A focus group for operationalizing software sustainability with the MEASURE platform

Nelly Condori-Fernandez
Universidade da Coruña, Spain
Vrije Universiteit Amsterdam, The Netherlands
n.condori.fernandez@udc.es
n.condori-fernandez@vu.nl

Alessandra Bagnato
Softeam
Paris, France
alessandra.bagnato@softeam.fr

Eva Kern
Leuphana University Lueneburg
Environmental Campus Birkenfeld
Germany
eva.kern@leuphana.de

***Abstract*—Measuring the sustainability of software products is sill in the early stages of development. However, there are different approaches how to assess sustainability issues of software and its engineering - including metrics with a practical orientation as well as more theoretical models covering software sustainability. As an example for one step in moving forward bringing existing approaches together, the paper presents a focus group study conducted to find out in which extent the quality attributes related to the technical sustainability can be measured by using existing metrics available at the MEASURE platform. Our first results show that the extent of measurability varies across the software development phases. Functional correctness, robustness, maturity, and testability are the most measurable quality attributes.**

***Index Terms*—technical sustainability, measurement, focus group, software metrics**

I. Introduction

Assessment based on the notion of sustainability, as a software quality property, is still emerging and poorly understood [1]. Consequently, how software should be assessed against sustainability concerns is still immature even though it is attracting increasing attention from both research and practice.

This is especially the case when it comes to technical sustainability of software. According to [2], [3] technical sustainability has the central objective of long-time usage of systems and their adequate evolution with changing surrounding conditions and respective requirements. However, so far, there is a knowledge gap how to transfer theoretical knowledge into practical routines [4]–[7]. Here, software measurement can help in creating transparency into software properties and in providing information on sustainability issues of software to developers. Sustainability issues of software are discussed in more details in [8]–[10]. Thus, in the following paper, we will concentrate on the presentation of bringing the metrics of a measurement platform - the MEASURE platform - and aspects of a Software Sustainability Model [11] together. Doing so, we bring practical and scientific approaches in assessing the technical sustainability of software products together.

The paper is structured as follows: Section II presents the MEASURE platform and the Software Sustainability Model. These are the basic information of the focus group workshop. The design of focus group, including a description of the participants, research questions, and methods, is introduced in Section III. Section IV illustrates the validity of the research done before summarizing and discussing the results of our study in Section V.

II. Background

A. The MEASURE platform

The MEASURE ITEA3 consortium (Softeam R&D, 2017) [12] aims to develop a comprehensive set of tools for automated and continuous measurement over all stages of the software development life cycle (specification, design, development, implementation, testing, and production). It includes the development of better metrics and ways to analyses the big data produced by continuous measurements, the validation of those metrics by the integration of the metrics and tools into running processes in various industrial partners, and the creation of decision support tools for project managers through the visualization of the collected data.

This European MEASURE ITEA 3 project develops a framework of metrics [13], [14] bottom-up with a list of industry partners and integrated them into a systematic structure to help creating a reference for companies to improve their assessment all phases of the software development life cycle metrics. MEASURE work is based on the OMGs Structured Metrics Metamodel (SMM) models [15]. The MEASURE platform consists of a web application that allows to deploy, configure, collect, compute, store, combine and visualize measurements by execution of software measures that may be defined according to the SMM specification. The MEASURE project can develop a body of knowledge that shows software engineers why, how and when to measure quality of process, products and projects. Nowadays, an emergent quality property of the software systems is sustainability. Although there is an urgent demand for innovative solutions and smart applications for a sustainable society worldwide, sustainability measurement and assessment is a big challenge. The MEASURE project developed a set of 150 metrics related to different aspects of software engineering. With the work done within this paper and the focus group we contribute to address sustainability under a multi-dimensional perspective on the entire software development life cycle. Figure 1 illustrates a typical dashboard in the MEASURE platform.

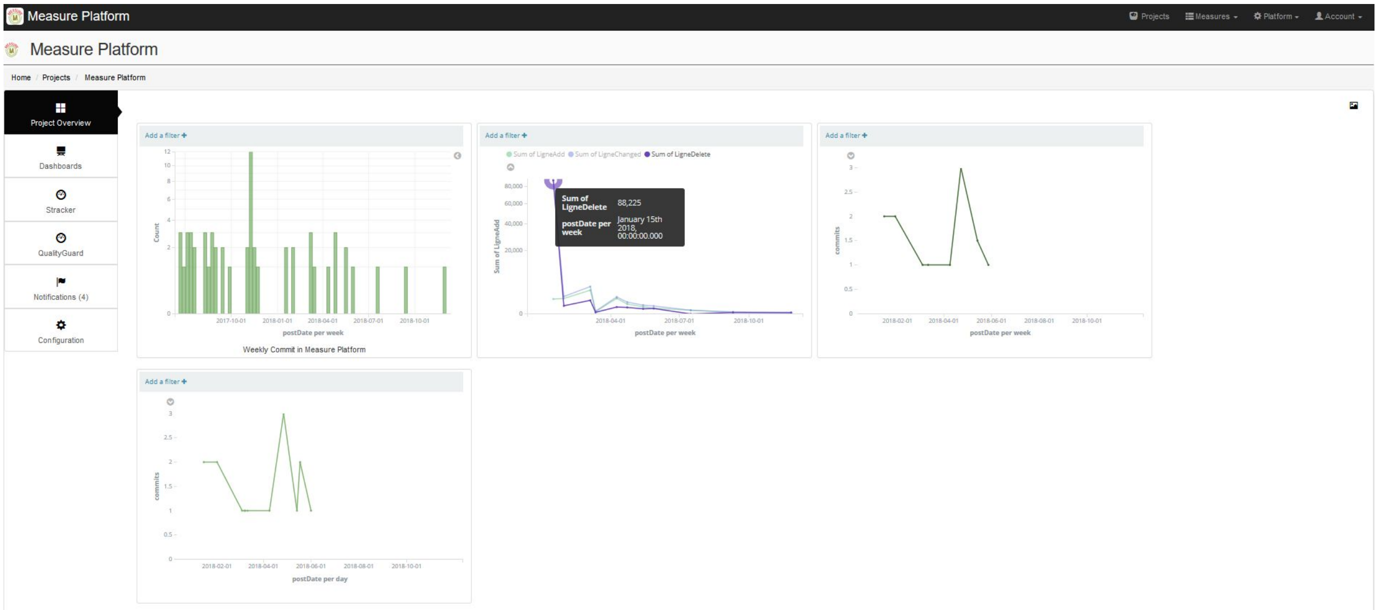


Fig. 1: Dashboard of the MEASURE platform

B. *The software sustainability-quality model*

Lago et al. [3] and Venters et al. [16] agree on defining software sustainability in terms of multiple and interdependent dimensions (e.g. economic, technical, social, environmental, individual). Several efforts have been put to define software sustainability in terms of quality requirements (e.g. [10], [16]–[19]). For instance, Condori-Fernandez and Lago [19] provided (i) a detailed characterization of each software sustainability dimension, which is a first step towards its respective operationalization, and (ii) a list of direct dependencies among the four sustainability dimensions: economic, technical, social, and environmental.

The *economic dimension* aims to ensure that software-intensive systems can create economic value. It is taken care of in terms of budget constraints and costs as well as market requirements and long-term business objectives that get translated or broken down into requirements for the system under consideration. The *social dimension* aims to allow current and future generations to have equal and equitable access to the resources in a way that preserves their socio-cultural characteristics and achieve healthy and modern society. The *environmental dimension* seeks to avoid that software-intensive systems harm the environment they operate in. And, the *technical dimension* is concerned with supporting long-term use and appropriate evolution/adaptation of software-intensive systems in constantly changing execution environment. Based on these definitions, quality attributes (QA) that contribute to the corresponding sustainability dimensions of software-intensive systems were identified [19]. As a result of this characterization per sustainability dimension in terms of quality attributes and identification of direct dependencies, a software sustainability-quality model was proposed, which can be found in [11].

III. Focus group study design

A. *Goal and research questions*

The goal of our focus group study, according to the Goal/Question/Metric template, is as follows:
Analyze metrics from the MEASURE platform and Software Sustainability-Quality Model [11]
for the purpose of operationalizing quality attributes that contribute to technical sustainability
from the viewpoint of software engineer (researcher or practitioner)
in the context of the MeGSuS workshop[1].

Our focus group study represents an early assessment exercise of the MEASURE platform. We define the following research question:

RQ$_1$: *In which extent can the MEASURE platform be useful for measuring technical sustainability?*

For determining the potential usefulness of MEASURE for operationalizing the sustainability-quality attributes, from our research question, we set out three specific questions to our participants:

RQ$_{1.1}$: *Do you agree with the contribution of the selected quality attributes as contributors to technical sustainability?*

RQ$_{1.2}$: *In which phase of the software development life cycle, do you think it would be feasible to measure the list of quality attributes?*

[1] http://eseiw2018.wixsite.com/megsus18

RQ$_{1.3}$: *Which metrics from the MEASURE platform can be useful for measuring technical sustainability?*

B. Participants

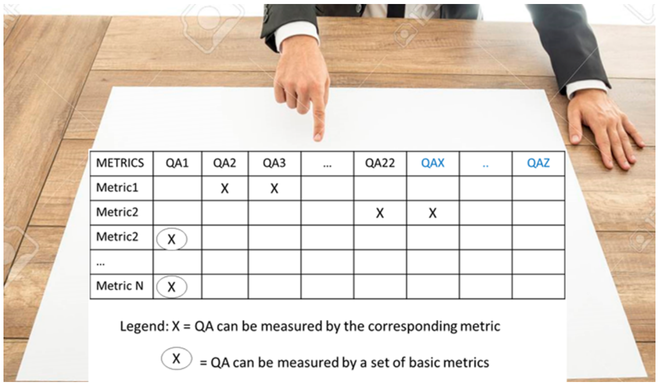
For answering our research question, we considered it advisable that our participants should have a very good knowledge competence on software measurement, as well as interest in any research topic related to software sustainability. Both criteria were successfully satisfied by our eight participants, attendees of the MeGSuS workshop. Two of them were practitioners. All of them contributed to the workshop focusing on software measurement and showed their interest in the topic by that.

C. Instrumentation and data collection

The focus group study was organized in four small groups, to run the study, the following instrumentation was distributed among the groups:

- Technical sustainability definition
- List of quality attributes and corresponding definitions of the attributes
- Metrics from the MEASURE platform, whose definitions were accessible via a wiki website[2]

After reading and clarifying the definitions, the participants selected the phase of the software development, they felt most familiar with. Regarding our two first specifics questions, verbal data was collected, whereas for our third question, a large sheet of paper containing a grid was used by each focus group.



METRICS	QA1	QA2	QA3	...	QA22	QAX	..	QAZ
Metric1		X	X					
Metric2					X	X		
Metric2	ⓧ							
...								
Metric N	ⓧ							

Legend: X = QA can be measured by the corresponding metric

ⓧ = QA can be measured by a set of basic metrics

Fig. 2: Matrix used for mapping selected metrics with the quality attributes (QA)

As shown in Figure 2, participants used an ”X” for representing the relation: *”M can measure QA”* or *”QA can be measured by M”*.
Those ”X” enclosed by a circle were used to identify a set of basic metrics that can measure a QA.

Table I shows the twenty two quality attributes of technical sustainability that were analyzed by our focus group participants.

TABLE I: Technical sustainability-quality attributes identified in [19]

ID	Characteristics	Quality attributes
QA1	Functional suitability	Functional correctness
QA2	Compatibility	Interoperability
QA3	Reliability	Availability
QA4	Functional suitability	Functional appropriateness
QA5	Satisfaction	Usefulness
QA6	Reliability	Fault tolerance
QA7	Maintainability	Modifiability
QA8	Satisfaction	Trust
QA9	Context coverage	Context completeness
QA10	Effectiveness	Effectiveness
QA11	Robustness	Robustness
QA12	Portability	Adaptability
QA13	Performance efficiency	Time behaviour
QA14	Maintainability	Modularity
QA15	Maintainability	Testability
QA16	Reliability	Recoverability
QA17	Compatibility	Coexistence
QA18	Reliability	Maturity
QA19	Efficiency	Efficiency
QA20	Survivability	Survivability
QA21	Performance efficiency	Capacity
QA22	Security	Integrity

D. Procedure

As shown in Figure 3, the procedure of our focus group study involves the following four phases:

1) Preparation phase: This phase has two objectives: i) to get a common understanding on what software sustainability means regarding technical sustainability dimensions, ii) to decide which sustainability dimensions are going to be used in the next phase. This phase has been carried out by the organizers of the focus group, consisting of one moderator and two assistants.

After having a discussion (before realizing the focus group), and considering also the time allocated for this study as part of the MeGSuS workshop, the researchers decided to work with the technical sustainability dimension.

The activities of the next phases were carried out during the focus group.

2) Phase 1: What?: The objective of this phase is to validate the contribution of the corresponding QAs to the technical sustainability dimension. Thus, in this phase, participants answered **RQ$_{1.1}$**. The moderator introduced briefly the motivation of the focus group, presented an overview of the sustainability-quality model as well as a plan of activities to be carried out. The outcome of this phase is a list of selected QAs that will be analyzed in the following phases.

The average time taken for this phase was about 10 minutes.

3) Phase 2: When?: The objective of this second phase is to discuss on which phases of the software life cycle the selected qualities could be measured. Thus in this phase, based on their participants experience, **RQ$_{1.2}$** was answered. The average time taken was about 5 minutes.

4) Phase 3: How?: The objective of this third phase is to assess the usefulness of the metrics from the MEASURE platform. Thus, in this phase, participants answered **RQ$_{1.3}$**.

[2] https://github.com/ITEA3-Measure/Measures/wiki

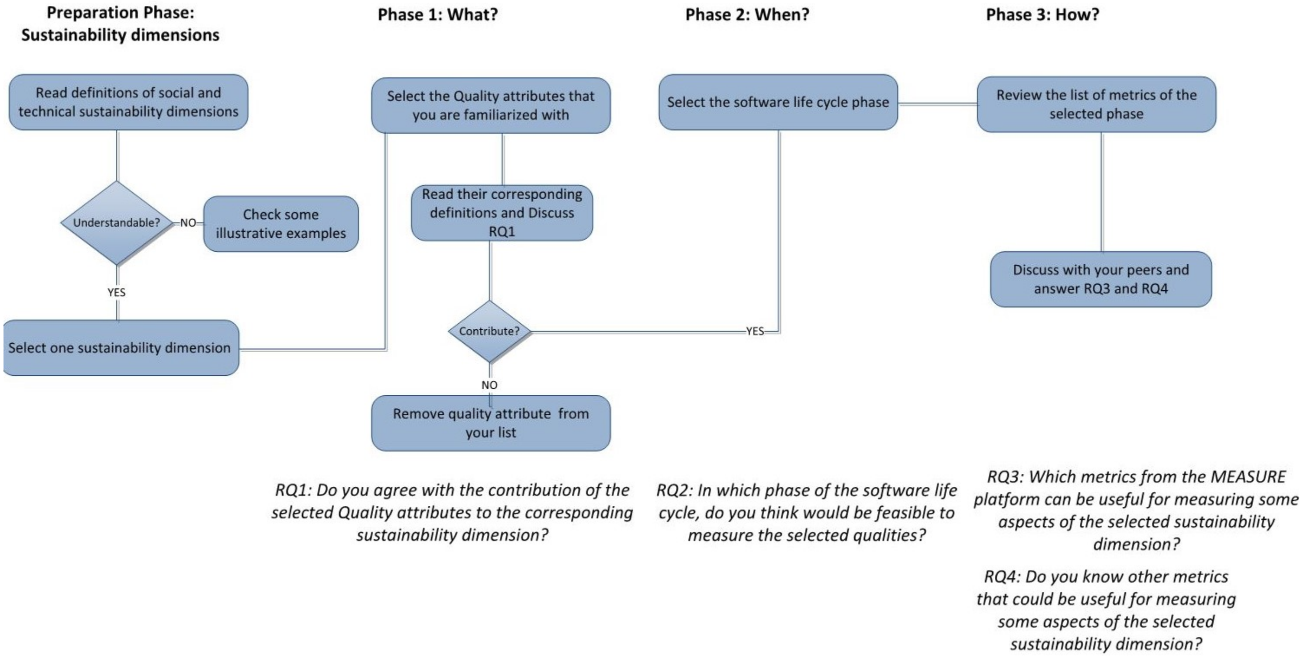

Preparation Phase:
Sustainability dimensions
Phase 1: What?
Phase 2: When?
Phase 3: How?
Read definitions of social and technical sustainability dimensions
Understandable?
NO
Check some illustrative examples
YES
Select one sustainability dimension
Select the Quality attributes that you are familiarized with
Read their corresponding definitions and Discuss RQ1
Contribute?
YES
NO
Remove quality attribute from your list
Select the software life cycle phase
Review the list of metrics of the selected phase
Discuss with your peers and answer RQ3 and RQ4
RQ1: Do you agree with the contribution of the selected Quality attributes to the corresponding sustainability dimension?
RQ2: In which phase of the software life cycle, do you think would be feasible to measure the selected qualities?
RQ3: Which metrics from the MEASURE platform can be useful for measuring some aspects of the selected sustainability dimension?
RQ4: Do you know other metrics that could be useful for measuring some aspects of the selected sustainability dimension?

Fig. 3: Focus group procedure

It took approximately 25 minutes. All participants of the four focus groups shared their mapping results, by emphasizing the reasoning behind the mappings, difficulties of understanding the purpose of some metrics and discussing open questions on the connection of the issues. In this phase, we were open to new metrics that could be suggested by the participants. However, due to time restrictions, this data was not collected.

IV. Threats to validity

We identified the following threats to validity [20] of our study.

- **External validity**. It is the ability to generalize the results from a sample to a population. As focus groups tend to use rather small, homogeneous samples, generalization is the main limitation of our study. Our study involved four mini-groups, with people from different countries, but most of them were researchers. To mitigate this threat, we are going to replicate this first focus group, involving more groups representing a diverse sample of people.
- **Internal validity**. It is strengthened by a moderator providing an appropriate amount of guidance without introducing any of his/her own opinion or stifling free expression. In order to reduce this threat, the moderator used an introductory material (Powerpoint-slides) for contextualizing the focus group study.
- **Construct validity.** It is concerned with whether the focus group is actually measuring what they are trying to measure. In our focus group, we focus on investigate the coverage and measurability aspects. By using two different existing approaches - one with a more practical orientation and one theoretical model - having a common focus, the direction of the focus group was specifically predefined. This ensured that the focus of discussion was also set on the technical sustainability dimension.

V. Results and discussion

In order to answer our main research question related to *the usefulness of the MEASURE platform for measuring the technical sustainability dimension*, we analyzed the collected data from each focus group (see matrix, Figure 6). Usefulness of the platform is analyzed regarding coverage and measurability aspects, which are discussed as follows.

A. Analyzing the coverage of the MEASURE platform

Considering the total of metrics available at the MEASURE platform [21], [22], [13], which are organized by software development phase, Table II shows the percentage of software metrics selected by the participants of each mini-focus group as useful for measuring any QA of the technical sustainability dimension. According to these results, we observe that all metrics available for the specification phase (100%) could be related with the corresponding QAs, whereas only 25% of 51 metrics for the implementation phase were related.

Next we present the selected metrics by each mini-focus group.

1) Metrics for the specification phase: Table III : ”Selected metrics of the specification phase” shows the 10 selected specification phase metrics that were mapped with the QAs of the technical sustainability dimension.

TABLE II: Percentage of metrics selected by the focus group participants per development phase

Phase	**Number of selected metrics**	**Total of metrics**[3]	**Percentage**
Specification	10	10	100%
Design	16	35	46%
Implementation	13	51	25%
Testing	15	22	68%

2) Metrics for the design phase: Table IV: ”Selected metrics of the design phase” shows the 18 selected design phase metrics that were mapped with the QAs of the technical sustainability dimension.

3) Metrics for the implementation phase: Table V : ”Selected metrics of the implementation phase” shows the 13 selected implementation phase metrics that were mapped with the QAs of the technical sustainability dimension.

4) Metrics for the testing phase: Table VI: ”Selected metrics of the testing phase” shows the 21 selected testing phase metrics that were mapped with the QAs of the technical sustainability dimension.

As shown in the matrix (Figure 2 in Appendix), the specification, design, implementation, and testing metrics where associated to the quality attributes presented in Table I.

B. Analyzing the measurability of the quality attributes

Considering the twenty-two QAs of the technical sustainability dimension (see Table I), Figure 4 shows the percentages of QAs that can be measured by at least one of the selected metrics. Most of the QAs can be measured at the specification phase (82%, 18 of 22 QAs), followed by design (59%, 13 of 22 QAs), testing (32%, 7 of 22 QAs) and implementation (23%, 5 of 22 QAs).

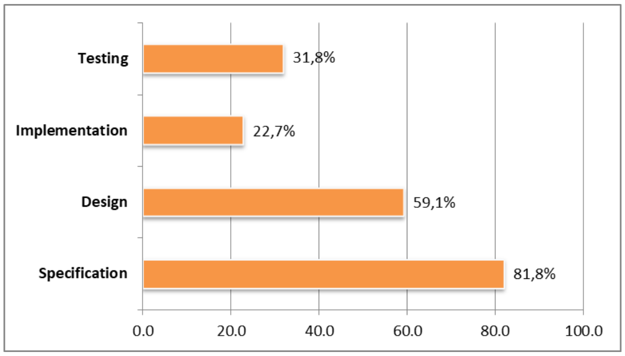


Fig. 4: Percentage of measurable quality attributes per phase

This indicates that most of the QAs related to the technical sustainability can be qualified as measurable. In order to represent the extent of measurability for each development phase, we calculated the number of available metrics selected from the platform for measuring each QA (see Figure 5). According to these results, we observe that our participants found that functional correctness, robustness and maturity can be measured by using a good number of metrics at the testing phase (13 metrics). In case of the specification phase, especially functional correctness and functional appropriateness are of high importance, meaning covered by many metrics (5 of 10 metrics). Efficiency is connected with most of the metrics of the design phase while the quality attribute is not covered in the other phases analyzed. Overall, functional suitability is connected to many of the proposed metrics for the analyzed software development phases.

VI. Conclusions

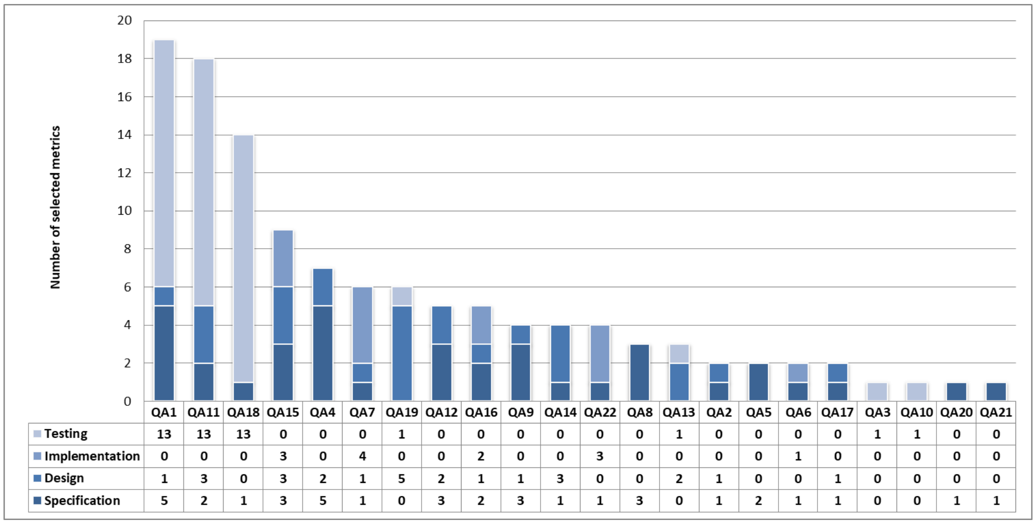
In this paper, we describe the result of the the focus group designed to discuss on ”What, when and how to measure software sustainability”. The authors organized the study within the MeGSuS 2018: 4th International Workshop on Measurement and Metrics for Green and Sustainable Software Systems [23].

Through the focus group, we found a good number of metrics that were selected from the MEASURE platform as ”potentially” useful for measuring quality attributes of the technical sustainability dimension along certain phases of the software development life cycle (i.e. design, specification, testing, implementation). This result provides evidence on the coverability of the MEASURE platform for the specification, design, implementation and testing phases.

Moreover, the study has also shown that most of the technical sustainability-quality attributes are measurable. The results can be appreciated and are summarized in Figure 5, where we can highlight the following results: .

- Metrics for the specification phase were distributed among the various QAs with higher metrics related to QA1 and QA4.
- Metrics for the design phase were distributed among the various QAs with higher metrics related to QA19, QA15 and QA14.
- Metrics for the implementation phase focus, according to the results of the focus group, on a limited number of QAs (QA15, QA7, QA22) related to technical sustainability dimension. The subgroup considered that maintenability / testability, maintenability / modifiability, and security were the QAs associated to the higher number of implementation metrics (see Table V) .
- Metrics for the testing phase focus on a limited number of QAs (QA1, QA11, QA18) related to technical sustainability dimension. The sub-group considered that functional suitability, robustness, and reliability were the QAs associated to the higher number of testing metrics (see Table VI).

Functional correctness, robustness, maturity and testability are the most measurable quality attributes considering the four phases. The focus group acknowledged that the technical sustainability dimension [19] could be operationalized by the MEASURE platform implemented metrics. For validating these results, we are going to replicate the study to find both similarities and differences.



	QA1	QA11	QA18	QA15	QA4	QA7	QA19	QA12	QA16	QA9	QA14	QA22	QA8	QA13	QA2	QA5	QA6	QA17	QA3	QA10	QA20	QA21
Testing	13	13	13	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0
Implementation	0	0	0	3	0	4	0	0	2	0	0	3	0	0	0	0	1	0	0	0	0	0
Design	1	3	0	3	2	1	5	2	1	1	3	0	0	2	1	0	0	1	0	0	0	0
Specification	5	2	1	3	5	1	0	3	2	3	1	1	3	0	1	2	1	1	0	0	1	1

Fig. 5: Measurability: Number of metrics per quality attribute related to technical sustainability

Another future work of the team include providing MEASURE visualization dashboards to support users in the evaluation of the technical sustainability of a given software artifact.

Acknowledgment

We thank all the participants who took part in our focus group study: Jérôme Rocheteau from the Institut catholique d'arts et métiers (Icam) site of Nantes, Birgit Penzenstadler from California State University Long Beach, Shola Oyedeji from Lappeenranta University of Technology, Denisse Muñante from University of Bordeaux, Diogo Silveira Mendoa from Pontifical Catholic University of Rio de Janeiro, and Thibault Beziers la Fosse from Laboratoire des Sciences du Numrique de Nantes, Software Modeling Group (LS2N-NAOMOD).

The research presented in this paper is partially funded by the ITEA3 Project no. 14009 called MEASURE (1st December 2015 and running till 31st August 2019).

References

[1] P. Lago, "Software and sustainability [inaugural lecture]," http://dare.ubvu.vu.nl, Jan. 2016.

[2] B. Penzenstadler and H. Femmer, "A generic model for sustainability with process-and product-specific instances," in *Proceedings of the 2013 workshop on Green in/by software engineering*. ACM, 2013, pp. 3–8.

[3] P. Lago, S. A. Koçak, I. Crnkovic, and B. Penzenstadler, "Framing sustainability as a property of software quality," *Communications of the ACM*, vol. 58, no. 10, pp. 70–78, 2015.

[4] S. Selyamani and N. Ahmad, "Green computing: the overview of awareness, practices and responsibility among students in higher education institutes," *J. Inf. Syst. Res. Innov*, 2015.

[5] R. Chitchyan, C. Becker, S. Betz, L. Duboc, B. Penzenstadler, N. Seyff, and C. C. Venters, "Sustainability design in requirements engineering: state of practice," in *Proceedings of the 38th International Conference on Software Engineering Companion*. ACM, 2016, pp. 533–542.

[6] I. Manotas, C. Bird, R. Zhang, D. Shepherd, C. Jaspan, C. Sadowski, L. Pollock, and J. Clause, "An empirical study of practitioners' perspectives on green software engineering," in *Software Engineering (ICSE), 2016 IEEE/ACM 38th International Conference on*. IEEE, 2016, pp. 237–248.

[7] C. Pang, A. Hindle, B. Adams, and A. E. Hassan, "What do programmers know about software energy consumption?" *IEEE Software*, vol. 33, no. 3, pp. 83–89, 2016.

[8] C. Calero and M. Piattini, "Introduction to green in software engineering," in *Green in Software Engineering*. Springer, 2015, pp. 3–27.

[9] L. M. Hilty and B. Aebischer, "Ict for sustainability: An emerging research field," in *ICT Innovations for Sustainability*. Springer, 2015, pp. 3–36.

[10] E. Kern, L. M. Hilty, A. Guldner, Y. V. Maksimov, A. Filler, J. Gröger, and S. Naumann, "Sustainable software productstowards assessment criteria for resource and energy efficiency," *Future Generation Computer Systems*, vol. 86, pp. 199–210, 2018.

[11] O. Condori Fernandez and P. Lago, *A Sustainability-quality Model: (version 1.0)*. VU Technical Report, 11 2018. [Online]. Available: https://research.vu.nl/en/publications/a-sustainability-quality-model-version-10

[12] Softeam R&D, "MEASURE project website," http://measure.softeam-rd.eu/, Oct. 2017, last accessed on 2018-11-01.

[13] A. Abherve, A. Bagnato, A. Stefanescu, and A. Baars, "Github project for the MEASURE platform," https://github.com/ITEA3-Measure/MeasurePlatform/graphs/contributors, Sep. 2017, last accessed on 2018-11-01.

[14] A. Abherve, "Github project for the SMM Measure API library," https://github.com/ITEA3-Measure/SMMMeasureApi, Aug. 2017, last accessed on 2018-11-01.

[15] Object Management Group, "The Software Metrics Meta-Model Specification 1.1.1," http://www.omg.org/spec/SMM/1.1.1/, Apr. 2016, last accessed on 2018-11-01.

[16] C. Venters, L. Lau, M. Griffiths, V. Holmes, R. Ward, C. Jay, C. Dibsdale, and J. Xu, "The blind men and the elephant: Towards an empirical evaluation framework for software sustainability," *Journal of Open Research Software*, vol. 2, no. 1, 2014.

[17] C. Calero, M. Á. Moraga, and M. F. Bertoa, "Towards a software product sustainability model," *CoRR*, vol. abs/1309.1640, 2013. [Online]. Available: http://arxiv.org/abs/1309.1640

[18] A. Raturi, B. Penzenstadler, B. Tomlinson, and D. Richardson, "Developing a sustainability non-functional requirements framework," in *Proceedings of the 3rd International Workshop on Green and Sustainable Software*, ser. GREENS 2014. New York, NY, USA: ACM, 2014, pp. 1–8. [Online]. Available: http://doi.acm.org/10.1145/2593743.2593744

[19] N. Condori-Fernandez and P. Lago, "Characterizing the contribution of quality requirements to software sustainability," *Journal of Systems and Software*, vol. 137, pp. 289 – 305, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121217302984

[20] R. A. Krueger, *Focus groups : a practical guide for applied research / Richard A. Krueger ; foreword by Michael Quinn Patton.*

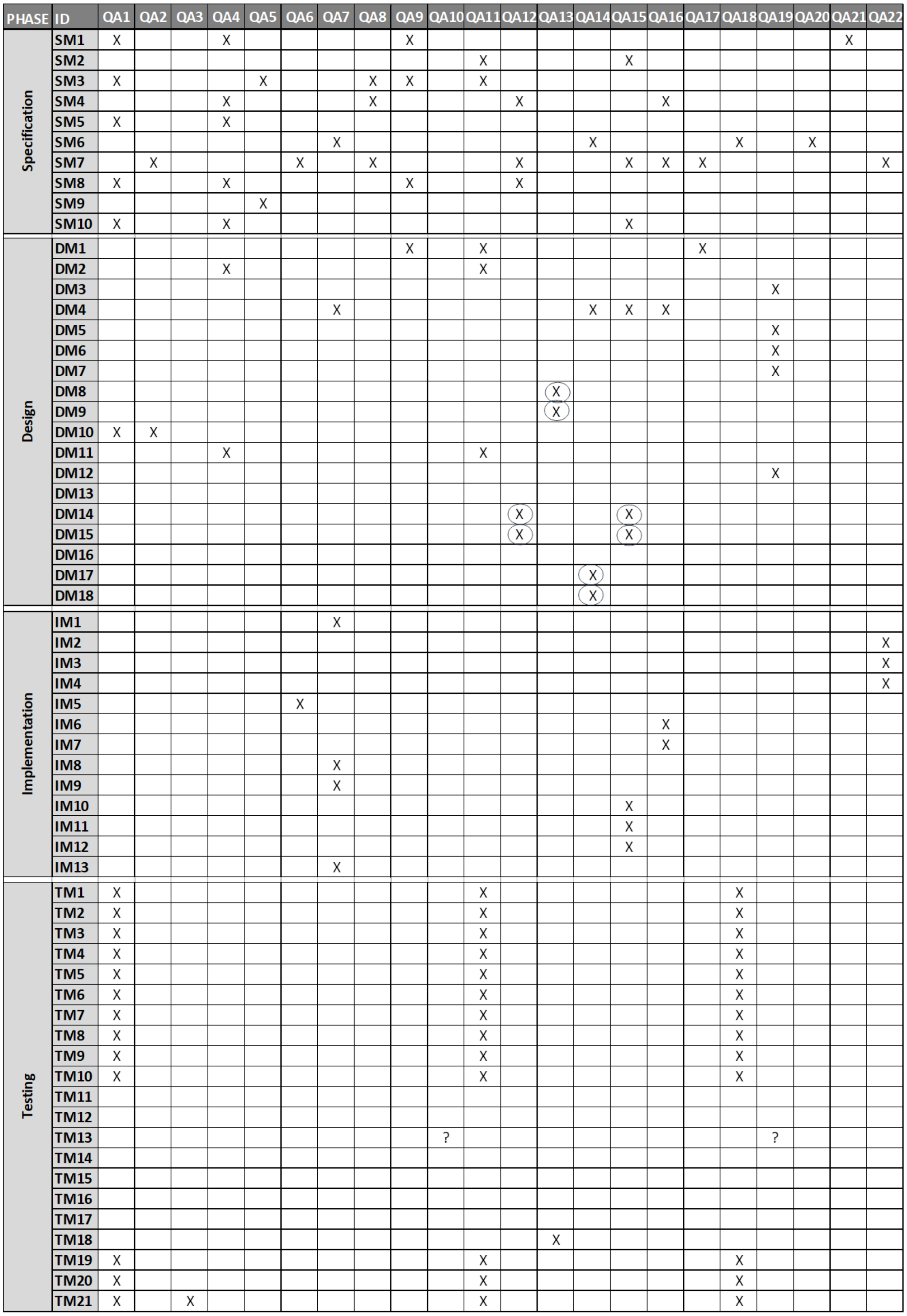
Sage Publications Newbury Park, Calif, 1988. [Online]. Available: http://www.loc.gov/catdir/enhancements/fy0654/87033413-t.html

[21] A. Abherve and A. Bagnato, “Repository of measures specification in smm,” https://github.com/ITEA3-Measure/Measures/wiki, Aug. 2017, last accessed on 2018-12-01.

[22] A. Bagnato and A. Abherve, “Repository of measure implementations,” https://github.com/ITEA3-Measure/Measures, Aug. 2017, last accessed on 2018-12-01.

[23] E. K. Alessandra Bagnato, Nelly Condori Fernandez, “4th workshop on measurement and metrics for green and sustainable software systems (megsus18) october 9, 2018 - oulu, finland,” http://eseiw2018.wixsite.com/megsus18, Aug. 2018, last accessed on 2018-12-01.

APPENDIX



PHASE	ID	QA1	QA2	QA3	QA4	QA5	QA6	QA7	QA8	QA9	QA10	QA11	QA12	QA13	QA14	QA15	QA16	QA17	QA18	QA19	QA20	QA21	QA22
Specification	SM1	X			X					X												X	
	SM2											X				X							
	SM3	X				X			X	X		X											
	SM4				X				X				X				X						
	SM5	X			X																		
	SM6							X							X				X		X		
	SM7		X				X		X				X			X	X	X					X
	SM8	X			X					X			X										
	SM9					X																	
	SM10	X			X											X							
Design	DM1									X		X						X					
	DM2				X							X											
	DM3																			X			
	DM4							X							X	X	X						
	DM5																			X			
	DM6																			X			
	DM7																			X			
	DM8													Ⓧ									
	DM9													Ⓧ									
	DM10	X	X																				
	DM11				X							X											
	DM12																			X			
	DM13																						
	DM14												Ⓧ			Ⓧ							
	DM15												Ⓧ			Ⓧ							
	DM16																						
	DM17														Ⓧ								
	DM18														Ⓧ								
Implementation	IM1							X															
	IM2																						X
	IM3																						X
	IM4																						X
	IM5						X																
	IM6																X						
	IM7																X						
	IM8							X															
	IM9							X															
	IM10															X							
	IM11															X							
	IM12															X							
	IM13							X															
Testing	TM1	X										X							X				
	TM2	X										X							X				
	TM3	X										X							X				
	TM4	X										X							X				
	TM5	X										X							X				
	TM6	X										X							X				
	TM7	X										X							X				
	TM8	X										X							X				
	TM9	X										X							X				
	TM10	X										X							X				
	TM11																						
	TM12																						
	TM13										?									?			
	TM14																						
	TM15																						
	TM16																						
	TM17																						
	TM18													X									
	TM19	X										X							X				
	TM20	X										X							X				
	TM21	X		X								X							X				

Fig. 6: Mapping between quality attributes related to technical sustainability dimension and metrics from the MEASURE platform. (X = ”Metric can measure quality attribute” or rather ”Quality attribute can be measured by metric”, ? = ”connection needs to be discussed”)

TABLE III: Selected metrics of the specification phase

ID	Short name	Description
SM1	Number of Requirement	Total number of requirement defined in the selected scope.
SM2	Number of Tests Requirements	Total number of tests defined in the selected scope.
SM3	Satisfaction Quality Indice	Percentage of requirements that have been satisfied.
SM4	Requirement Traceability To Implementation Indice	Percentage of requirements that have been satisfied.
SM5	Requirement Coverage Indice	The average number of requirements tracing an architecture model.
SM6	Requirement Complexity Indice	The average number of sub requirements defined to rafine an existing requirement.
SM7	Number Of Risks	Total number of risks defined in the selected scope. .
SM8	Number Of Business Rules	Total number of requirement defined in the selected scope.
SM9	Number Of Goals	Total number of goals defined in the selected scope.
SM10	Requirement Traceability To Test Indice	The % of requirement of tracing a test model.

TABLE IV: Selected metrics of the design phase

ID	Short name	Description
DM1	Class Complexity Index	The number of direct subclasses of a class. A class implementing an interface counts as a direct child of that interface.
DM2	Package Dependecies Ratio	The average number of dependencies from a package.
DM3	Number of Methods	Total number of methods defined in the selected scope.
DM4	Software Component decomposition	The number of software compoments identified in an application architecture.
DM5	Number of Classes	Total number of classes in the selected scope
DM6	Number of Interfaces	Total number of interfaces in the selected scope.
DM7	Number of Methods	Total number of methods defined in the selected scope.
DM8	Number of Components	Total number of Components defined in the selected scope.
DM9	Number of Packages	Total number of Packages defined in the selected scope.
DM10	Class Dependency Ratio	The average number of dependencies from a class.
DM11	Package Dependency Ratio	The average number of dependencies from a package.
DM12	Model Abstractness Index	The% of abstract classes (and interfaces) divided by the total number of types in a package.
DM13	Number of Fields	Total number of fields defined in the selected scope.
DM14	Number of Use Cases	Total number of use cases defined in the selected scope.
DM15	Number of Actors	Total number of actors defined in the selected scope.
DM16	Number of Component Types	Total number of interfaces component types in the java Modelio model. Along with the total number of data, this metric provides an idea of the functional richness of the modelled application.
DM17	Number of Aggregated Components	Total number of aggregated components in the java Modelio model. Along with the number of composed components, this metric reflects the usage of the software decomposition in the modelled application.
DM18	Number of Composed Components	Count the number of Interface annotated @ComposedComponent in Java Model.

TABLE V: Selected metrics of the Implementation phase

ID	Short Name	Description
IM1	Cognitive Complexity	defining how hard it is to understand the code's control flow.

IM2	Security Rating	defining as A = 0 vulnerability,
		B = at least 1 minor vulnerability,
		C = at least 1 major vulnerability,
		D = at least 1 critical vulnerability,
		E = at least 1 blocker vulnerability

IM3	Security remediation effort on new code	Effort to fix all vulnerability issues found on the code changed in leak periods
IM4	Security remediation effort	Effort to fix all vulnerability issues.

IM5	Reliability rating	A = 0 bug, B = at least 1 minor bug,
		C = at least 1 major bug,
		D = at least 1 critical bug,
		E = at least 1 blocker bug

IM6	Reliability remediation effort	Effort to fix all bug issues.
IM7	Reliability remediation effort on new code	Effort to fix all bug issues found on the code changed in the leak period
IM8	File complexity	Average complexity by file.
IM9	Code Smells	Number of code smells.
IM10	New issues by severity	Number of new issues with severity (blocker,critical, major, minor)
IM11	New Issues	Number of new issues
IM12	Maintainability rating	Rating given to your project related to the value of your Technical Debt Ratio.
IM13	Technical debt	Effort to fix all maintainability issues.

TABLE VI: Selected metrics of the testing phase

ID	**Metric**	**Description**
TM1	Condition Coverage	On each line of code containing some boolean expressions, the condition coverage simply answers the following question: ’Has each boolean expression been evaluated both to true and false?’. This is the density of possible conditions in flow control structures that have been followed during unit tests execution.
TM2	Condition Coverage On New Code	On each line of code containing some boolean expressions, the condition coverage simply answers the following question: ’Has each boolean expression been evaluated both to true and false?’. This is the density of possible conditions in flow control structures that have been followed during unit tests execution.
TM3	Condition Coverage Hits	List of covered conditions.
TM4	Conditions By Line	Number of conditions by line.
TM5	Covered Conditions By Line	Number of covered conditions by line.
TM6	Coverage	It is a mix of Line coverage and Condition coverage. Its goal is to provide an even more accurate answer to the following question: How much of the source code has been covered by the unit tests?
TM7	Coverage On New Code	It is a mix of Line coverage and Condition coverage. Its goal is to provide an even more accurate answer to the following question: How much of the source code has been covered by the unit tests? Restricted to new / updated source code.
TM8	Line Coverage	On a given line of code, Line coverage simply answers the following question: Has this line of code been executed during the execution of the unit tests?. It is the density of covered lines by unit tests:
TM9	Line Coverage On New Code	On a given line of code, Line coverage simply answers the following question: Has this line of code been executed during the execution of the unit tests?. It is the density of covered lines by unit tests Restricted to new / updated source code.
TM10	Line Coverage Hits	List of covered lines.
TM11	Lines To Cover	Number of lines of code which could be covered by unit tests
TM12	Lines To Cover On NEw Code	Number of lines of code which could be covered by unit tests Restricted to new / updated source code.
TM13	Skipped Unit Tests	Number of skipped unit tests.
TM14	Uncovered Conditions	Number of conditions which are not covered by unit tests.
TM15	Uncovered Conditions On New Code	Number of conditions which are not covered by unit tests. Restricted to new / updated source code.
TM16	Uncovered Lines On New Code	Number of conditions which are not covered by unit tests. Restricted to new / updated source code.
TM17	Unit Tests	Number of unit tests.
TM18	Unit Tests Duration	Time required to execute all the unit tests.
TM19	Unit Test Errors	Number of unit tests that have failed.
TM20	Unit Test Failures	Number of unit tests that have failed with an unexpected exception.
TM21	Unit Test Success Density Percent	Test success density = (Unit tests - (Unit test errors + Unit test failures)) / Unit tests * 100