On the Use of Non-technical Requirements for the Evaluation of FOSS Software Components

Juan Pablo Carvallo[1(✉)], Fabián Carvajal[1], Esteban Crespo[1], Lucia Mendez[1], María José Torres[2], and Rosalva Vintimilla[1]

[1] Universidad del Azuay (UDA), Av. 24 de Mayo 7-77 y Francisco Moscoso, Cuenca, Ecuador
{jpcarvallo, fabianc, ecrespo, lmendez, rvintimilla}@uazuay.edu.ec

[2] Integral IT, La Verbena 26 y Emilio López Ortega, Cuenca, Ecuador
mariajose.torres@integralit.ec

Abstract. Modern enterprises rely on Information Systems specifically designed to manage the increasing complexity of their operation. In the usual case, they are built as hybrid systems which integrate several software components of different nature and origins e.g.; legacy systems, web services, commercial components (typically referred as COTS) and, Free and/or Open Source Software (FOSS). The evaluation of individual software components is highly relevant in this kind of system and is usually conducted with the support of software Quality Models. However, these artifacts usually consider only the evaluation of technical quality requirements, in detriment of non-technical ones (e.g. costs, legal and quality of suppliers) which can be just as critical, particularly in the selection of COTS and FOSS. In this paper, we propose an extension to preexisting software Quality Models, intended to deal with technical and non-technical quality requirements in a homogeneous and holistic way. The relevance of the approach is illustrated in relation to four industrial FOSS adoption processes.

Keywords: Quality model · FOSS · Free and open source software · Quality requirements · Non-technical quality requirements

1 Introduction

Modern enterprises rely on Information Systems (IS) specifically designed to manage the increasing complexity of their operation. In the last decades, several alternatives to the traditional development of software from scratch, have emerged in order to construct such IS. In the usual case, they are built as hybrid systems which integrate several software components of different nature and origins e.g., legacy systems, web services, commercial components (typically referred as COTS) and, Free and/or Open Source Software (FOSS). Because of the nature of hybrid systems, proper selection evaluation of software components plays a prominent role. This critical task is usually conducted with the support of *Software Quality Models* (QM) [1], artifacts specifically engineered to support the specification, prioritization and evaluation of requirements, the

M. Piattini et al. (Eds.): QUATIC 2019, CCIS 1010, pp. 64–78, 2019.
https://doi.org/10.1007/978-3-030-29238-6_5

description of software components, and the identification of gaps among their features and requirements.

Several proposals for QM and QM standards have been proposed [1–4], particularly in the case of FOSS several proposals exist e.g. QSOS [5], OpenBRR [6], QualOSS [7], EFFORT [8], most of them as extensions of the ISO/IEC 9126 [9] standard. However, these approaches focus on the evaluation of technical functional and non-functional (internal and external) requirements, forsaking other non-technical aspects which can be just as relevant (e.g. economic, legal and quality of the provider), particularly when acquiring software from third parties e.g. communities and commercial suppliers.

In [12, 13] we found an extension to the ISO/IEC 9126 quality standard, which deals with technical and non-technical quality requirements in a homogeneous and holistically way. 3 characteristics and 15 non-technical characteristics were added to the original ISO/IEC 9126 catalog (which can also extend its successor, the ISO/IEC 25010 standard) for this purpose. This extension is called the extended ISO/IEC 9126 catalog. However, due to its nature, and after conducting an extensive systematic literature review (SLR) to identify critical success factors, potential benefits and risks in the adoption of OSS, we identified that this extension is not enough to deal with non-technical quality requirements in the case of FOSS. In this paper, we propose a new extension to the extended ISO/IEC 9126 quality standard required to support the evaluation and selection of OSS.

In addition to this section, this paper is structured as follows: Sect. 2 introduces previous and related work; Sect. 3 presents the SLR in which this work is based and its main findings; Sect. 4 introduces the new proposal for the extended ISO/IEC 9126 catalog; Sect. 5 evaluates its relevance in relation to four FOSS industrial adoption processes in which we have actively participated; finally, Sect. 6 presents the conclusions and lines of future work.

2 Previous and Related Work

2.1 The ISO/IEC 9126 Quality

The ISO/IEC 9126 software quality standard is one of the most widespread quality standard available in the software engineering community. It proposes quality models as the artifacts that keep track of the quality factors that are of interest in a particular context, i.e. for a software domain of interest. From the 4 parts that compose it, we focus here on the 9216-1 part of the standard [9], which presents a catalogue of quality factors intended for the evaluation of software components.

The ISO/IEC 9126-1 standard fixes 6 top-level characteristics: functionality, reliability, usability, efficiency, maintainability, and portability. It also fixes their further refinement into 27 sub-characteristics but does not elaborate the quality model below this level, making thus the model flexible. Sub-characteristics are in turn decomposed into attributes, which represent the properties that the software products belonging to the domain of interest exhibit. Intermediate hierarchies of sub-characteristics and attributes may appear making thus the model highly structured.

When the domain of interest is complex, building ISO/IEC 9126-based quality models may be tough. We apply the methodology described in [6] for **I**SO/IEC 9126-based **Q**uality **M**odel **C**onstruction.

2.2 The Extended ISO/IEC 9126

[12, 13] contain an extension of ISO/IEC 9126 technical characteristics and sub-characteristics with additional non/technical ones arranged in an ISO/IEC 9126-1 tree-like structure, thus the resulting catalogue includes high-level quality characteristics and sub-characteristics, and also lower-level quality attributes. To build the hierarchy we followed the 6 step method presented in [10, 11].

The top-level of the hierarchy has been structured with 3 non/technical characteristics: *Supplier*, *Costs*, and *Product*. These three characteristics group non-technical quality features required to measure the supplier capacity to address and support the project, the implementation costs and the out-of-the-box quality and effort required to get the component running. These non-technical quality characteristics have been further decomposed into 15 non-technical sub-characteristics (see Table 1). Some of them have been decomposed into other sub-characteristics, whenever they were required for structuring or leveling purposes. Following the construction approach, sub-characteristics have been further decomposed into over 160 non-technical quality attributes.

Table 1. Non-technical quality features upper-level hierarchy.

Characterstics/ Sub-charact.	Definition
Supplier	
Organizational structure	Description of the organizational structure of the supplier company.
Positioning and Strength	Description of the position of the supplier company in the market.
Reputation	Capability of the supplier to perform similar projects based on past experiences and certifications.
Services Offered	Description of the services offered by the supplier.
Support	Description of the support mechanisms offered by the supplier.
Cost	
Licensing Schema	Description of the COTS component licensing options.
Licensing Costs	Detail of the costs for the different licensing options
Platform Cost	Estimation of the cost for the required production platform
Implementation Cost	Estimation of implementation costs based on similar past experiences
Network Cost	Estimation of additional cost for network operation.
Product	
Stability	Attributes of the product that bears on the stability of the product.
Ownership	Attributes in relation to intellectual property rights.
Deliverables	Detail of the out of the box and post-implementation deliverables.
Parameterization / Customization	Attributes in relation to the initial effort required for the product to operate.
Guarantees	Detail of the guarantees provided over the product.

Other approaches (please refer to [7, 8] for detailed description) also address the need of non-technical attributes to be considered in evaluation and selection, however,

their proposals can be considered non-structured a subset of our approach. We claim that technical and non-technical aspects shall be dealt similarly using a common framework instead of divided assets. To achieve this goal, we propose to extend the ISO/IEC 9126-1 catalog of quality factors with non-technical factors following the same layout than in this standard. In particular, as done in the standard, we fix just the two higher levels of the hierarchy, avoiding excessive prescription of the proposal. We call this catalog the non-technical extension of the ISO/IEC catalog (NT-ISO/IEC catalog for short) to distinguish it from the previous one (Fig. 1):

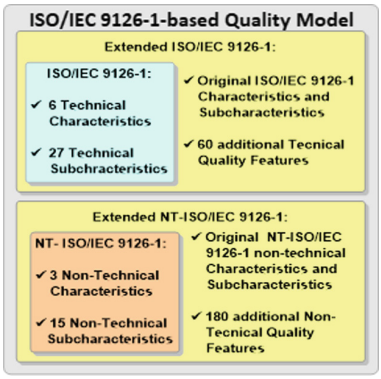

Fig. 1. The different catalogues found.

2.3 OSS Adoption Strategies

The concept of strategy comes from the Greek 'strategos' to denote 'leadership'. An OSS Adoption Strategy is the way in which an organization incorporates OSS as part of its customer offering. In [14], authors describe six different OSS adoption strategies which in the following paragraphs, are presented in ascendant order according to the strength of the relationship between the adopter and its FOSS Developer Community (FOSS-DC). For instance, in Release strategy, there is not relationship with FOSS-DC. In the following bullets, the benefits, and requirements are presented for each strategy.

- **Release:** It implies that the organization releases personalized software as FOSS but does not care whether an OSS community takes it up or forms around it. Additionally, no FOSS-DC is involved. The purpose of this strategy is to develop FOSS (with technical quality) for own use, and offer it to organizations with similar requirements in order to: (a) standardize its subjacent processes; or (b) spread good practices, or (c) reduce cost and risk by avoiding the development. The first two may be related to improving the market, and the third may be focused on strengthening the corporate group companies. In this sense, an example of this strategy can be observed in the public sector, where public entities make their software available to others, releasing it under an OSS license. This strategy

demands a minimum involvement with FOSS-DC and the organization does not care for FOSS evolution or maintenance.

- **Acquisition:** This strategy refers to the use of existing FOSS code without contributing to its OSS project/community. Because the adopter organization does not necessarily has, an interest in new releases of the FOSS component, the involvement with the community is minimum after obtains the software and its documentation. In some cases, other companies or freelancers can provide the FOSS component. The specific benefits are that it requires a minimum involvement with FOSS-DC, and does not care about FOSS evolution or maintenance.
- **Integration:** It involves the active participation of an organization in an FOSS community (to share and co-create FOSS) but not necessarily leading or influencing it. The organization establishes a relationship with a new external stakeholder: The Support Forum, from which obtains bug fixes that not necessarily are present in the next version that the FOSS Community releases for the component.
- **Fork:** It means to create an own independent version of the software that is available from an existing FOSS project or community. The FOSS-DC is forked too. This strategy is applied by an organization that decides to continue the (generally critical) FOSS component and FOSS-DC evolution for its own account, to achieve specific requirements.
- **Takeover:** In this strategy, the organization attracts an existent FOSS community to support its business activity. The creation of its own FOSS-DC pursues to 'take the control' of critical software development.
- **Initiative:** It is oriented to initiate an FOSS project and to establish a community around it. The organization creates its own FOSS-DC, because requires to 'take the control' of critical software development.

3 Searching for Relevant NT-Quality Attributes for OSS

Although FOSS has been increasingly used, particularly in the last decade, critical success and failure factors, risks, benefits, and barriers associated with this paradigm have not been deeply analyzed. The lack of knowledge in these relevant issues hampers FOSS adoption both in private and public industries. In 2018, we performed a systematic literature review (SLR) to gain understanding of how these factors affect FOSS implementation. 11 previous SLR where identified, none of them addressing these issues, which endorses our claim that more study is required in the field.

The SRL was conducted following Kitchenham's [15] methodological guidelines. To be rigorous, a protocol was established to search and collect data from the existing FOSS literature. Research question included in the protocol where:

Q1: Which success and failure factors have been documented in the adoption of FOSS in the industry?
Q2: Which are the main risks identified in the adoption of FOSS?
Q3: Which are the perceived benefits in the adoption of FOSS?
Q4: In which software domains has the adoption of FOSS been successful?
Q5: Which are the main factors that prevents organizations from adopting FOSS?

The generic search string was constructed using the words *Open Source Software*, *OSS*, *FOSS*, *FLOSS* and *Freeware* as synonyms; to complete the string we added the words *risk*, *failure*, *success*, *barriers*, *advantages*, *benefits* and *application domain*. The search string was built using the "OR" and "AND" logical operators. Search string was constructed to address titles, abstracts and related words: *Title: (Open Source Software OR OSS OR FOSS OR FLOSS OR Freeware) AND Abstract: (Open Source Software OR OSS OR FOSS OR FLOSS OR Freeware) AND (risk OR failure OR success OR barriers OR Advantages OR Opinion OR Impediments OR FLOSS).* The string was properly adapted to the characteristics of the repositories where searches where conducted: IEEE Xplore, ACM, Springer Link, and WOS.

A total of 782 documents were retrieved from the repositories; titles and abstracts where stored in an excel sheet. In a first refinement, only academic articles written in Spanish or English, with methodological basis (e.g. experiment, case study, Systematic Reviews, Systematic Mappings), with at least 5 references in google scholar where included in the study.

Duplicated articles were excluded and we limited the study to written after 1998 of the following types: Journals, Conferences, Magazines, Technical Reports and Books. In a second refinement, all abstracts were reviewed in full, by at least two researchers of the team, to select which were relevant and sound for the study. Documents marked as "relevant" or "maybe relevant" for at least one of the researchers were selected to be full text redden. The documents were extracted, to read the summary, to see if the article could contribute with the investigation, once read, the researchers granted ratings of: "Yes", "No" and "Maybe". Only articles written in Spanish and English where included in the study. At the end, only 155 documents were selected to be full text reviewed. From those, only 92 papers answered, at least, one of the research questions. In addition, backward snowballing techniques allowed for the identification of 33 additional references; only 17 resulted relevant for the study. A table was specifically designed for data extraction and implemented using a spreadsheet editor. Table was refined/extended through the study, to include all information extracted from selected papers, relevant to answer research questions.

The most relevant success factors identified in the study are: (a) Management: software evolution, licensing, project management; (b) Community: community activities and communication; (c) Support: internal and external; (d) training: staff training; (e) Policies: IT enterprise policies.

Regarding failure factors, the most relevant are: (a) Human Resources: IT provider, end users, technical users, development and community; (b) Competitiveness: Market characteristics; (c) Policy: OSS and IT adoption policies absence.

Several risks were also identified, the most relevant are: (a) Laws: assume the responsibility; (b) Software: complexity, processes, tests, support, technology, integration, and architecture; (c) Human resources: developers, community, experts, management, lack of guarantee.

Several benefits have also been identified in the study: (a) Flexibility: source code availability; (b) human resources: community, providers and contributions.

The application domains identified in the study are: (a) system software: Operating systems, server software; (b) application software: mobile apps, office, medical, commercial software, educational, browsers, databases managers, automation and control.

Finally, several barriers for the adoption of FOSS have also been identified: (a) human resources: training, developers, experience, uncertainty, motivation, lack of trained professionals; (b) software: quality, migration, product, version, repositories.

Table 2 presents a relation among the factors and the percentage of papers relating them with the main issues considered in the SLR, namely success and failure factors, risks, benefits, and barriers associated with the FOSS paradigm.

Table 2. Percentages of documents relating factors with categories in the SLR.

Factor	Success	Failure	Risks	Benefits	Domain	Barriers
Management	20%	-	-	-	-	
Community	18%	-	-	-	-	
Support	11%	6.10%	-	-	-	4.60%
Training/learning	10%	-	-	1.80%	-	
Policy	10%	12.10%	-	-	-	5.60%
Costs	8%	-	9.50%	-	-	4.60%
Software	7%	-	23%	8.60%	-	15.70%
System Software	-	-	-	-	53.80%	-
Application Software	-	-	-	-	43.10%	-
Software testing	-	6%	-	-	-	-
Enterprise and organizations	6%	-	-	8.20%	-	9.30%
Control	4%	-	1.40%	-	-	-
Human Resources	4%	39.40%	18.60%	19.50%	-	38.90%
Public sector management	1%	-	-	-	-	-
Information technologies	0.50%	-	-	-	-	-
Perceived values	0.50%	-	-	-	-	-
Competitiveness	-	18.20%	-	-	-	-
Planning	-	9.10%	-	-	-	-
Licensing	-	6.10%	-	0.50%	-	4.60%
Usability	-	3%	-	-	-	-
Laws	-	-	29.70%	-	-	-
Security	-	-	9.50%	5.40%	-	7.40%
Adoption	-	-	4.10%	-	-	-
Model	-	-	1.40%	-	-	-
Scalability	-	-	1.40%	-	-	-
Documentation	-	-	1.40%	-	-	1.90%
Profitability	-	-	-	16.70%	-	-
Flexibility	-	-	-	20.80%	-	-
Development	-	-	-	7.20%	3.10%	-
Time saving	-	-	-	5.40%	-	-
Component Integration	-	-	-	3.20%	-	2.80%
Innovation	-	-	-	1.80%	-	-
Hardware	-	-	-	0.90%	-	-
Compatibility	-	-	-	-	-	4.60%

* Main characteristics from each factor ave been shaded.

4 The Catalogue of NT-Quality Attributes for OSS

Similarly, to the catalogue in [12, 13], the *Extended FOSS NT-ISO/IEC 9126-1* quality catalog (see Table 3), has been structured with a three-levels hierarchical decomposition. At the first level, we have included four main Categories: External Context Actors

Table 3. Extended NT-ISO/IEC 9126-1 quality catalogue.

Code	Hierarchical Categorization	Description	No. Ref.	Meaning	ADOPTION STRATEGY					
					A	I	F	T	I	R
A	COMMUNITY									
A.1	FOSS-DC									
A.1.1	Perceived image	The negative impression about service and future support of FOSS when the payment option is dominant.	3	Barrier		x	x	x	x	
A.2	FOSS-DC SUPPORT									
A.2.1	Feedback	The FOSS-DC feedback implies a set of community practices focused on reinforce the relationships with adopters.	17	Success		x	x	x	x	
A.2.2	Lack of Support	The lack of support that FOSS-DC offer to adopter	5	Barrier		x	x	x	x	
A.2.3	Continuity of project	Loss of interest by part of the developers or FOSS-DC, which causes that FOSS project disappears.	2	Faliure		x	x	x	x	
A.2.4	Continuity of community	Loss of interest by part of the developers or FOSS-DC, which causes that FOSS-DC disappears.	2	Faliure		x	x	x	x	
A.2.5	Continuity of software development orientation	Loss of interest by part of the FOSS-DC, which conduct the software development in a different direction than the adopter	3	Faliure		x				
A.2.6	Version control	The FOSS-DC manages a versioning of FOSS and its documentation	3	Success	x	x	x	x	x	x
A.2.7	Version frequency	The FOSS-DC versioning FOSS with high frequency. It can generate a problem	2	Risk	x	x	x	x	x	x
A.3	FOSS-DC SUPPORT IMPACT ON ADOPTER									
A.3.1	Development effort reduction	The FOSS support contributes to reduce the development time	17	Success		x	x	x	x	
A.3.2	Accessibility to public information	FOSS can benefit to adopters and citizens in general, fostering the access to public information.	2	Success	x	x	x	x	x	x
A.4	UNIVERSITIES, RESEARCH INSTITUTES, AND COMMERCIAL ENTERPRISES									
A.4.1	Cooperation	Cooperation activities between research staff and development staff	17	Success	x	x	x	x	x	x
A.5	EXTERNAL DEVELOPERS									
A.5.1	Participation	Voluntary participation of external developers.	5	Success	x	x	x	x	x	x
B	ORGANIZATIONAL ISSUES									
B.1	RESOURCE MANAGEMENT POLICIES									
B.1.1	Resource optimization	In the vertical application development, the open source models have promote a cost reduction through reuse, collaboration, and shared code.	18	Benefit	x	x	x	x	x	x
B.1.2	No dependence on software providers/vendors	FOSS-DC support to avoid traditional software providers/vendors lock-in (dependencies of negotiation conditions imposed by the supplier)	11	Benefit	x	x	x	x	x	x
B.1.3	Cost management	In order to reduce the costs, the adopter identifies the components of TCO (Total Cost of Ownership) correspondent to FOSS adoption process	5	Risk	x	x	x	x	x	x
B.2	QUALITY POLICIES									
B.2.1	Security strengths	FOSS brings more security than proprietary software because FOSS is audited constantly by external entities.	12	Benefit	x	x	x	x	x	x
B.2.2	Security weakness	Possible vulnerabilities in security schema, which can be originated in the use of libraries of open source, architectural weakness, human errors, incorrect code modifications, among others)	6	Risk		x	x	x	x	
B.2.3	Reliability	FOSS is more reliable than proprietary software because FOSS has the collaboration of developers around the world.	4	Benefit	x	x	x	x	x	x
B.3	INNOVATION POLICIES									
B.3.1	Innovation promotion	The FOSS adoption contributes to increase the innovation	3	Benefit	x	x	x	x	x	x
B.4	LEGAL AND REGULATORY FULFILLMENT									
B.4.1	Lawful knowledge	This knowledge is required to manage adequately some legal and regulatory FOSS issues, mainly related to licensing, Intellectual Property (IP) and Copyright.	---	Barrier	x	x	x	x	x	x
B.4.2	Licensing	Problems of license use and license compatibility	5	Barrier	x	x	x	x	x	x
B.5	TRAINING POLICIES									
B.5.1	Cost of FOSS training	The training investment required by an FOSS project can constitute an entry barrier, if it has not an adequate estimation and management	2	Barrier	x	x	x	x	x	x

(continued)

Table 3. *(Continued)*

B.6	DECISION-MAKING POLICIES									
B.6.1	Competitiveness	The FOSS projects must have the minimum competitiveness level in order to make viable them, comparing with proprietary software. The decision about initiate or not an FOSS project considers its competitiveness.	4	Faliure	x	x	x	x	x	x
B.6.2	Strategic Staff Commitment	The support of Strategic Staff to FOSS project constitutes a critical success factor, in particular when the risk or uncertainty levels are high.	2	Success	x	x	x	x	x	x
B.7	KNOWLEDGE MANAGEMENT POLICIES									
B.7.1	Organization's experience with OSS	Knowledge and improvements obtained by the organization from previous technical experiences about OSS.	---	Benefit		x	x	x	x	
B.7.2	Staff's OSS experience outside the organization	Knowledge and improvements that the staff has had outside the organization.	---	Benefit		x	x	x	x	
B.7.3	Previous Related Knowledge	This knowledge enables the organization to assimilate and use new outside knowledge. It is both commonality (required to facilitate the communication) and individual (required to foster the diversity).	---	Success	x	x	x	x	x	x
C	GOVERNMENT POLICIES									
C.1	ECONOMIC DEVELOPMENT									
C.1.1	FOSS Economic Postioning	Refers to the current situation of FOSS in the economic development of the country	6	Benefit	x	x	x	x	x	x
C.1.2	FOSS Economic sustainability	Refers to the trend of FOSS in the economic development of the country	---	Success	x	x	x	x	x	x
C.2	INNOVATION POLICIES									
C.2.1	FOSS adoption promotion policies	Policies designed by government to coordinate the innovation development	2	Success	x	x	x	x	x	x
C.3	PUBLIC POLICIES									
C.3.1	Incentives to TICs' adoption	The Government encourages organizations to test, adopt, and use new technologies.	3	Support	x	x	x	x	x	x
D	STAFF CAPACITIES DEVELOPMENT									
D.1	CHANGE MANAGEMENT									
D.1.1	FOSS acceptance	The FOSS adoption is affected by staff level of preparation and motivation to change	6	Barrier	x	x	x	x	x	x
D.1.2	FOSS Skills Development	The inner staff has not the skills required to support the FOSS adoption process (e.g. collaborative teams, crowdsourcing, agile paradigm, etc.)	2	Barrier		x	x	x	x	
D.1.3	Technical and Business training	The FOSS adoption process requires that the staff has acquired a minimum knowledge (concepts and practices)	6	Success		x	x	x	x	x
D.1.4	Soft Skills	**Learning Capacity:** to use and renew available knowledge. **Absorptive Capacity:** to identify, acquire, assimilate, transform, and exploit knowledge. **Networking Capacity:** to establish, maintain, and expand relationships among actors to exchange knowledge, value, and collaboration. **Disseminative Capacity:** facilitates the external and internal knowledge and expertise transfering.	---	Success	x	x	x	x	x	x
D.2	ORGANIZATIONAL CULTURE									
D.2.1	Openness and Flexibility	Openness to new ideas and approaches to solve problems, facilitates the creative solutions generation, discovering new paths to achieve them, and decreases the change resistance.	---	Success		x	x	x	x	
D.2.2	Risk Tolerance	The organizational risk tolerance implies the willingness to engage in projects that can have uncertain outcomes.	---	Success		x	x	x	x	
E	FOSS SUPPORT									
E.1	SUPPORT AVAILABILITY									
E.1.1	Inner technical support	The inner staff should be enough to solve user requirements	3 4	Barrier Risk		x	x	x	x	
E.1.2	Source code availability	Availability of source code to review and/or modify.	13	Benefit		x	x	x	x	
E.1.3	Business-IT alignment	Alignment between Information Technologies and Business Model, i.e., to what extent the IT operations support the business goals, business strategy and mission.	---	Success	x	x	x	x	x	x
E.1.4	Strategic commitment for FOSS adoption	Strategic commitment to FOSS: initiative sponsoring and basic requirements (e.g., budget, resources management support, business process management support)	---	Success		x	x	x	x	
E.2	SUPPORT IMPROVEMENTS									
E.2.1	Efficiency	The efficient use of resources destined to FOSS adoption	6	Benefit		x	x	x	x	
E.2.2	Integration	Applications with open source code can be used and modified to be integrate according to particular requirements.	4	Benefit	x	x	x	x	x	x
E.2.3	Compatibility	A vertical change from proprietary software to FOSS, can originate compatibility problems with previous formats.	5	Barrier	x	x	x	x	x	x

Relationships, Organizational Issues, Government Policies, Staff Capacities Development, and FOSS Support. We have also proposed a refinement of the *Support* sub-characteristic, categorized under the *Supplier* Characteristic, in the original Extended NT-ISO/IEC 9126-1 quality catalogue, to make it more suitable for the FOSS evaluation context. This sub-characteristic has been renamed FOSS Support when used for FOSS evaluation purposes.

Each characteristic has been further decomposed with two additional hierarchical levels, the first one including sub-characteristics and the second one including attributes, which are observable quality features associated to one or more measures, relevant for the statement of organizational quality requirements and the assessment of FOSS quality, during evaluation.

We have also incorporated additional columns to shape the catalog structure. The first one "Meaning to Adopter", allows for traceability with the main category of factors in the SLR from which the attribute has been identified, namely, Benefit, Entry Barrier, Risk, Success Factor, Fail Factor, or Support. In addition, we have also considered the six OSS adoption strategies proposed in [14] (see Sect. 2.3). These strategies model the strength of the relationship between the adopter and its more representative external context actor: the FOSS Developer Community (hereafter called FOSS-DC). Each attribute has a different relevance (or even no relevance) depending on which adoption strategy is selected by the organization. For instance: if the organization opts for the *Acquisition Strategy*, the attributes *A.2.1 FOSS-DC feedback* and *A.2.2 Lack of FOSS-DC Support* are not applicable, because, in the practice, the FOSS component is used as-is, without modification, and in consequence, FOSS-DC feedback or support is not required. On the other hand, if the adopter opts for the *Fork Strategy*, the active role of FOSS-DC in terms of feedback and support is critical.

The attributes *A.2.6 Version Control*, and *A.2.7 Version Frequency* already appear in the extended NT-ISO/IEC 9126-1 quality catalog (under the product/stability sub-characteristic), but our study has shown that it may have greater impact in FOSS adoption processes, in terms of risks introduced to the project. Version release frequency is usually higher and more flexible in FOSS, since it responds to the dynamics of the FOSS-DC. The increase in version release frequency hampers ability of technical staff to incorporate (parametrize, adapt, modify, test, integrate, and deploy) new versions of FOSS, with significant impact maintainability of IS.

Attributes *B.1.3 Cost Management* and *B.5.1 Cost of Training* are included in the original extended NT-ISO/IEC 9126-1 quality catalogue, but in this work we present a refinement with measures specific for FOOS. In the same way, attributes categorized under Quality Policies (*B.2.1 Security strengths*, *B.2.2 Security weakness*, and *B.2.3 Reliability*) are part of original extended Technical ISO/IEC quality catalogue, but here they are oriented to FOSS adoption support.

In addition, the original extended NT-ISO/IEC catalogue includes a significant amount of attributes intended for the evaluation of licensing schemas. In our context, however, we need to be specific in relation to aspects such as the access, manipulation and redistribution of open source code. Specific measures for each attribute have been defined (not included in this paper due to space limitations, see Table 4 for an excerpt).

Table 4. Except of measures for extended quality catalogue

Code	METRIC				Requires Baseline
	Name	Description	Measurement	Acceptable values	
A	COMMUNITY				
A.1	FOSS-DC				
A.1.1	Support image perceived	Identifies if adopter has a positive or negative image of FOSS-DC support, in a considering where payed options are the majority.	Image perception identified in adopters	Yes / No	No
A.2	FOSS-DC SUPPORT				
A.2.1	Feedback support	The support provided by community in form of patches, documentation, business best practices description, software parametrization, instalation/integration guidelines, and so on.	Effective Support provided by FOOS-DC	Yes / No	No
A.2.2	Lack of support from FOSS-DC	Occasions when the OSS-DC brings insufficient or inexistent support to software products that adopter works with	Responsesw from the FOSS-DC with no sufficient information about FOSS component	Positive integer values	No
A.2.3	FOSS-DC continuity	Identifies if FOSS-DC maintains the support to specific software project	Number of versions released, patches, or effective support activities for the specific FOSS softwar,e in the last period.	Positive integer values	No
A.2.4	FOSS support continuity	Identifies if FOSS-DC maintains its activity	Number of versions released, patches, or effective support activities for any of its FOSS projects, in the last period.	Positive integer values	No
A.2.5	Software development orientation continuity	The FOSS-DC maintain its development and support for the software, with no modifications in relation to the adopter expectation	Continuity verified through the releases, patchs, documentation, and functionalities of FOSS	Yes / No	No
A.2.6	Version control	The FOSS-DC manages its versioning schema for OSS and its documentation	Availability verified in a location referred by FOSS-I	Yes / No	Yes

5 Validating the Catalogue in Practice

In 2008, the Government of "Country Blinded" issued a specific order to enforce the FOSS adoption in the Public Administration. During the first years of the ruling, the efforts for the adoption were isolated and chaotic, causing some disappointment and a kind of misgiving about the regulation.

Table 5. Validation cases.

Verification Case		DESCRIPTION	TYPE	SIZE/SCOPE	GOAL	REQUIREMENT	ACHIVEMENT
	1	VrC1 is a public company. It's scope of action is defined as a strategic public service. Its main purpose is the generation and provision of electrical service and this must respond to the principles of mandatory, generality, uniformity, responsibility, universality, accessibility, regularity, continuity and quality.	PUBLIC	BIG / NATION WIDE SCOPE	Migrate an mail server plataform from an Exchange Architecture to an Zimbra FOSS architecture	1. Implement a seven nodes plataform for balancing the mail traffic that comes from the the mail domains that are imple-mented in the Ministerio de Electricidad. 2. Fine Tune the plataform in order to protect the mail boxes from no desired messages –SPAM-. 3. Integrate Zimbra FOSS authentication with a Privative Authentication Plataform. 4. Integrate Zimbra FOSS whith an li-censed mail filter solution they already have imple-mented.	After three years, users have adopted the new plat-form successful-ly. Technicians have learnt to maintain and correct any issues that have come during operation. They don't ask for external support
	2	VrC2 is a public institution whose mission is to *"distribute and market electric energy in optimal technical and economic conditions to meet the needs of our customers as a basic foundation of society, in accordance with the current legal framework, seeking social benefit, efficient use of energy and sustained economic equilibrium, through processes of continuous improvement and protection of the environment."*	PUBLIC	SMALL/ PROVINCE SCOPE	Mail Server's Plataform using Zimbra FOSS implementation and maintenance	1. Implement a single node plataform for the institutional mail service. 2. Configure the DNS based mail authentica-tion registries. 3. Train the insitution's TIC personal about mail service and basic Linux Sys-tem Administra-tion concepts.	After a year and a half, users have adopted the new plat-form; but still have some problems about Mail service's best practices. Technician have some ma-jor problems when trouble-shooting common operating issues. The need for a higher support level is manifested.
	3	VrC3 is an agency in charge of control and regulate the protection and improvement of animal health, plant health and food safety.	PUBLIC	MEDIUM / PROVINCE SCOPE	Implement an Open Source environment for controlling user's workstation software life cycles, centralized authentication, and deploy office software according the job in four different areas	1. Implement an alternative to MS Active Direc-tory for control-ling the domain' workstations. 2. Analyze and Deploy an OSS alternative to MS Windows for the users' workstations. 3. Analyze and deploy an OSS alternative to the user's workstations software accord-ing to the need of the area.	After six months, users have adopted the new operating system, even when there are some applications that are completely new for them. Technician have experi-enced one major problem they couldn't solve by their self. They have an aditional contract for higher support level for one year.
	4	VrC4 is a private company at the service of the Ecuadorian community for 40 years, developing healthy, natural and functional foods for all its consumers.	PRIVATE	BIG / NATION WIDE SCOPE	Mail Server's Plataform using Zimbra FOSS implementation and maintenance	1. Implement a single node plataform for the institutional mail service. 2. Configure the DNS based mail authentica-tion registries. 3. Train the insitution's TIC personal about mail service and basic Linux Sys-tem Administra-tion concepts.	After four years, users have adopted the new plata-form successfully. Some SPAM problems have arised, but they can be solved easly. Technician have earned experience in troubleshooting operating issues after the training sessions. They have an addi-tional support contract for consumed hours.

To address this issue, a group of engineers working in the public sector, conducted a project to develop a set of artifacts to guide the adoption of open technologies in the country. The results of this project include a catalog of templates and guidelines specifically designed for this purpose. With the support of these deliverables, since 2016 the number of projects to migrating to FOSS alternatives in the country has increased both, in the public and private sectors.

Table 6. Impact of NT-quality attributes in validation cases.

Non-Technical Quality Features		Verification Case				Total Impact
		1	2	3	4	
A	COMMUNITY					
A.1	OSS-DC					
A.1.1	Perceived image	5	5	3	5	18
A.2	OSS-DC SUPPORT					
A.2.1	Feedback	5	5	5	5	20
A.2.2	Lack of OSS-DC Support	5	N/A	3	N/A	8
A.2.3	Continuity of OSS-DC	5	3	3	4	15
A.2.4	Continuity of FOSS-DC	N/A	N/A	2	N/A	2
A.2.5	Continuity of Software development orientation	N/A	N/A	N/A	N/A	0
A.2.6	Version control	5	5	5	5	20
A.2.7	Version frequency	2	1	1	2	6
A.3	OSS-DC SUPPORT IMPACT ON ADOPTER					
A.3.1	Development effort reduction	2	5	5	4	16
A.3.2	Accessibility to public information	5	5	5	5	20
A.4	UNIVERSITIES, RESEARCHINST AND COMM ENTERPRISES					
A.4.1	Cooperation	N/A	N/A	N/A	N/A	0
A.5	EXTERNAL DEVELOPERS					
A.5.1	Participation	2	N/A	N/A	N/A	2
B	ORGANIZATIONAL ISSUES					
B.1	RESOURCE MANAGEMENT POLICIES					
B.1.1	Resource optimization	5	5	5	5	20
B.1.2	No dependence of software providers/vendors	2	4	2	2	10
B.1.3	Cost management	3	N/A	N/A	5	8
B.2	QUALITY POLICIES					
B.2.1	Security strengths	5	N/A	5	5	15
B.2.2	Security weakness	5	N/A	5	5	15
B.2.3	Reliability	5	N/A	4	5	14
B.3	INNOVATION POLICIES					
B.3.1	Innovation promotion	2	N/A	N/A	N/A	2
B.4	LEGAL AND REGULATORY FULFILLMENT					
B.4.1	Licensing	2	5	5	5	17
B.5	TRAINING POLICIES					
B.5.1	Cost of OSS training	3	4	2	5	14
B.6	DECISION-MAKING POLICIES					
B.6.1	Competitiveness	N/A	N/A	N/A	N/A	0
B.6.2	Strategic Staff Commitment	5	5	5	5	20
C	GOVERNMENT POLICIES					
C.1	ECONOMIC DEVELOPMENT					
C.1.1	OSS Economic positioning	5	5	5	3	18
C.1.2	OSS Economic sustainability	5	5	5	4	19
C.2	INNOVATION POLICIES					
C.2.1	OSS adoption promotion policies	5	5	5	3	18
C.3	PUBLIC POLICIES					
C.3.1	Incentives to TICS adoption	5	5	5	2	17
D	STAFF CAPACITIES DEVELOPMENT					
D.1	CHANGE MANAGEMENT					
D.1.1	OSS acceptance	5	4	4	5	18
D.1.2	OSS Skills development	4	3	2	4	13
D.1.3	Technical and Business training	4	3	2	5	14
E	OSS SUPPORT					
E.1	SUPPORT AVAILABILITY					
E.1.1	Inner Technical support	3	2	1	3	9
E.1.2	Source code availability	1	1	1	1	4
E.2	SUPPORT IMPROVEMENTS					
E.2.1	Efficiency	5	5	5	5	20
E.2.2	Integration	5	5	3	3	16
E.2.3	Compatibility	5	5	5	3	18

In the last two years we got involved in four of such projects, see Table 5 for some details. We use the knowledge gained in these projects to validate the relevance of the attributes included in the catalogue introduced in Sect. 4.

In Table 6, for each verification case, we have assessed the impact of each of the non-technical quality attributes described in the catalogue. Rows include the non-technical quality attributes in the catalogue whilst columns represent} the verification cases. We have used Likert's five levels scale to assess the relevance of each attribute in the projects, (one for the lowest and five for the higher impact).

Last column presents the total impact for each attribute, calculated as the sum of impacts in the four projects. 24 of the 44 non-technical attributes included in the catalogue (54%), were graded with a relevance higher tan of 15 over 20 possible points; 7 (16%) additional attributes, scored between 10 and 15 points, which makes evident the relevance of the attributes in the catalogue for these kind of projects.

It is important to mention that some attributes in the A category were not evaluated. This is due to the fact that, these organizations had already adopted some FOSS and the focus of the project was on upgrading and giving it a better maintenance to the software instead of adopting new software. On the other hand, it is important to consider that, in some cases, public institutions are forced by ministerial decrees to implement specific FOSS products, without chance to evaluate alternatives.

Organizational/Quality Policies attributes have been valuated with the highest scores. It can be argued that OSS products could be more vulnerable due to their open code nature. Consequently, security policies are required to minimize related threats. In contrast, some factors e.g., the *independence of the provider*, are perceived as means to increase protection of personal data, since code can be explored to prevent the existence of "backdoors", used for some manufacturers to introduce remote control codes into the software. In addition, open standards, that grant the freedom of users to exchange information with a FOSS community, are perceived as an important security factor when considering FOSS adoption.

Licensing is another interesting factor to consider. In the context of "Country Blinded", most Chief Technical Officers (CTO) still perceive Open Source as synonym of free software. Therefore, it is evident the lack of accurate economic feasibility and resource allocation studies in the projects.

In three of the four cases presented in Table 5, some consultants tried to sell expensive licensing schemas with the promise of "easiness" in FOSS product's troubleshooting. In fact, lack of technical skills and knowledge has been identified (in the study presented in Sect. 3), as one of the main barriers and risks in the adoption of FOSS. At the end, this fear factor causes some enterprises to purchase expensive licensing for premium functionality.

Despite the fact that attributes in OSS-DC SUPPORT category have the highest impact, it is important to consider attributes in the STAFF CAPACITIES DEVELOPMENT category. Some anthropologic aspects e.g., soft skills, risk acceptance, openness and flexibility ae included in this category. These attributes are highly important in the context of FOSS adoption projects, since successful adoption largely relies in technicians' attitude. Cultural change of technical and end users needs to be carefully addressed in this kind of project.

6 Conclusions and Future Work

The traditional approach to evaluate software components rely on technical requirements widely described in diverse quality models, but does not consider the high relevance of non-technical factors. In this research work, we have proposed a Non-Technical attributes catalogue, which can be used join with Technical ones, to evaluate FOSS Software Components considering the way in which they were adopted by the organization. These requirements are an extension of preexisting software quality models, and were obtained from SLR as a basis to identification success and failure factors, risks, benefits, main software domains, and entry barriers related to FOSS adoption. These requirements were structured in a three level hierarchical catalogue of non-technical attributes, covering from strategic to operative issues in FOSS evaluation context. Finally, the relevance of non-technical attributes was validated in relation to four industrial FOSS adoption cases taken from the experience of our team. Even when some of the attributes couldn't be evaluated, we confirm the suitability of the catalogue when evaluating and selecting a FOSS product. As future work, we will focus on refinement and validation of the measures set for each non-technical attribute in the catalogue, in order to make feasible an objective and systematic evaluation of FOSS component adoption.

References

1. ISO/IEC Standard 25000. Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE (2005)
2. Radulovic, F., García-Castro, R.: Extending software quality models - a sample in the domain of semantic technologies. In: SEKE 2011 (2011)
3. Kläs, M., Lampasona, C., Munch, J.: Adapting software quality models: practical challenges, approach, and first empirical results. In: EUROMICRO-SEAA 2011 (2011)
4. Lampasona, C., et al.: Software quality modeling experiences at an oil company. In: ESEM 2012 (2012)
5. Atos: Method for qualification and selection of open source software (QSOS) version 2.0. http://backend.qsos.org/download/qsos-2.0_en.pdf
6. Wasserman, A.I., Pal, M., Chan, C.: Business readiness rating for open source. In: Proceedings of the EFOSS Workshop, Como, Italy, 8 June 2006 (2006)
7. Soto, M., Ciolkowski, M.: The QualOSS open source assessment model measuring the performance of open source communities. In: Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement, 15 October 2009 (2009)
8. Aversano, L., Tortorella, M.: Applying EFFORT for evaluating CRM open source systems. In: Caivano, D., Oivo, M., Baldassarre, M.T., Visaggio, G. (eds.) PROFES 2011. LNCS, vol. 6759, pp. 202–216. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21843-9_17
9. International Organization for Standarization. ISO Standard 9126: Software Engineering – Product Quality, part 1. International Organization for Standarization (2001)
10. Franch, X., Carvallo, J.P.: A quality-model-based approach for describing and evaluating software packages. In: Proceedings 10th IEEE Joint Conference on Requirements Engineering (RE) (2002)

11. Carvallo, J.P., Franch, X.: Using quality models in software package selection. IEEE Softw. **20**(1), 34–41 (2003)
12. Carvallo, J.P., Franch, X., Quer, C.: Managing non-technical requirements in COTS components selection. In: Proceedings of 14th IEEE International Requirements Engineering Conference (RE 2006) (2006)
13. Carvallo, J.P., Franch, X., Quer, C.: Towards a unified catalogue of non-technical quality attributes to support COTS-based systems lifecycle activities. In: Proceedings of 6th IEEE International Conference on COTS-Based Software Systems (ICCBSS 2007) (2007)
14. López, L., et al.: Adoption of OSS components: a goal-oriented approach. Data Knowl. Eng. **99**, 17–38 (2015)
15. Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. Technical report EBSE-2007-01 (2007)