Qualipso Project: Open Process for Evolution, Maintenance and Administration of OMM

Viviane Malheiros, José Carlos Maldonado

RT-347
Abstract

This technical report focuses on setting the basis for an open approach for evolution, maintenance and administration of OMM (Qualipso Open Source Maturity Model). We define a collaborative and distributed strategy and platform to support these perspectives. Such open collaborative platform is designed to support comments, discussion and evolution of the OMM. OMM is a CMMI-like model [4] for FLOSS (Free and open source software) that can be implemented in software organizations to enable FLOSS usage both in production and development of software products. The more OMM is used the more improve opportunities to the model will show up. Handling such improvements is essential to keep the model updated and alive.
# Table of Contents

1 INTRODUCTION .................................................................................................................5

2 BACKGROUND .....................................................................................................................7

   2.1 Open Source Development .......................................................................................... 7

   2.2 Software Process Improvement - SPI ........................................................................... 8

      2.2.1 SPI deployment guidelines ............................................................................... 8

      2.2.2 SPI Success Factors: ......................................................................................... 10

   2.3 The Open Source Maturity Model - OMM ................................................................. 12

3 OMM Evolution Strategy ....................................................................................................13

   3.1 OMM Maintenance, evolution and administration principles ................................. 13

   3.2 OMM maintenance, evolution and administration communities ............................. 14

   3.3 OMM evolution process .......................................................................................... 17

4 OMM Evolution Platform ..................................................................................................21

   4.1 OMM Evolution Requirements ................................................................................. 22

   4.2 Administration Module ............................................................................................ 23

   4.3 Evolution Module ..................................................................................................... 26

5 Final Remarks and Acknowledgement .............................................................................28
6 REFERENCES ........................................................................................................................................29

7 APPENDIX 1 – SUGGESTED STEPS FOR SELECTING THE CDE .................................................32

8 APPENDIX 2 – EVOLVING OMM WITH EPF COMPOSER AND EPF WIKI .................................33
1 Introduction

The Open Source Maturity Model (OMM) is one of the key results of the work done inside Activity 6 of the Qualipso project [2]. Together with OMM other important results will be: (i) the set of tools supporting the use of the model; (ii) the assessment process based on OMM; (iii) the strategy and platform to evolve OMM. Henceforth, the OMM, the supporting tools and the assessment process will be collectively referred to as “OMM process suite”[1]. To evolve the OMM process suite it is important to define an evolution strategy (principles, communities and process) and to prepare an evolution platform.

This technical report sets the basis for an open approach for evolution, maintenance and administration of OMM. We define a collaborative and distributed strategy and platform to support these perspectives. Both of them are designed to support comments, discussion and evolution of the OMM. Such strategy and platform were inspired on Software Process Improvement and process evolution experiences. The real adequacy of this proposal to evolve an open maturity model can only be analyzed from its effective usage. For instance the evolution of a maturity model is expected to be slower that a process evolution.

By the end of the QualiPSo project, in October 2010, the OMM process suite version 1.0 is expected to be provided. Before the project end, there is the necessary funding for proposing and improving the OMM process suite, however after that all three products will have to be self-sustainable. We envisioned that the support for further OMM evolution can come from different sources such as the Qualipso Competence Centers and from the individual or industrial users of OMM [1].
OMM will be described in detail in Qualipso deliverables and working documents. Nevertheless, an important dissemination path for the model will be the training courses prepared inside Activity 11 of the Qualipso project and offered to everybody as open educational material that, in fact, will also compose the OMM process suite. The OMM process suite will be presented at key conferences and in specialized journals. An optional goal, with respect to the visibility and further development of OMM, is the development of an integrated presentation of the OMM process suite, on a web-based platform. The OMM process evolution proposed here considers such goal [1].

Such integrated presentation may consist of [1]:

- a detailed presentation of OMM;
- a presentation of key tools offered as support to OMM and related open training material; and
- the whole assessment environment; a presentation and detailed explanation of the assessment process, and related open training material.

Together with these three key components we believe that it can be useful to provide a few examples of use cases of OMM, and also a large set of assessment results done on popular FLOSS products such as Firefox and OpenOffice. Since the criteria can be different for different users we plan to provide a variegated set of results for common types of use. As for use types we believe it will be of benefit for users to see which are default requirements for other users with similar needs.

Along with the static content, we believe that an OMM evolution strategy and platform may foster fruitful contributions and feedback from users and external experts for further
development of the OMM process suite. For this purpose we plan to provide some interactive functionality supported either by mailing lists, forums, or wikis (see Section 4).

We also argue for establishing different groups for promoting OMM evolution (see Section 3.2). It is important to clearly identify each stakeholder group, as interests and communication needs are different.

An important aspect of the future development of the OMM process suite is its alignment with the FLOSS paradigm (see Section 3.1). For achieving this goal we decided to implement all tools based on FLOSS licenses as FLOSS projects, therefore everybody can contribute to their development; similarly we provided all documents describing OMM, training material and the assessment process as creative commons license compliant and available for free.

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 [9]:

- 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 [10], Qualipso Project contributors have provided an overview of major studies done in the area.

2.2 Software Process Improvement - SPI

Software Process Improvement (SPI) is fundamental to software quality. Different advances have been made in the deployment of SPI standards and models (e.g.: SPICE; CMMI [4]). However, despite all these valuable contributions, there is still a lack of an effective strategy to successfully implement these standards and models [12]. For this reason before defining the OMM evolution strategy, we have identified reported major success factors to SPI (section 2.2.2) and considered SPI deployment guidelines (section 2.2.1).

2.2.1 SPI deployment guidelines

Some available models already helps to deal with SPI, compiling best practices on software development: CMMI [4], ISO/IEC 15504 [13], MPS-Br [14]. OMM (see Section 2.3) goes in the same direction of compiling best practices, but it particularly focused on enabling software companies to use FLOSS software in production and in their main stream products (and not just in prototypes).
To adopt such models’ practices organizations have identified the need of specific deployment guidelines to provide systematic approaches to support deploying and evolving SPI. There some iterative guidelines available: PDCA [15]; QIP - Quality Improvement Paradigm and the Experience Factory [11] and IDEAL [16].

Generally these guidelines can give support to managing SPI as a project. Much of the IDEAL guideline encompasses PDCA and QIP guidelines. The IDEAL is more focused on SPI programs. The IDEAL model (www.sei.cmu.edu/ideal/) is an organizational improvement model that serves as a roadmap for initiating, planning, and implementing improvement actions (see Figure 1). It offers a guideline and life-cycle model for planning and implementing a software process improvement program.

![IDEAL Model](image)

**Figure 1: The IDEAL Model**

As IDEAL has been developed based on industrial cases and as it is the founding strategy employed in delivering many Software Engineering Institute (SEI) services, it can be a good
source of inspiration to define OMM evolution strategy. On the other hand, OMM evolution strategy, being OMM an open model, may demand complementary guidelines to consider: (i) issues related to distributed environment. The new strategy must deal with multi-site, while IDEAL is more suitable to a single-site; (ii) the openness of the model and the need to give all improvements rights to interested communities; (iii) detailing how to implement certain actions and not only what to do. In a previous work, an organizational structure that would improve developer participation on SPI efforts was suggested [8] based on experimental experiences, a first step to distributed SPI. That time focused on how to organize the group responsible for the process evolution

2.2.2 SPI Success Factors:

From the literature we can identify more than a hundred factors/motivators (or barriers) of SPI in different contexts, in the following studies:

Goldenson and Herbsleb (1995) [17] conducted one of the first surveys to evaluate various organizational factors that were believed to influence SPI. The survey scope was 138 individuals from 56 organizations in the United States and Canada. The factors found to be statistically significant were used as input to our research.

Stelzer and Mellis (1999) [18] conducted an exploratory study of factors that potentially affect organizational change in SPI and analyzed experience reports of 56 companies that had either implemented an ISO 9000 quality system, or had conducted a CMM-based process improvement initiative. From this review they identified 10 potential factors.

Dyba (2001) [19] developed an instrument for measuring critical success factors in SPI based on data collected from 120 software organizations. He provides a synthesis of prescriptions for successful quality management and process improvement found from
literature review and confirmed by empirical studies. The outcome of this study was a set of six enabling factors for SPI success.

Baddoo and Hall (2002) [20] present findings based on an empirical study of SPI in 13 software companies where they conducted focus groups with nearly 200 software practitioners. They were focused on better understanding how companies could maximize practitioner support for SPI. The authors also reported two more articles taking different points of view on the same study.

Rainer and Hall (2002) [21] report on a questionnaire survey of key success factors that impact software process improvement (SPI). They refer to Goldenson and Stelzer’s work. They conclude that organizations with different maturity capabilities consider different critical success factors to have a major impact on SPI.

Niazi et al. (2005) [12] also identified some success factors also found in our field study. They compare their results to the literature and rank the factors according to their occurrence in either literature or in interviews scenario.

Montoni and Rocha (2007) [22] recently synthesized results from empirical studies aiming to identify what factors have influence on SPI. In addition, they have conducted a survey with SPI practitioners involved in Brazilian software industry experiences. Their findings reaffirmed some critical success factors cited in the literature.

Awareness of the environment and what has been happening within the organization when executing its SPI process (observing SPI) can provide valuable clues to the causes of diminished software process performance or compliance. Our field observations were based on twelve software development units challenged to continuously improve their development
process and enhance their maturity level (CMMI compliant). We’ve based our findings on the content analysis of a myriad of sources such as assessment reports, Software Engineering Process Groups (SEPG) discussions and feedback from developers.

2.3 The Open Source Maturity Model - OMM

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 [2]. 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).

One of Qualipso project goals is 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. Such CMMI-like model is under construction and its temporary name is OMM – Open Source Maturity Model[1]. Details on it can be found at: http://www.qualipso.org/sites/default/files/A6.D1.6.3CMMLIKEMODELFOROSS.pdf.

Objectives of OMM are: (i) to provide FLOSS communities a basis for developing products efficiently and in a way that makes their products trustworthy for the potential customers, and also for integrating companies; to provide FLOSS integrators a basis for evaluating the FLOSS communities’ processes.

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 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]. Figure 2 shows OMM trustworthy elements and their distribution according to the three levels.

![Figure 2: The OMM Pyramid [1]](image)

### 3 OMM Evolution Strategy

This section describes OMM evolution strategy. In particular, it describes how the OMM communities lead, influence, and collaborate with OMM evolution; basic principles upon which OMM evolution is based and evolution general lifecycle.

#### 3.1 OMM Maintenance, evolution and administration principles

The principles that nurture the maintenance of OMM process suite are:
• OMM is open to all. Everyone participates with the same rules; there are no rules to exclude any potential contributor which includes potential competitors;

• OMM process suite discussions, minutes, deliberations, roadmap, plans for new features, and other artifacts are open, public, and easily accessible. Maintaining an open and public communication channel with all stakeholders is important. The communication channels are to be easy to find;

• The more contributions to OMM the more responsibility the community will earn. Participation in OPG is also merit-based (see Section 8.2); and

• OMM evolution will be compatible with the OMM roadmap.

3.2 OMM maintenance, evolution and administration communities

As OMM is an open model and will be used by Open Source enthusiastic, typical actors were defined based on Open Source development typical roles and adapted based on experiences on Software Process Improvement (SPI). Figure 3 presents the core actors that may contribute to OMM evolution and form OMM communities.

![OMM Evolution Actors](image-url)
As a developer (or other user) contributes to the model he/she may be promoted from a general developer up to an OPG member, being responsible for core decisions on the model maintenance and administration.

**OPG** – OPG is the OMM steering group. They are a group of experts (“the OMM group or OPG”) that will regularly manage the further development of the OMM process suite. OPG will be responsible for managing the OMM evolution, including coordinating the participation and contributions of others; the evaluation of core improvement suggestions to the model and the definition of OMM vision. They are responsible for OMM roadmap and for prioritizing improvement proposals according to OMM releases. OPG is the equivalent of administrators in Open Source Projects. They will be organized as a board of directors. Initially directors will be members of the Qualipso Competence Centers (one indicated member per Center). Such members are expected to be experts and users participating in the improvement of the OMM process suite. Based on the further plan for the development of OMM process suite, the OPG and the specialist groups will include contributions to the model. The OPG will collaborate regularly with the help of virtual and possibly geographically co-located meetings. The future composition of the OPG will depend on the future contributions from Competence Centers or from other users or groups of users and will be subject to change. OPG will take care of OMM evolution using open source rules of engagement: meritocracy, transparency, and open participation (see Section 3.1).

**Specialist Groups** – While OPG are the equivalent of “administrators” in Open Source Projects, Specialist groups are the “committees” of the model. These groups have the right to make changes to the OMM that will be available in the next release of the model. Specialist groups may be organized according to their expertise. For instance a particular group can be
assigned to deliberate on the OMM assessment process and another one to the metrics definition. Or a group may focus on the FLOSS integrators point of view while another one focuses on FLOSS communities’ concerns. Maybe in the beginning there is no need to differentiate OPG from specialist groups. However the scope of an Open Source project is unpredictable, and so, we cannot determine in advance how many people will be interested in OMM evolution. If such group grows, it might be useful to distinguish OPG (focusing on OMM vision and administration) from specialist groups (focusing on commits and quality control). Specialist groups should be an open, transparent and inclusive community. And OPG must encourage contributors to become specialist group members. Specialist group members are required to monitor the communication channels (mailing lists, forums, etc.)

**General users and contributors** – All OMM users might be considered as a potential contributor. They might be FLOSS integrators or FLOSS communities’ members, or even a Qualipso Competence Center member. They can contribute to the OMM evolution: (i) reporting their experience on using the model and reinforcing usage scenarios; (ii) issuing model improvement proposals; (iii) finding and reporting inconsistencies while applying the model. Such contributions will feed OPG and specialist groups discussions. They can also contribute for instance translating the model to other languages. Due to the openness of OMM and the FLOSS nature of supporting tools everybody will be able to adapt the model or the tools to their needs, however only suggestions accepted by the OPG, and specialist groups, and inserted in the newest version of the model will be assigned an official level of compliance to OMM. Contributors who have the trust of the specialist groups can, through election, be promoted to specialist group member.
OMM Sponsors – An additional support can eventually be contributed by future projects funded either by commercial companies or public bodies that will focus on the further development of OMM. OMM sponsors may provide funding for proposing and improving the OMM process suite, after the end of the Qualipso project. They are a particular OMM community as they are not directly involved in the OMM technical decisions, but they contribute providing means to keep the OMM platform alive.

It is important to clearly identify each stakeholder groups, as interests and communication needs are different.

3.3  OMM evolution process

This section presents OMM evolution process. For executing this process it is important to prepare to evolution, defining evolution principles and interactions (See Figure 3). The present technical report represents the execution of the preliminary step “0”. Once the collaborative evolution strategy is designed (principles, requirements, communities and process are proposed), the platform is customized and the OMM version 1.0 is delivered, OMM evolution process will then take place.

OMM evolution will be handled as a project. A roadmap will be established and maintained by the OPG. It is an iterative evolution of continuous definition, development, release and use. The iterative characteristic of OMM evolution process reflects the awareness to major SPI guideline recommendations. Figure 3 presents the intended OMM evolution process. It major step is detailed in the following subsections.
1. Define and refine OMM roadmap

OMM roadmap will describe the future directions of the OMM process suite, clearly defining major themes and its priorities and general OMM evolution project plan. It entails strategic goals and future directions of the model. The OMM roadmap will be revised and approved by the OPG once every two years.

Creating or updating OMM roadmap begins with the OPG proposing a set of Themes and Priorities that constitutes the future directions and the improvements proposed by users and contributors and sponsors. The entire OPG review these Themes and Priorities. The process of producing or updating the Roadmap is expected to be iterative. Specialist groups may interact
and agree to the roadmap. For instance, an initial set of Themes and Priorities may be infeasible to implement in the desired timeframe, specialist groups will point that.

At the end of the QualiPSo project, in October 2010, within the OMM version 1.0, a preliminary roadmap will be proposed.

2- Plan OMM releases

In the context of OMM evolution a release is a project that addresses a group of recommendations for improving the model. Each release will provide incremental improvement over the previous releases.

Once every year a new OMM process suite release is expected to be planned and distributed. Meanwhile, intermediate releases may be published according to OPG determination. Based on the roadmap direction and specialist groups’ suggestion on prioritization, OPG will decide on what improvement proposals will be delivered at each release. A release schedule will be published clearly defining all release milestones. When deciding what to include in each release whenever possible, OPG should consider the balance between different stakeholders. Sponsors will want to see the return on investment (ROI) for the organization.

A release may include any subset of the OMM process suite. The infrastructure provided to release OMM may vary over time and is defined outside this activity (see OMM Evolution Platform – Section 4).

3- “Develop” OMM improvements
Once the suggested improvement is approved by the OPG it should be developed by the specialist group. All changes to OMM will be versioned. All OMM process suite shall be documented in an open format. We are suggesting XML.

4- Review OMM changes

All major changes to OMM process suite must be announced and reviewed by the membership-at-large (at least OPG and specialist group members will participate in a review). Major changes include OMM planned releases as well as the introduction or exclusion of significantly new capabilities. Before each release a review will be initiated through the OMM evolution platform. A review schedule will be disseminated, no less than one week before the review period. Together with all planned changes, a summary of major changes will be provided to the reviewers.

5- Publish OMM new release

OMM should be released according to the release plan. Every publication shall be communicated to all OMM registered community. Any release changing OMM description must be correctly labeled.

6- Handle OMM improvement proposals

To handle OMM improvement proposals an issue tracker is suggested. The issue tracker (adapted to OMM improvement proposals) records the suggestion and allows track of proposals until its resolution. Proposal analysis will minimally look like this:

- A general user proposes and files an improvement to OMM. He/she provides a summary and an initial description. The improvement is then in the Open state.
Other contributor may read the suggestion and perhaps ask for clarification or reaffirm his/her common interest in the suggestion;

- A specialist group member analyzes the suggestion, emits her/his evaluation, including a suggested priority. The member may also at this moment take the ownership of the improvement;

- The improvement gets schedule for resolution, which means deciding which future release the improvement should be released in;

- The improvement is developed and the new status is recorded in the improvement; and

- The improvement is closed or marked as resolved.

4 OMM Evolution Platform

To integrate all OMM related information an OMM Evolution platform will be deployed (see Figure 4). Such platform will contain:

- Tools to support the model maintenance in a controlled manner, as an open content management system; and

- Communication, coordination and collaboration mechanisms to maintain and evolve the model.
Along with the openness of the process model we plan to provide a web presence of OMM open to contributions and improvements by everybody with only editing privileges guaranteed to the OPG.

4.1 **OMM Evolution Requirements**

Considering the OMM evolution context, we have included as basic sources for defining the platform requirements:

- OMM trustworthy elements [3] – the goal is to make OMM evolution itself benefit from the trustworthy elements identified in the model for FLOSS development. For instance Licensing is an important element to FLOSS development, we have clearly defined that OMM is creative commons license compliant and available for free. This information will clearly be published in OMM site; and

- SPI critical success factors - Recurrent factors are: (i) the need of staff motivation and involvement; (ii) the benefits of feedback, support for discussions and clear establishment of goals; and (iii) the availability of resources to improve.
The rationale to define such requirements was, from the SPI critical success factors and trustworthy elements, the major necessities for the OMM evolution platform were identified. For each necessity the expected result was established as well as how such necessities could be attended. Major requirements to OMM evolution platform are presented in Table 1.

Table 1 – Major requirements to the OMM evolution platform

<table>
<thead>
<tr>
<th></th>
<th>Communication mechanisms that can enable cooperation such as discussion forums and mailing lists around OMM. In addition, the possibility of communicating events or news about OMM; publishing information on a message board or informing a particular community of interest (for instance OPG members);</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Access to OMM information (OMM process suite and OMM administration) through a unique starting point;</td>
</tr>
<tr>
<td>3</td>
<td>OMM under version control and the possibility of changing the OMM by more than one person from more than one place;</td>
</tr>
<tr>
<td>4</td>
<td>Improvement Proposals handling and the possibility of tracking each proposal status until its conclusion (deployment or rejection by OPG);</td>
</tr>
<tr>
<td>5</td>
<td>Transparent backlog of process improvement and support requests allowing anyone to contribute to improvement analysis and to clarify questions; and</td>
</tr>
<tr>
<td>6</td>
<td>User spaces filtering information regarding user actions, such as a list of improvement suggestions submitted by the user.</td>
</tr>
</tbody>
</table>

4.2 Administration Module

OMM Evolution platform will be a customization of a collaborative development environment (CDE) and an editing tool that can be integrated to a control version management system.
To stimulating volunteer’s contributions we believe it is helpful that the OMM evolution platform is compatible to the environment they are used to. For this reason OMM maintenance, evolution and administration will be supported by a collaborative development environment (CDE). CDE usage is a one of FLOSS best practices, all major Open Source project is evolved and administrated with the support of a forge.

A CDE is a virtual space wherein all the stakeholders of a project – even if distributed by time or distance – may negotiate, brainstorm, discuss, share knowledge, and generally work together to carry out some task, most often to create an executable deliverable and its supporting artifacts. CDEs are largely used by open source community and will be an important tool in the Qualipso Factory. The same CDE selected by the Qualipso factory will be used to OMM Evolution Platform. For selecting such CDE we suggest following some major steps and considering some minimal demanded characteristics as defined in Appendix 1).

CDEs provide integrated access for different mechanisms and tools, creating a virtual project space focused on the particular goal of a team. Considering OMM evolution as such goal, we can imagine a virtual project space for the OMM process suite, wherein OPG members, specialist groups and contributors can work together to carry out its evolution. Likewise, they can negotiate, brainstorm, discuss and share knowledge about improvements toward a better model. They may propose, analyze, and discuss model improvements; and implement (edit) and deploy them, making use of the Internet as the medium for their interactions.

Figure 5 shows a mock up of the OMM administration module. The screen shot highlights the virtual space of OMM “project”. It can be accessed through one unique URL (see number 1, Figure 5). Once in the virtual project space, all communication and collaboration
mechanisms are available to maintain and evolve the module: forums; news; trackers (bug-tracker like tools); reports; and historical data about it. Also, information about all actor interactions is available.

Accessibility through the Internet can enhance flexibility, distribution and easy connection of new tools.

At the main page, it is possible to know the latest news about the model (see number 2, Figure 5). For instance, OPG can use this feature to inform when the next conference about OMM will take place or when a new major release of the model will be published. Also, from the main page, it is possible to access all trackers related to the improvement of the OMM process suite. For instance, it is possible to access all filed improvements, their status and who is handling each one (Figure 5, id. 4).

![Figure 5 – The virtual project space of a software development model](image)

25
4.3 *Evolution Module*

To editing the OMM process suite different editing tools can be used. Two examples are Atabaque ([http://sourceforge.net/projects/atabaque/](http://sourceforge.net/projects/atabaque/)) [23] and EPF Composer ([http://www.eclipse.org/epf/downloads/tool/tool_downloads.php](http://www.eclipse.org/epf/downloads/tool/tool_downloads.php)). Both of them are open source tools that support process documentation allowing all content managed to be published to HTML and deployed to Web servers for distributed usage. They both support segregation between process content and format. They both aim providing a knowledge base of intellectual capital that can be browsed, managed and deployed. They are designed for process engineers and project managers to author, tailor and publish methods and processes for development organizations and projects.

As an initial step, the actual version of OMM, available in [1] was documented using EPF Composer. EPF composer allows the implementation of Software Process Engineering Meta-Model, which might be interesting for representing OMM components. Figure 6 shows OMM structured using EPF Composer. Details on how EPF composer may be used to evolving the model are presented in Appendix 2. Such version was also uploaded in EPF wiki (see Appendix 2).

EPF Composer provides the following key capabilities [24]:

- Provides completely redesigned tools for authoring, configuring, viewing, and publishing methods and processes;
- Provides just-in-time generation of publication previews in dedicated browsing perspective that allows rapid configuration switching;
• Manages method content using simple form-based user interfaces. Therefore, UML modeling skills are no longer required;

• Provides intuitive rich text editors for creating illustrative content descriptions. Editors allow use of styles, images, tables, hyperlinks, and direct HTML editing;

• Allows creating processes with breakdown structure editors and workflow diagrams through use of multi-presentation process editors. Breakdown structure editor supports different process views: work-breakdown view, work product usage view, and team allocation view. EPF Composer automatically synchronizes all presentations with process changes;

• Improved reuse and extensibility capabilities. The plug-in mechanisms from past versions have been extended to support extensions for breakdown structures; and

• Supports reusable dynamically-linked process patterns of best practices for rapid process assembly via drag-and-drop.

In the left (Figure 6), a navigation tree allows easy access to all Open process suite components, documented using EPF composer.

The CDE (Section 4.1) contains a link to the OMM process suite site.

The OMM process suite is expected to be under configuration management control (like a software would be) and it can be accessed through a CVS link (see number 3, Figure 5). CVS keeps track of changes in model files and allows several stakeholders to collaborate simultaneously.
5 Final Remarks and acknowledgement

This report has presented an Open Process for Evolution, Maintenance and Administration of OMM, built upon open source development processes and the critical success factors to SPI. The authors are participants of the Qualipso Project, contributing to different deliverables and working documents of the project. This process will be reviewed and included in the OMM working document [3].

The authors thanks Qualipso project, CAPES/PDEE program (BEX2979/07-1), FAPESP and CNPQ. They also thank Marco Aurelio for installing the EPF wiki server to run the OMM instantiation.
6 References


7 Appendix 1 – Suggested Steps for selecting the CDE

In other to select the tools to OMM evolution platform, the following steps are suggested (Figure 7).

![Diagram of suggested steps]

Figure 7: Steps for selecting a CDE to be customized
8 Appendix 2 – Evolving OMM with EPF Composer and EPF wiki

This appendix describes how OMM can be evolved with EPF Composer and EPF wiki. EPF composer development guideline (http://www.eclipse.org/epf/composer_dev_guide/) and EPF wiki user guide (http://wiki.eclipse.org/EPF_Wiki_User_Guide) were very helpful.

After installing EPF Composer (Eclipse Process Framework Composer version 1.5), the first step was to map OMM components to EPF Composer key terminology and concepts. EPF Composer provides various form-based editors to create new method content elements. EPF Composer allows categorizing the content based on a set of predefined categories (for example, tasks can be categorized into development disciplines, work products can be categorized into domains). Though more suitable for documenting processes it can be used to document the model also.

Figure 8 provides a summary of the key elements used in EPF Composer and how they relate to method content or process. Method content is primarily expressed using work products, roles, tasks, and guidance. As OMM is not a process, but a model, the elements in the left circle were used.
Figure 8: Key terminology used by EPF Composer [24]

An OMM method framework was created, as a general workspace to hold OMM. A method content library named OMM was created. Method content describes roles, the tasks that they perform, the work products that are used and produced by those tasks, and supporting guidance.

All OMM TWEs were mapped as the type “practice”, a guidance element in EPF Composer. Special guides on how communities or FLOSS integrators may use OMM differently according to their particular interests were mapped as the “guideline” guidance element. Each OMM level was mapped as a category element in the EPF Composer. Key concepts of OMM, for instance the definition of a TWE were mapped as the element “concept”. Additional info about the model such as the related working documents, OMM background and how to contribute to OMM were mapped as the “report” guidance element. Mapping OMM to EPF composer
elements can help linking model practice to new open process to be generated using EPF Composer.

Figure 9 shows a TWE authoring example. Guidelines on how to use EPF Composer tool can be found in the tool Help menu.

![Figure 9: Editing the OMM TWE [24]](image)

In EPF Composer Method configurations are the basis for publishing method content and processes. A published configuration is an HTML web site that presents all the method content and processes of the method configuration in a navigable and searchable way. It uses the relationships established during method content and process authoring to generate hyper-links between elements as well as provides tree browsers based on the configuration view and user-defined categorizations of the content [24]. A method configuration is simply a selection of the method plug-ins and packages.
An OMM method configuration was created to organize OMM and related content, as shown in Figure 10.

![OMM method configuration diagram]

**Figure 10: OMM method contents elements organized by OMM Method Configuration**

Once all content is created, the publication wizard helps publishing the content that is part of the method configuration. It automatically adopts content to the configuration such as removing references of method content elements to elements outside of the configuration set.

Two sets of documents are then available: the model “code” (elements in the XMI format) and the published documents. Two tools can then be used to evolve the model the EPF composer itself and the EPF wiki. The EPF wiki infrastructure is available in Figure 11. EPF Wiki is Wiki technology designed to be used together with Eclipse Process Framework (EPF). This
allows combination of Wikis’ features and process (structured) frameworks. It combines a modular method construction approach and the flexibility and ease of use that is the defining characteristic of a wiki.

![EPF Wiki Infrastructure](image)

**Figure 11: EPF Wiki Infrastructure [25]**

The EPF wiki tool may help collecting improvement proposals to the model. It is a Web Application created using Ruby On Rails Framework and is Wiki technology that works on plain HTML (no Wiki markup language) easy editable using commonly available browser-based WYSIWYG HTML editors.

After EPF wiki is installed it can be accessed using an Internet browser. A particular OMM wiki was created using a baseline model (a published OMM generated by EPF Composer, see “process presentation” in Figure 11). Help on creating baseline model can be found at [http://wiki.eclipse.org/EPF_Wiki_User_Guide](http://wiki.eclipse.org/EPF_Wiki_User_Guide). Once the OMM wiki site is created, contributors can suggest changes to the model by naturally editing the wiki pages. EPF wiki will log all
changes, as many wiki environments. Users may discuss changes, add comments, and control changes using check-in and checkout features.

Periodically, or according to the OMM roadmap, the OPG (see Figure 3) can harvest the suggestions and decide on the change proposals. Harvesting basically consists of reviewing changes, new pages, uploads and updating the process asset library using EPF Composer. EPF wiki allows version comparisons and OPG can easily identify the difference of the suggested version to the previous one. Figure 12 illustrates how an improvement proposal to OMM can be easily identified and analyzed by OPG. Insertions are highlighted in green and deletions in red.

![Figure 12: Changing an OMM TWE using EPF wiki](image)

Figure 13 illustrates how OPG can identify which pages were changed. In such example a suggestion of change was proposed in the Reputation page by Viviane Malheiros at July 30th.
For each page changed a new entry will be generated in the table. Once the suggestion is analyzed the “Review Complete” column can be checked.

Figure 13: Example of OMM change