the FLOSS product. This development process must be compatible and supported by the integrated development environment and developed by geographically distributed teams.
- Consider established development processes as reference to defining and evolving such process.
- Identify and make available the decision process employed to make decisions about new features to be added to a FLOSS project.
- Monitor and control the process, ensuring appropriate visibility of the process.
- Enforce good maintainability of the software within the development process.
- Reinforce the adoption of development rules.
- Adapt the development process to correct deviations and to attend suggestions from the community.
- Ensure lasting stability of the development process.

Related trustworthy element: Availability and use of a (product) roadmap (RDMP), Use of established and widespread standards (STD)

3.2.11 RECOMMENDATION 10: License

R10 – License

Manage licenses. License is an important topic to FLOSS. The understanding of free software licensing issues can contribute, both to ensure that project license is adequate to project needs, and to support decision making on selecting FLOSS products. The decision about which license will be used is a critical decision that has to be taken in the beginning of the FLOSS development process.The license of the product is particular important for integrators as it must be in accordance with the use that the company needs it for.

- Define a decision making process to select licenses.
- Control the software licenses adopted in the project source code. It is important to monitor both the number of licenses and the compatibility among them. It is also important to ensure that no copyright is being violated.
- Establish the need of properly documenting licensing within the code. Verify if it is happening.
- Clearly establish and communicate the license of the product being developed.
- Observe if the project contains software licensed under free software licenses.
- Consider if right to use the incorporated FLOSS has been granted free of charge and the copyright requirements.

Related trustworthy element: License (LCS)

3.2.12 RECOMMENDATION 11: Software product and project assessment

R11 - Software product and project assessment

Pursue good performance of the software product and project and assess such performance.

- Define assessment results about the project.
- Provide automated tools to extract assessment information from the project.
- Assess project results.
- Define clear policy to incorporate feedback and Quality Recommendations from this assessment back again into the development process (see R9 for development process evolution).
- Define and collect software product performance indicators.
- Define and follow a methodology to enhance performance based upon these indicators.

- Search for results of assessment of the product by 3rd party companies

Related trustworthy element: Results of assessment of the product by 3rd party companies (RASM)

3.3 Influences among Quality Recommendations

Quality recommendations are in general strongly related. We've also identified possible influences among Quality Recommendations (see Figure 5). For instance, the availability of project documentation may help promoting effective communication. Effective communication may, in its turn, improve community involvement. In a FLOSS project, the more people are involved to it, the results of project assessment may be positive, as FLOSS project success is dependent of collaboration. The usage of adequate license and well-known FLOSS standards are also positive influences to the results of project assessment. Good results on assessment may increase community involvement to the project. An integrated development environment and an established development process (or roadmap) can also contribute to community involvement, as stakeholders may understand clearly how to contribute. To continuously improve the project, positive influences may be managing efficiently resources, project documentation and release management.

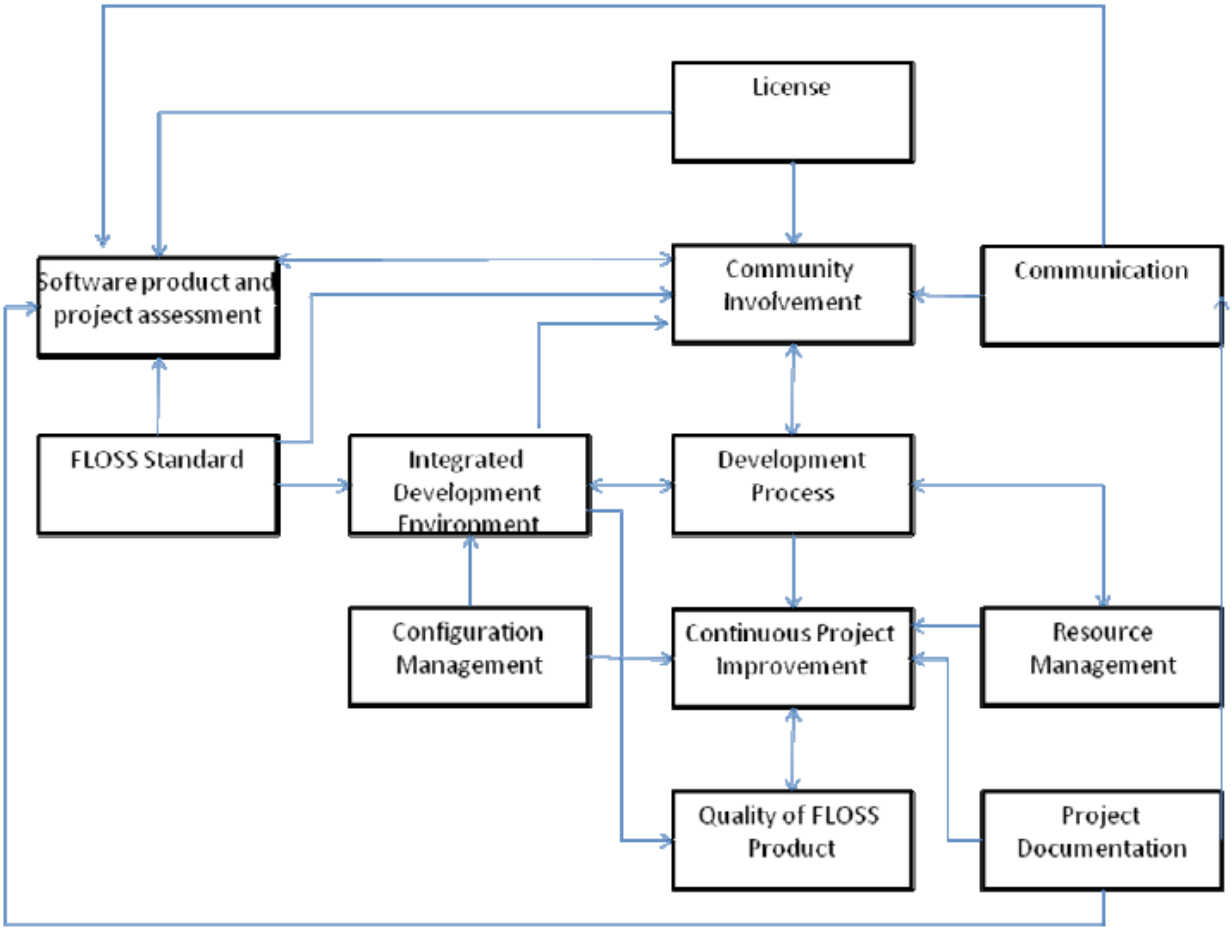

Figure 5: Influences among Quality Recommendations

4. Final Remarks

This report has presented Quality Recommendations for FLOSS development process, built upon trustworthy elements and the OMM model, which is under development. Quality Recommendations are an intermediate step toward the definition of OMM practices. During the identification of those Quality Recommendations some improvement opportunities were identified to the current OMM version. The authors are participants of the Qualipso Project, contributing to different deliverables and working documents of the project. The Quality Recommendations will be included in the OMM working document. It is also setting basis to the definition of new surveys on FLOSS development.

5. References

[1] Qualipso Consortium. Qualipso - Quality Platform for Open Source Software. http://www.qualipso.org, 2009.

[2]QualiPSo: A6 WP6.1 wd6.1.1 (2008). Analysis of the clusters of Open Source processes in FLOSS Communities and in companies: http://www.qualipso.org

[3] Qualipso Project. Deliverable A6.D2.6.2 - Trustworthy elements identified in OS processes, October, 2008.

[4]QualiPSo: A6 WP6.2 wd6.2.1 (2008): Trustworthy elements identified in OS processes: http://www.qualipso.org

[5] SEI. Capability Maturity Model Integration V1.2: http://www.sei.cmu.edu/cmmi/

[6] FSF. The Free Software Definition. Available at: http://www.fsf.org/licensing/essays/free-sw.html. Last access: Nov, 2008.

[7] Qualipso Project. Working Document WD6.3.1 - CMM-like model for OSS, version 1. October, 2008.

[8] Basili, V., Caldiera, G., Rombach, H. "The Goal Question Metric Approach". Encyclopedia of Soft. Eng., vol. 2, September 1994, pp. 528-532, John Wiley & Sons, Inc.