QUALGEN: Modeling and Analysing the Quality of Evolving Software Systems

Tom Mens and Leandro Doctors
Institut d’Informatique, Faculté des Sciences
Université de Mons – UMONS, Mons, Belgium
{ tom.mens | leandro.doctors }@umons.ac.be

Naji Habra and Benoit Vanderose and Flora Kamseu
PReCISE, Faculté d’Informatique
FUNDP, Namur, Belgium
{ naji.habra | benoit.vanderose | flora.kamseu }@fundp.ac.be

Abstract—In this article we present an ongoing interuniversity research collaboration in the context of a large ERDF-funded research project aiming to enhance and support the quality of evolving software-intensive systems. The project focuses on two aspects in particular, namely the development of a quality metamodel for measuring and controlling the quality of software-related activities, and the instantiation of this framework to measure the quality of evolving libre software distributions from the point of view of different stakeholders.

Keywords-software evolution, software quality, libre software, software distribution, software ecosystem

I. INTRODUCTION

The European Regional Development Fund (ERDF) promotes economic and social cohesion by correcting the main regional imbalances and participating in the development and conversion of regions. In its 2007-2013 programme, the ERDF finances a consortium consisting of 4 universities from Wallonia\(^1\), (FUNDP, UMONS, UCL and ULB), 3 research centers (CETIC, CENAERO and MULTITEL) and several small- and medium-sized local companies involved in ICT. This project, supported by Wallonia, is lead by the Center of Excellence in Information and Communication Technologies (CETIC). The goal of this project is to stimulate transfer of technologies and research results from academia to local industrial partners. More specifically, it focuses on developing a portfolio of innovative techniques related to the engineering of information systems, allowing local companies to master the diversity, complexity, quality and rapid evolution of these systems.

To achieve this, two virtual centers of expertise have been put into place:

- CEIQS: This center of expertise in engineering and quality of systems focuses on the problems and challenges typically encountered while developing and maintaining heterogenous and service-oriented software-intensive systems. The goal is to come up with automated solutions to control the quality of these systems, allowing companies to reduce the time-to-market and development cost of their systems.

- CELAVI: This center of expertise on the use of libre software in industry aims to provide a set of services allowing companies and governmental organisations to exploit the most recent developments in libre and open source software, in order to reduce their development cost and obtain a higher quality end-product.

The authors of this article, originating from two different Belgian universities (FUNDP and UMONS), closely collaborate together since 2010 in the context of the CEIQS center. This work has also clear links to the research activities carried out in the CELAVI center, since most of the case studies used during our research will be chosen in the domain of libre and open source software.

Due to the size and the number of participants involved, the project is subdivided along different themes. The collaboration that we report upon in this article is carried out in the context of the theme “Methodologies for developing and evolving (software-intensive) systems”. More specifically, we work together on the workpackage QUALGEN, focusing on the modeling and evolution of quality from different points of view. Our research is subdivided into two main activities with some clear interdependencies between them: (i) the development and subsequent instantiation and validation of a quality metamodel; and (ii) the study and control of quality evolution of software ecosystems. These activities are presented in more detail in the next two sections.

The remainder of this article is structured as follows. In section II, we present our work on the MoCQA Quality Metamodel. Section III outlines our ongoing research on the study of evolving software ecosystems. Section IV explains how these two different lines of research are combined as part of our ongoing collaboration. Section V presents related work and section VI concludes.

II. QUALITY METAMODEL

The Model-Centric Quality Assessment (MoCQA) framework [1] is a theoretical framework that applies software measurement principles, associated with a quality model approach, to allow a continued quality assessment of software all along its lifecycle. Its quality metamodel (Fig. 1) supports the design of Customised Assessment Quality Models
(CAQM) that fit the specific needs of any given software project development lifecycle and integrates three groups of concepts: quality-related, measurement-related and project-related concepts.

Thanks to the quality-related and measurement-related concepts captured in the metamodel, the framework allows the integration of various existing quality models (or parts of them) and various measurement and/or estimation methods within the same CAQM. This integrated quality model addresses the different aspects of a specific software project all along its lifecycle.

Meanwhile, the project-related concepts of the metamodel support an original view of software, focusing its quality assessment on the software project. The software project is defined as a structured collection of artefacts linked by derivations and produced to support/provide a collection of in use behaviours in order to satisfy a set of user requirements. This definition reflects the fact that software is no more considered as a black-box, but as a network of interconnected artefacts with various levels of abstraction and completion (e.g., intermediary diagrams or pieces of code, user-, designer- or programmer-oriented documents, etc.) associated with observable behaviours. Due to this specific point of view, MoCQA is perfectly adapted to address the notion of ecosystem that will be introduced in section III and can take it into account during the quality assessment process.

Once the CAQM is designed, the actual measurement plan is defined. The objective of this plan is to allow the measurer to find the right measurable entities and to apply corresponding adequate measurement procedures. These are defined for each measurement method of the CAQM. Assessment guidelines are concretely defined and actual project resources (i.e., the concrete artefacts or behaviours to be measured) are located and identified accordingly to the measured entities defined in the CAQM.

III. EVOLVING SOFTWARE ECOSYSTEMS

In this section, we present our research on the study and analysis of evolving software ecosystems. We take an ecosystem-centered approach because “no project is an island. Software projects exist in larger contexts [. . .] ecosystems” [2]. The analysis of ecosystems requires different kinds of artefacts used and produced during the software development process, beyond source code [3]. In our particular context of use, we understand the term software ecosystem as the set of all possible artefacts (source code, documentation, mailing list archives), entities (communities, projects), processes (development and business models) and any other aspect that might influence the system under study.

The systems we are studying are libre and open source software distributions. These are collections of “assemblable” software components that work seamlessly together (software packages), along with tools to manage and configure the entire system. Examples of well-known libre software distributions are operating systems based on the Linux and/or *BSD kernels, like Ubuntu, Debian and FreeBSD.

We chose to focus on libre software distributions because, even though they are very important in the software landscape, they have been rarely studied. They are also “probably the most complex type of software ecosystem” [2]. This is because they include several types of data that are not contained in any isolated software project, such as both the original and modified source code of their software packages, and the information related to the dependencies across them.

The quality of such libre software distributions can be studied from different stakeholders’ points of view: the user communities and the developer communities. It is clear that these different types of stakeholders will have different quality priorities and favor different, often contradictory, ways to evolve the system. On the one hand, user satisfaction of a libre software distribution relates to quality aspects such as popularity, usability and lack of defects. On the other, developers are be more interested in quality aspects such as readability, reusability, maintainability and portability. To study these different quality issues and their impact on how the system evolves, we follow the Goal-Question-Metric approach [4].

We are currently in the process of applying the aforementioned approach to the case study of two libre GNU/Linux-based software distributions, namely Ubuntu and Debian (see Table I for a comparison). The choice of these distributions is motivated by two factors:

1) Ubuntu and Debian are co-evolving software distributions: many of the packages and developers are shared across both distributions. As such, the evolution of the quality of either distribution will influence the other.

2) For our study, we can rely on the availability of the Ultimate Debian Database, a federated database storing many different kinds of information pertaining both studied distributions, from both the developer and the user point of view.

IV. INSTANTIATING THE MoCQA FRAMEWORK

As a concrete application and a step for validating the MoCQA quality framework of Section II, we are currently instantiating it in the context of evolving software distribution ecosystems (see Section III). Since the MoCQA framework provides a quality model lifecycle and offers a viewpoint that is perfectly adequate for the notion of ecosystem, we will follow an iterative approach to produce the CAQM for evolving software distributions. Each iteration will generate a candidate CAQM. Fig. 2 shows the different steps followed to generate each CAQM. These steps are:

1) Structure the quality hypothesis in terms of a quality factors hierarchy, that is: (a) identify the stakeholders
Table I
MAIN DIFFERENCES BETWEEN DEBIAN AND UBUNTU.

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Debian</th>
<th>Ubuntu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nature</td>
<td>Non-commercial, community coordinated, volunteer driven</td>
<td>Commercial, driven and financed by company Canonical</td>
</tr>
<tr>
<td>Supported hardware platforms</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>Offered packages</td>
<td>28,000+</td>
<td>30,000+</td>
</tr>
<tr>
<td>Supported packages</td>
<td>All packages belonging to each release are officially supported</td>
<td>Only a subset of packages is officially supported</td>
</tr>
<tr>
<td>Official developers</td>
<td>1,000+</td>
<td>150+</td>
</tr>
<tr>
<td>Release management</td>
<td>“Release when ready” (on average every two years)</td>
<td>Time-based (every six months)</td>
</tr>
<tr>
<td>Age</td>
<td>17 years (since 1993)</td>
<td>6 years (since 2004)</td>
</tr>
</tbody>
</table>

The generation of the CAQM (and indirectly, the MoCQA metamodel) will be validated on the Debian and Ubuntu software distributions.

V. RELATED WORK

The MoCQA methodology is similar to the Goal/Question/Metric (GQM) approach [4]. Both frameworks are based on a top-down approach designed to define relevant measures from predefined measurement goal. The MoCQA framework provides a more structured approach because it relies on its own quality metamodel (Fig. 1). There exist some frameworks that also rely on metamodels (e.g., [5]). The MoCQA metamodel is more flexible as it provides a rich characterization of the measured entities. Additionally, the quality metamodel is compliant with the ISO 15939 Measurement Information...
Model [6].

Regarding software ecosystems, Lungu [2] presented a theoretical framework on software ecosystems, along with a methodology on how to reverse engineer it. Although it is not aimed at distributions, we plan to extend the framework to support them. The only distribution evolution case study we are aware of is a study of the Debian distribution by Gonzalez-Barahona et al. [7]. They consider Debian as a system and its contained applications and libraries as subsystems. They show findings regarding Debian’s size, packages size, package dependency complexity and most popular programming languages. However, they do not consider the ecosystem of the distribution.

VI. DISCUSSION AND FUTURE WORK

The QUALGEN workpackage on which we are collaborating has now finished its first year (out of 4 years). The final objective would be to deliver two prototype tools, one for each of the main activities in the workpackage. The first tool would enable and support the creation and use of customised assessment quality models (CAQM) based on the general MoCQA framework. The second tool would enable the analysis of the quality and evolution of software distributions, based on a dedicated CAQM that has been created for this purpose. This tool would have different interfaces, depending on the stakeholder using the tool: end users may consult the tool to choose the most appropriate software distribution according to their desired quality criteria; developers may use the tool to compare and improve the quality of software distributions.

To reach the QUALGEN objectives of analysing software quality from different points of view, we follow an iterative approach. We are currently in the process of instantiating the MoCQA quality metamodel into a new CAQM for evolving software distributions. The CAQM (and indirectly, the MoCQA metamodel) will be validated on the Debian and Ubuntu case studies. The results of this validation will be fed back into a further improvement of MoCQA, and will lead to a generalisation of the CAQM, by applying it to other case studies of evolving software distributions. One of these will be the FreeBSD operating system, but we will also study the evolution of other types of libre software distributions. In parallel with these validation studies, the aforementioned prototype tools will be developed.

ACKNOWLEDGMENT

This research has been co-funded by the European Regional Development Fund (ERDF) and Wallonia, Belgium. The research is also partially supported by the F.R.S.-FNRS FRFC project 2.4515.09.

REFERENCES


