The Journal of Systems and Software 137 (2018) 289–305

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Characterizing the contribution of quality requirements to software sustainability

Nelly Condori-Fernandez[a,b,*], Patricia Lago[b]

[a] Universidade da Coruna, Spain
[b] Vrije Universiteit Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 27 June 2016
Revised 23 November 2017
Accepted 8 December 2017
Available online 9 December 2017

Keywords:
Sustainability
Software-intensive systems
Quality requirements
Survey

ABSTRACT

Background: Since sustainability became a challenge in software engineering, researchers mainly from requirements engineering and software architecture communities have contributed to defining the basis of the notion of sustainability-aware software.
Problem: Despite these valuable efforts, the assessment and design based on the notion of sustainability as a software quality is still poorly understood. There is no consensus on which sustainability requirements should be considered.
Aim and Method: To fill this gap, a survey was designed with a double objective: i) determine to which extent quality requirements contribute to the sustainability of software-intensive systems; and ii) identify direct dependencies among the sustainability dimensions. The survey involved different target audiences (e.g. software architects, ICT practitioners with expertise in Sustainability). We evaluated the perceived importance/relevance of each sustainability dimension, and the perceived usefulness of exploiting a sustainability model in different software engineering activities.
Results: Most respondents considered modifiability as relevant for addressing both technical and environmental sustainability. Functional correctness, availability, modifiability, interoperability and recoverability favor positively the endurability of software systems. This study has also identified security, satisfaction, and freedom from risk as very good contributors to social sustainability. Satisfaction was also considered by the respondents as a good contributor to economic sustainability.

1. Introduction

Society increasingly depends on software-intensive systems[1] that are becoming ever more distributed, heterogeneous, decentralized and inter-dependent, while operating in dynamic and often unpredictable environments Gerostathopoulos et al. (2014), Hölzl et al. (2008). As software systems are expected to become even more ubiquitous in the coming years, software sustainability is gaining attention as one of the key and urgent challenges of the 21st Century.

In recent years, there has been an increasing interest in understanding the importance of i) designing sustainability-aware software systems (e.g. Betz et al. (2015); Becker et al. (2015)); and ii) assessing sustainable software products (e.g. Koziolek (2011); Venters et al. (2014); Albertao et al. (2010)).

Koziolek (2011) argues that software architectures are a major driver for the sustainability (in terms of longevity and evolvability) of software-intensive systems, because they influence how quickly and correctly a developer is able to understand, analyze, extend, test, and maintain a software system. On the other hand, in software architecture, possible design solutions are evaluated against relevant quality requirements. The choice of design alternatives heavily impact the quality of software systems. Hence, characterizing the contribution of quality requirements to software sustainability should be a major concern for software architects. Relying on a sound definition of sustainability quality requirements is also the first step to extend the established software architecture practices for sustainability - these practices including design architecture viewpoints (for sustainability modeling), architecture analysis of trade-offs (for sustainability design decision making), quality assessment and measurement (for sustainability monitoring), and architectural tactics (for reusing design decisions addressing sustainability). In short, we argue that the key challenge for software sustainability is its characterization as a software quality requirement.

The most frequently used definition of sustainability refers to dimensions of economic, social, technical and environmental sus-

* Corresponding author.
E-mail addresses: n.condori.fernandez@udc.es, n.condori-fernandez@vu.nl, ncfyllen7@gmail.com (N. Condori-Fernandez).

[1] Systems in which software interacts with other software, systems, devices, sensors and people Hölzl et al. (2008).

https://doi.org/10.1016/j.jss.2017.12.005
0164-1212/

tainability Lago et al. (2015), Razavian et al. (2014). According to Lago et al. (2015), a sustainability model should include both traditional (system-related) quality requirements and sustainability-related ones. This does not mean that all sustainability dimensions must be always addressed together to guarantee sustainability. In fact, the relevance of the different dimensions depends on the type of software system. For instance, a software system designed to remind patients whether or not a medicine has been taken typically requires to address requirements of the social sustainability (like usability, safety, persuasion and health risk mitigation) and of the technical sustainability (like adaptability and availability, which are important to ensure user satisfaction and hence incentivize patients to use the application). Of course requirements of the environmental dimension could be met as long as required.

In order to investigate which relevant quality requirements are related to sustainability concerns of software-intensive systems, we designed and conducted a survey involving different target audiences such as requirements engineers, software architects, ICT practitioners and researchers with expertise in Sustainability, and project managers.

Our main contribution consists of identifying the relevant quality requirements that contribute to the economic, technical, environmental and social sustainability dimensions of software-intensive systems. Moreover, direct dependencies between the four sustainability dimensions were identified from the survey results. We also evaluate the perceived usefulness of our software sustainability model as well as the importance level of each sustainability dimension from different perspectives.

This paper is organized as follows. In the next section, we describe our research methodology. Section 3 presents and discusses the survey results. Section 4 discusses possible threats to validity, while Section 5 presents the related works. Finally, in Section 6 we provide our conclusions.

2. Research methodology

The study takes the form of a survey, which is a research methodology suitable for gathering self-reported quantitative and qualitative data Pfleeger and Kitchenham (2001). It uses an on-line questionnaire for data collection. In the following we describe the goal and research questions, the survey design, and the survey conduction.

2.1. Goal and research questions

The goal of the survey is to determine the extent to which quality requirements contribute to the sustainability of software-intensive systems with respect to the technical, economic, social and environmental dimensions of software-intensive systems.

The following research questions are addressed:

- RQ1: How do quality requirements contribute to the sustainability dimensions of software-intensive systems? and in particular,
  RQ1.1.: What quality requirements can contribute to which sustainability dimension of software-intensive systems with a high consensus?
  RQ1.2.: What types of dependencies among sustainability dimensions can be identified?
- RQ2: How do practitioners and researchers perceive the importance/relevance of each sustainability dimension?
- RQ3: How useful is the Software Sustainability model perceived for supporting the design, assessment and requirements prioritization of software-intensive systems?

Table 1
Interpretation scale for Kappa.

Kappa value	Interpretations
<0	Poor
0.00–0.20	Slight
0.21–0.40	Fair
0.41–0.60	Moderate
0.61–0.80	Substantial
0.81–1.00	Almost perfect

Table 2
Software sustainability Survey questions before the pilot conduction.

Parts	Questions type	Scale
Demographic part	4 closed questions	Nominal
Sustainability dimensions rank	1 closed question	Ratio
Technical sustainability	12 closed questions	Nominal
Economic sustainability	12 closed questions	Nominal
Social sustainability	8 closed questions	Nominal
Environmental sustainability	9 closed questions	Nominal

2.2. Survey design

Given software sustainability is considered as a relatively new concept including both traditional quality requirements and unconventional sustainability-related requirements, as shown in Fig. 1, the survey design was carried out in two stages, between July and October 2015, further described in the following.

2.2.1. Identification of relevant quality requirements from ISO/IEC 25010

With the purpose of identifying those traditional quality requirements that can contribute to software sustainability, we started with a checklist containing the software quality characteristics defined by the ISO/IEC 25010 standard quality models ISO/IEC (2010). In particular, we focused on the product quality model (see Fig. 2) and quality in use model, which is the users view of the quality of a system containing software, and is measured in terms of the result of using the software in specific contexts of use. The quality in use model consists of five characteristics: effectiveness, efficiency, satisfaction, freedom from risk and context coverage.

The checklist consists of 40 items that correspond to the software qualities of the ISO/IEC 25010 standard (product quality model and quality in use model). Definitions of each sustainability dimension and respective software qualities were also given. In this first preliminary stage, five senior researchers of the Software and Services (S2) Research Group at the Vrije Universitiet Amsterdam completed the checklist. Based on experience, the participants were asked to identify which qualities can contribute to which sustainability dimension/s.

In order to measure the agreement of the respondents, we calculated the Kappa statistic for each item of the checklist. All quality attributes that were rated at least as "fair" were considered as items of the sustainability survey (input of the second stage). The interpretation scale for Kappa is shown in Table 1 (see Table A.11 in the Appendix for the full data set).

2.2.2. Building a web-based survey

A web-based survey was built using the Surveygizmo tool. As shown in Table 2, the survey questions have been organized in 6 parts.The first one aims to characterize the participants of the survey by means of 4 closed demographic questions. The second part aims to get an understanding about how respondents perceived

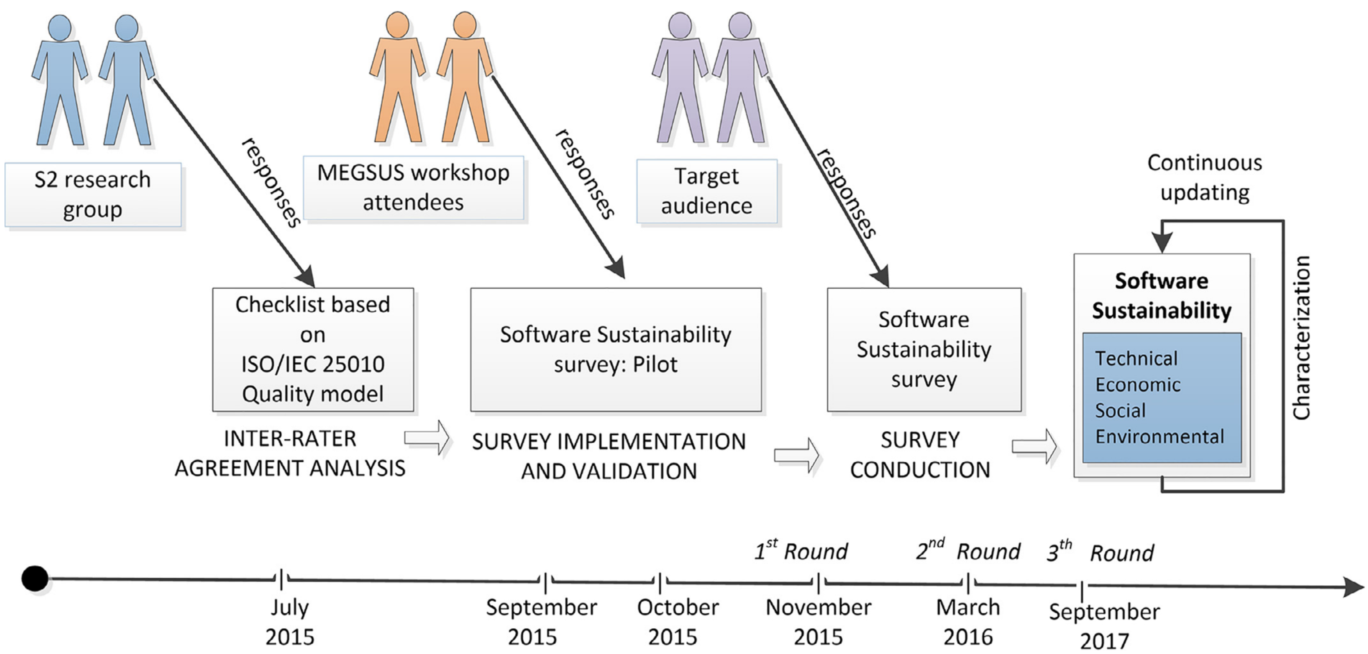

Fig. 1. Software Sustainability Survey design and conduction process.

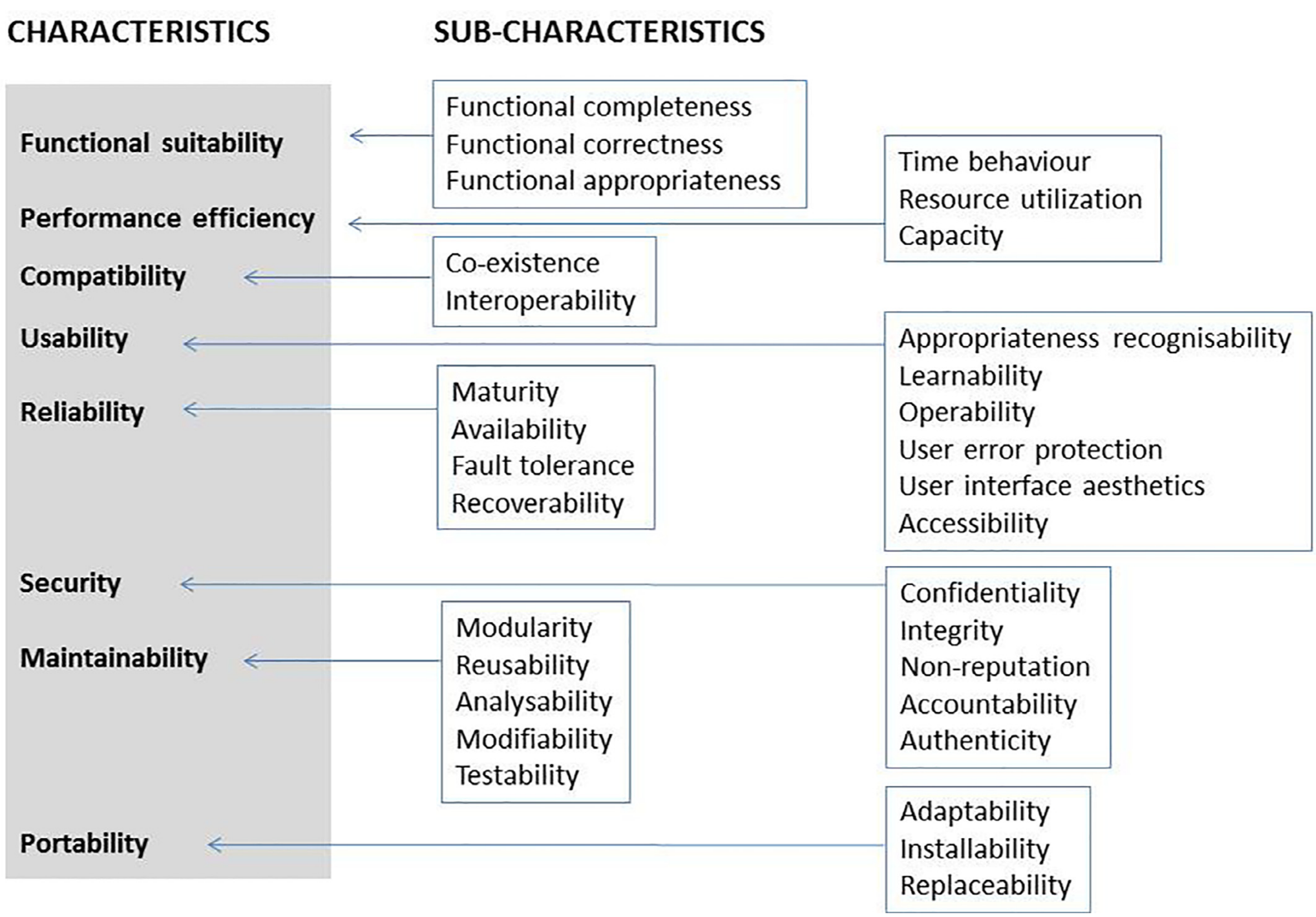

Fig. 2. Product quality model in ISO/IEC 25010.

Table 3
Target audience by sustainability dimension.

Dimension	Target audience
Technical	Software architects, Requirements engineers
Economic	Software architects, Project managers
Social	Requirements engineers, Researchers in ICT & Sustainability ICT practitioners supporting sustainable development
Environmental	ICT practitioners supporting sustainable development Researchers in ICT & Sustainability

Table 4
Source of population for each sustainability dimension.

Dimension	Source of population
Technical	ISO/IEC/IEEE 42010 Users Group, LinkedIn members, and REFSQ conference attendees
Economic	ISO/IEC/IEEE 42010 Users Group, LinkedIn members
Social	LinkedIn members, REFSQ conference attendees EnviroInfo conference attendees
Environmental	LinkedIn members, EnviroInfo conference attendees

the importance/relevance of each sustainability dimension. A definition of sustainability in terms of the technical, social, economic and environmental dimensions was given to the participants.

The next four parts aim to determine the contribution level of each software quality to the respective sustainability dimension. For each part of the survey, we included at the top of the survey page a definition of the respective dimension as well as the definitions of each quality requirement. Moreover, an additional open question was formulated for allowing respondents to add any missing quality requirement they deserved as relevant for the respective sustainability dimension.

2.2.3. Survey validation

In the second stage of the survey design (ref. Fig. 1), eight attendees of the MEGSUS workshop[2] volunteered to respond the survey. The background of the participants was diverse (5 researchers and 3 practitioners, of which 2 were project managers and 1 a software architect). We carried out a survey pilot to ensure the survey questions are comprehensible and valid with respect to the study constructs (sustainability dimensions). This pilot led to the following improvements: i) changing the rating scales from nominal (Yes/No/Undecided) to 4-points ordinal scale (Null, Low, Medium, High); ii) adding new qualities to some of the dimensions (i.e. robustness and survivability to technical dimension); iii) adding a question that allowed us to know about how respondents perceived the usefulness of the software sustainability model, by using a 5-point Likert scale (strongly agree, agree, neutral, disagree, strongly disagree).

We also split the survey in two dimensions and adjusted the survey instructions. This allowed us to reduce the time required to complete the survey (25 min instead of 60 min) and mitigate threats to data quality due to fatigue effects. The final version of the web-based questionnaire is accessible online[3].

2.3. Survey conduction

As shown in Fig. 1, the characterization of each software sustainability dimension (in terms of QRs) is the main outcome of the survey conduction. Each dimension should be continuously updated if new sustainability requirements were identified during the survey replications. Next we present the participant selection and data collection activities that were carried out by following the survey guidelines proposed by de Mello and Travassos (2016).

2.3.1. Participant selection

Being sustainability defined along four different dimensions, we identified the target audience accordingly, based on who can contribute the best professional knowledge needed for characterizing software sustainability. Table 3 shows the sustainability dimension that was assigned to the respective target audience. This way, each participant focused her/his answers on the two dimensions s/he was most knowledgeable about. For instance, we consider requirements engineers that play an important role for addressing social aspects (nearer to user requirements) and that naturally affect directly or indirectly the technical aspects of the software development Jirotka and Goguen (1994), Nuseibeh and Easterbrook (2000).

Based on the target audience, our study subjects (individuals) are researchers or practitioners with more than 1 year working experience in software architecture, requirements engineering, project management, or sustainability in ICT. The sources used to identify an accessible and representative population to support our survey are shown in Table 4.

2.3.2. Data collection

Considering the characteristics of our population, the data collection was carried out in two rounds by distributing the survey to specialized groups with subject-matter experts, as described in the following.

First round:

1. Social-Environmental sustainability survey. On November 18, 2015, we sent a single e-mail to 37 LinkedIn members with expertise on sustainable development and current job within an ICT company located in The Netherlands. The recruitment was carried out by using the LinkedIn Premium account.
2. Technical-Economic sustainability survey. On November 27, 2015, we sent a single generic message to the mailing list of the ISO/IEC/IEEE 42010 Users Group[4]. The invitation should have reached about 150 software architecture experts (both researchers and practitioners).

For both surveys in the first round, a reminder email was sent two weeks after the first invitation. We closed the survey on January 29, 2016.

Second round:

3. Social-Technical sustainability survey. On March 15, 2016 we conducted a live study within the research methods track of the Working conference on Requirements Engineering: Foundations for Software Quality (REFSQ). Researchers and practitioners with background in Requirements engineering and Software Architecture completed this survey.

Third round:

4. Social-environmental sustainability survey. On 14th September, 2017, we sent a single e-mail to 12 attendees of the special session on Energy Aware Software-Engineering and Development collocated at the EnviroInfo conference.
5. Technical-Economic sustainability survey. Starting on 16th September, 2017, we sent a single e-mail to 45 invitees of the LinkedIn group 'Software Project Risk Management'. As we used a LinkedIn free account (allowing sending a max of 15 single e-mails per day), this invitation was spread out over three days (16th–18th September).

Both surveys were closed on 27th September.

[2] http://www.iwsm-mensura.org/2015/megsus.
[3] http://www.s2group.cs.vu.nl/wp-content/uploads/2017/02/surveyS.pdf.
[4] http://www.iso-architecture.org/ieee-1471/.

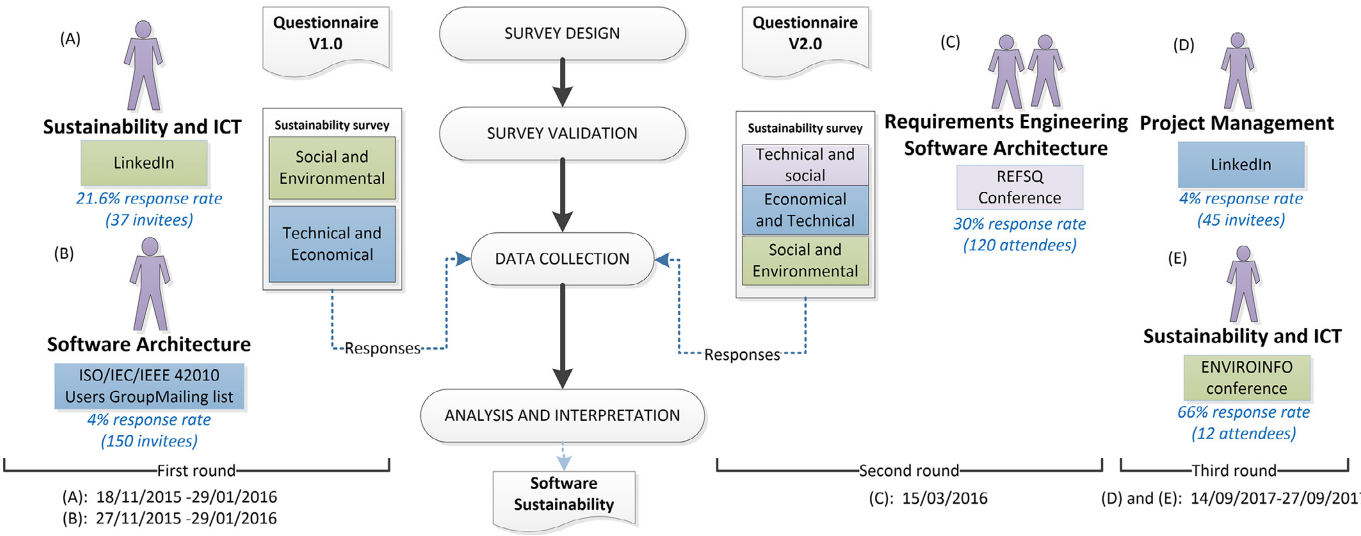

Fig. 3. Summary overview: target audiences and response rates.

Fig. 3 provides a summary overview of the different target audiences involved in the study and their corresponding response rate. The full dataset of this study is available online at https://tinyurl.com/y7uf4z4y.

3. Analysis of the results

This section provides our most significant observations on the obtained results. They are organized in three sections corresponding to our main research questions.

3.1. Analyzing the contribution of qualities to software sustainability (RQ1)

In order to determine how qualities contribute to the respective sustainability dimensions, we used a four-point ordinal scale (high, medium, low and null). A frequency distribution was calculated. Then, based on the obtained results (the frequencies), quality requirements were included in Tables 5–8, which report the frequencies of the quality requirements contributing to the four sustainability dimensions, respectively. The quality requirements are

Table 5
Social sustainability.

	Characteristics	Quality attributes	High	Medium	Sum
A1	Security	Confidentiality	42/50	6/50	48/50
A2	Security	Authenticity	35/50	11/50	46/50
A4	Satisfaction	Trust	33/51	14/51	47/51
A5	Freedom from risk	Health risk and safety risk mitigation	31/52	17/52	48/52
A3	Security	Accountability	29/50	17/50	**46/50**
A6	Security	Integrity	29/50	17/50	**46/50**
A7	Effectiveness	Effectiveness	29/51	15/51	44/51
A8	Satisfaction	Usefulness	26/51	22/51	**48/51**
A18	Compatibility	Interoperability	24/50	15/50	39/50
A14	Accessibility	Accessibility	22/50	25/50	**47/50**
A11	Freedom from risk	Environmental risk mitigation	21/51	24/51	**45/51**
A12	Usability	User error protection	20/50	25/50	**45/50**
A9	Usability	Operability	20/51	25/51	**45/51**
A13	Usability	Learnability	18/50	25/50	43/50
A15	Usability	Appropriateness recognizability	15/50	24/50	39/50
A16	Compatibility	Co-existence	11/50	28/50	39/50

Table 6
Technical sustainability.

	Characteristics	Quality attributes	High	Medium	Sum
A17	Functional suitability	Functional correctness	33/46	10/46	43/46
A18	Compatibility	Interoperability	33/46	12/46	45/46
A19	Reliability	Availability	32/46	8/46	40/46
A20	Functional suitability	Functional appropriateness	28/46	15/46	43/46
A8	Satisfaction	Usefulness	27/46	15/46	**42/46**
A21	Reliability	Fault tolerance	27/46	12/46	39/46
A22	Maintainability	Modifiability	26/46	18/46	**44/46**
A4	Satisfaction	Trust	26/46	14/46	40/46
A25	Portability	Adaptability	24/46	17/46	41/46
A23	Context coverage	Context completeness	24/46	15/46	39/46
A7	Effectiveness	Effectiveness	23/46	20/46	**43/46**
A27	Maintainability	Modularity	23/46	17/46	40/46
A26	Performance efficiency	Time behaviour	22/46	15/46	37/46
A24	Robustness	Robustness	21/42	17/42	38/42
A28	Maintainability	Testability	21/46	16/46	37/46
A29	Reliability	Recoverability	21/46	23/46	**44/46**
A16	Compatibility	Coexistence	19/46	22/46	41/46
A31	Efficiency	Efficiency	13/46	26/46	39/46
A33	Performance efficiency	Capacity	12/46	27/46	39/46
A30	Reliability	Maturity	14/46	24/46	38/46
A32	Survivability	Survivability	12/42	22/42	34/42

Table 7
Economic sustainability.

	Characteristics	Quality attributes	High	Medium	Sum
A7	Effectiveness	Effectiveness	5/5		5/5
A19	Reliability	Availability	4/5	1/5	5/5
A8	Satisfaction	Usefulness	4/5	1/5	5/5
A34	Freedom from risk	Economic risk mitigation	4/5	1/5	5/5
A29	Reliability	Recoverability	4/5	1/5	5/5
A20	Functional suitability	Functional appropriatness	4/5	1/5	5/5
A4	Satisfaction	Trust	3/5	2/5	5/5
A17	Functional suitability	Functional correctness	3/5	2/5	5/5
A35	Context coverage	Flexibility	3/5	2/5	5/5
A23	Context coverage	Context completeness	2/5	3/5	5/5
A31	Efficiency	Efficiency	2/5	3/5	5/5
A37	Functional suitability	Functional completeness	2/5	3/5	5/5

Table 8
Economic sustainability.

	Characteristics	Quality attributes	High	Medium	Sum
A38	Maintainability	Reusability	11/16	2/16	13/16
A39	Performance efficiency	Resource utilization	11/16	4/16	15/16
A31	Efficiency	Efficiency	9/16	6/16	15/16
A22	Maintainability	Modifiability	8/16	4/16	12/16
A16	Compatibility	Co-existence	8/16	7/16	15/16
A19	Reliability	Availability	7/16	4/16	11/16
A11	Freedom from risk	Environmental risk mitigation	6/16	6/16	12/16
A26	Performance efficiency	Time behaviour	6/16	6/16	12/16

clustered and colored based on the degree of contribution. Inclusion in the tables has been determined as follows:

- Quality requirements with a 'High' frequency value assigned by ≥ 60 % of the respondents are classified as *highly contributing*. This group of requirements is colored in green.
- Quality requirements with a 'High' frequency value assigned by <60% and ≥ 40 % of the respondents are classified as *contributing*. This group of requirements is colored in blue.
- Quality requirements with a 'High' frequency assigned by <40% of the respondents, but a 'High' and 'Medium' frequency values assigned by ≥ 80% of the respondents as *somehow contributing*. This group of requirements is colored in red.
- Quality requirements whose frequency values do not fulfill any of the above criteria are not includes in the tables. However, the interested reader can find the related details in the study dataset[5].

In particular, regarding the social sustainability dimension (Table 5), most of the security requirements (i.e. confidentiality and authenticity), health risk and safety risk mitigation, and satisfaction (in terms of trust) were rated as high contributors. Satisfaction (in terms of usefulness) was considered also as a relevant contributor to social sustainability (48 out of 51 respondents), followed by accessibility (47 out of 50 respondents) and security (in terms of integrity and accountability - 46 out of 50 respondents). Surprisingly, although most of the usability requirements were well ranked within group 2, only 15 respondents considered the "appropriateness recognizability" requirement as a high contributor, despite this usability requirement allowing users to recognize whether a system is appropriate for their needs. Similar results were obtained for learnability, where half of the respondents considered the relevance of both usability requirements but with a medium contribution. Another requirement that was considered as the least relevant by the respondents is "co-existence" (meaning that a product can perform its functions efficiently while sharing its environment and resources with other products).

Regarding quality requirements that contribute to technical sustainability (Table 6), most of the respondents considered functional correctness, functional appropriateness, interoperability, and availability as the best contributors. Modifiability and recoverability were also very well ranked since most of the respondents considered both qualities as good contributors (44 out of 46 respondents). As expected, other maintainability requirements like modularity and testability were also well ranked. However, despite the fact that maintainability (in terms of testability and modifiability), adaptability and interoperability requirements can benefit the flexibility of software systems, flexibility itself was only considered as a good contributor for the economic sustainability. For the technical dimension, flexibility was considered as a contributor (with high or medium ranks) but only by a total of 33 out of 46 respondents (and for this reason did not appear in Table 6).

Overall, all the qualities of Table 6 benefit the longevity of the software systems, as well as their appropriate evolution/adaptation in a constantly changing execution environment.

Regarding the economic sustainability dimension (Table 7), in contrast to the other three dimensions, here we considered only quality requirements that were ranked (as high) by at least 4 from 5 respondents. We did so due to the low response rate that provided feedback on this dimension (see Section 4 for further discussion). Our respondents were 3 senior software architects and 2 project managers (both were with more than 10 years of experience). Results show that economic risk mitigation, effectiveness, availability,recoverability, usefulness, and functional appropriateness tend to be the best contributors to this dimension. The other frequencies are shown just for completeness.

Finally, regarding quality requirements that contribute to environmental sustainability (Table 8), maintainability (in terms of reusability) and resource utilization were considered as the best contributors to this dimension. Although environmental risk mitigation and time behavior were also preferred by most of the respondents (12 of out 16 respondents), both qualities belong to the third group because their responses were distributed equally between high and medium contribution levels.

3.1.1. Major findings on QR contribution per sustainability dimension (RQ1.1)

Social sustainability. Our results regarding social sustainability confirm clearly the importance of the security requirement (in terms of confidentiality, authenticity and accountability) to contribute to the social sustainability, by means of providing an appropriate data/information access. However, as this resource access should be equal and equitable, the security requirement such as integrity was also considered as a good contributor, by preventing unauthorized access to, or modification of, data.

About the satisfaction requirement, trust and usefulness were also considered as good contributors to social sustainability. These results support previous research into the social technology acceptance area Wixom and Todd (2005), which links the user satisfaction with their perceived achievement of pragmatic goals (usefulness) and confidence on software systems that will behave as intended (trust). However, in contrast to the security requirements discussed previously, trust and usefulness still require more attention from the software engineering community Assefa and Frostell (2007) (e.g. to address the lack of objective measurements).

Another quality requirement that was also very well ranked was usability in terms of user-error protection and operability. This result is in line with previous works (e.g. Venters et al., 2014, Raturi et al. (2014)). However, despite the extensive amount of work already dedicated to usability (e.g. Panach et al., 2008; Ormeo et al., 2013; Nayebi et al., 2013), there is no consensus regarding its definition Abran et al. (2003). For instance, Venters considers the usability requirements but in terms of effectiveness, efficiency, and satisfaction; whereas in our work, usability is more considered as a property of the software system (i.e. learnability, operability, user error protection, and appropriateness recognizability).

Additional quality requirements that are very well related to usability and satisfaction are accessibility and freedom from risk. By means of accessibility, software system can be used by people with the widest range of characteristics and capabilities, equality and equity are positively favored. About freedom from risk, software systems must also mitigate the potential risk to people in the intended contexts of use. For instance, a software game can be highly usable, but also have addictive properties that encourage users to

[5] https://tinyurl.com/y7uf4z4y.

play more. If so, health risk and safety mitigation is very relevant for contributing to social sustainability, too.

Technical sustainability. Our study found that functional correctness, functional appropriateness, availability and interoperability are the best contributors, followed by modifiability and recoverability. Although these results slightly differ from the sustainability models proposed by Calero et al. (2013), Koziolek (2011) and Venters et al. (2014), they are consistent and provide further support for the concept of endurability of software systems.

For instance, interoperability, which refers to the ability of a software system to cooperate with other relevant software systems, contributes positively to software reuse (e.g. Khan, 2006). However, it is surprising that reusability (when an asset can be used in more than one system) did not appear in the ranking since it was considered as high or medium contributor only by 30 out of 46 respondents. Nevertheless, other maintainability requirements such as modularity and testability were better ranked. These results might be related to (i) the nature of both qualities, which can be measured by available and known direct metrics (e.g. cohesion, depth of inheritance tree), and (ii) the usefulness of these qualities for predicting external qualities like modifiability and reusability.

Another expected contributor was reliability in terms of availability, fault tolerance and recoverability. This finding has also important implication for endurability, by allowing that a software system can perform specified functions under specific conditions for a long period of time.

Economic sustainability. Our study found that six quality requirements contribute to the economic dimension, with effectiveness being the best contributor.

The current ranking list (see Table 7) further highlights the importance of addressing the user's goals (in terms of availability, usefulness, and functional appropriateness) that contribute to economic sustainability.

On the other hand, considering that availability of a software system may be increased by the strategy of focusing on increasing maintainability during the early design phase, a surprising result is that no maintainability requirement (i.e. testability) has been considered as direct contributor for this dimension such as it was for the technical and environmental sustainability dimensions. However, availability can be benefited by recoverability, since a software system could re-establish its desired level of performance in case of failure. This observation suggests the existence of potential direct or indirect relations among qualities from different dimensions or between qualities within the same dimension.

Environmental sustainability. Our study found that resource utilization (in the literature typically investigated in terms of energy consumption) was considered as the best contributor. Our respondents also acknowledged the importance of maintainability (in terms of reusability and modifiability) for contributing to this dimension. However, environmental risk mitigation and time behavior were considered as high contributors only by 6 respondents. This result can be due to the scarce of knowledge from our participants with respect to these specialized qualities that contribute to environmental sustainability.

3.1.2. Direct dependency relations analysis (RQ1.2)

According to Lago et al. (2015), software sustainability is defined in terms of four dimensions that are tightly interdependent. These dependencies can exist both among the four dimensions (level 1), and between the quality requirements within a dimension (level 2).

With the purpose of determining which direct dependencies of level 1 exist among the four sustainability dimensions, we performed an analysis of the relationships among the identified sustainability quality requirements. To this aim, we considered the following definitions.

Let $\mathcal{SD}_\alpha$ be an α sustainability dimension defined as a set of sustainability quality requirements: $\mathcal{SD}_\alpha = \{x | x \text{is a sustainability quality requirement}\}$.

We define a **direct dependency relation** $\mathcal{D}$ between two sustainability dimensions $\mathcal{SD}_\alpha$ and $\mathcal{SD}_\beta$ as a finite set of ordered pairs that is reflexive, symmetric and transitive:

$$(x, x) \in \mathcal{D} \iff \forall x \in \mathcal{SD}_\alpha \cap \mathcal{SD}_\beta$$

Moreover, we also define the **direct dependency strength** $\overline{\mathcal{D}}$ between two dimensions ($\mathcal{SD}_\alpha$ and $\mathcal{SD}_\beta$) as a ratio between the cardinality of the respective intersection and the minimum cardinality of the sustainability dimensions that form part of the intersection.

$$\overline{\mathcal{D}} = \frac{|\mathcal{SD}_\alpha \cap \mathcal{SD}_\beta|}{min|\mathcal{SD}_\alpha||\mathcal{SD}_\beta|}$$

In order to visualize all the direct dependencies between dimensions derived from the survey results, a Venn diagram was built (see Fig. 4). It shows the quality requirements that belong to the corresponding intersections. For example, the direct dependency between Technical and Environmental dimensions consists of five ordered pairs whose qualities requirements are: Co-existence (A16), Availability (A19), Modifiability (A22); Time behavior (A26), and Efficiency (A31). This type of direct dependency means that implementing each one of these qualities contributes positively to both technical and environmental dimensions. For instance, if a software system can co-exist with other independent software in a common environment by sharing resources, it will have a good contribution to the environmental sustainability dimension (i.e.less energy consumption). But, the technical dimension will be also affected positively because the software can efficiently perform its functionality.

Regarding the direct dependencies between social and environmental dimensions, two ordered pairs were identified whose quality requirements are: Co-existence (A16) and Environmental risk mitigation (A11). A software system that mitigates the potential risk to the environment, it will have also a positive impact on the users that interact with such software system (social sustainability).

Regarding the direct dependencies between technical and economic dimension, nine ordered pairs were identified whose quality requirements are: Functional Correctness (A17), Functional Appropriateness (A20), Context Completeness (A23), Effectiveness (A7), Recoverability (A29), and Efficiency (A31), Availability (A19), Usefulness (A8), and Trust (A4). This high number of identified ordered pairs implies that the dependency between these two dimensions is very strong since 9 out of 12 quality requirements of the economic dimension are directly related with the technical dimension.

The strengths of the corresponding direct dependencies among the four dimensions were calculated and shown in Fig. 5. From these results we observe that the technical sustainability dimension is strongly related to economic (0.75) and environmental (0.63) dimensions, whereas the strength of each direct dependency with the social sustainability dimension is lower.

We also observed that there are some specific quality requirements that are related to several dimensions. For instance, Trust (A4), Usefulness (A8), and Effectiveness (A7) enable to contribute positively to technical, social and economic sustainability dimensions. According to our results, there is no quality requirement that contributes to all four dimensions.

3.2. Ranking sustainability dimensions (RQ2)

In order to answer our second research question: "How do practitioners and researchers perceive the importance/relevance of

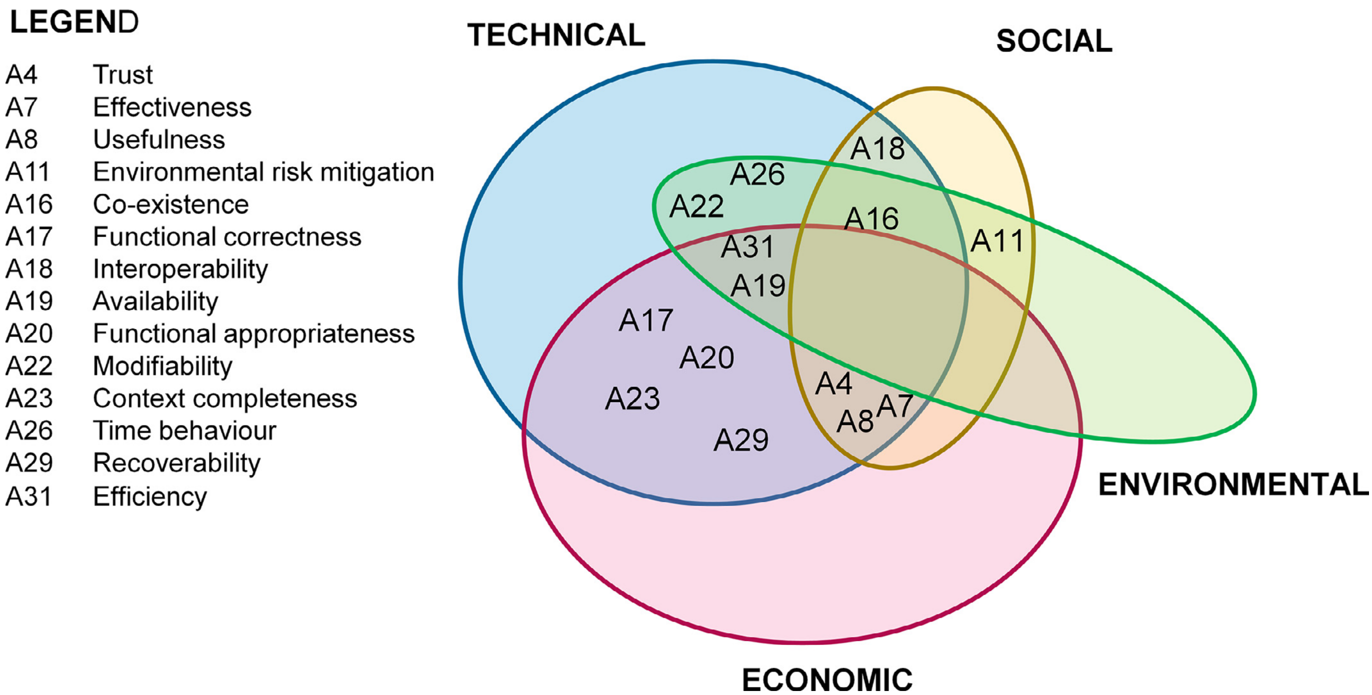

Fig. 4. Venn diagram of Sustainability quality requirements in direct dependency relations among dimensions.

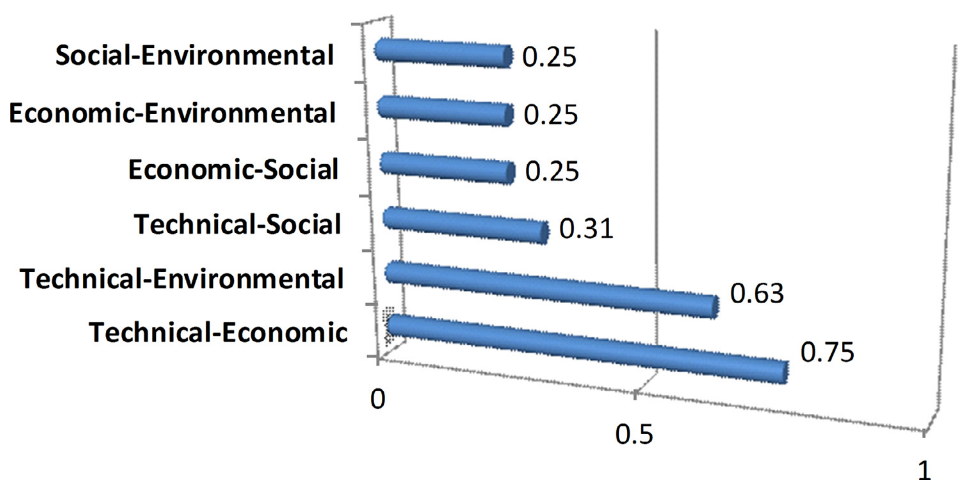

Fig. 5. Dependency strengths between sustainability dimensions.

each sustainability dimension?", respondents ranked the four sustainability dimensions based on their own viewpoint. According to the overall scores obtained for each dimension (see Figs. 6 and 7), economic dimension was ranked at the first or second place among most of the target audiences of this study (i.e. software architects, requirements engineers, ICT Sustainability practitioners, project managers).

It is also interesting to notice that in comparing the ranks among the target audiences, shown in Fig. 6, the environmental dimension was almost always overtaken by other dimensions. However this dimension was very well ranked by researchers on ICT Sustainability (see Fig. 7). This result may be explained by the fact that individuals may tend to judge factual data to be more important than abstract or conceptual data. For most of our industrial respondents, environmental sustainability could be perceived as less important because it is not yet well understood about how this dimension could be assessed/addressed in their software engineering practice.

From the software architect viewpoint, technical sustainability was the most relevant dimension. This was confirmed for requirements engineers, too. However, for the respondents with a background on ICT sustainability, the social dimension resulted as more relevant. As it was expected from this audience, the technical dimension was ranked as fourth and third place by practitioners and researchers respectively.

Moreover, by observing in Fig. 6 the scores from the third group of respondents (requirements engineering audience) there is no significant difference between technical/economic and environmental/social sustainability dimensions. However, this difference is noticeable in the first group (software architects), where social

Fig. 6. Overall ranks of software sustainability dimensions by software architecture (top), sustainability in ICT (middle) and requirements engineering (bottom) audiences.

and environmental sustainability were considered as less relevant, whereas technical and economic sustainability were ranked as the most relevant.

Discussion: The results support the idea that our five audiences tend to judge each sustainability dimension as important according to how familiar they are with the notion of sustainability. For instance, software architects are more familiar with the notion of *technical debt* Kruchten et al. (2012), which enables to optimize the cost of software system maintenance and evolution over time (i.e. in economic and technical sustainability, respectively).

3.3. Analyzing the perceived usefulness of the software sustainability model (RQ3)

Prior questions deal with the extent to which quality requirements may contribute to a sustainability model (i.e. technical and social dimensions). However, we also need to know more about the perceived usefulness on the sustainability model. Perceived Usefulness was evaluated with respect to the degree of agreement on how the sustainability model could be used to support the following activities:

- Quality requirements prioritization. Decision-makers often face the challenge of having more requirements than are possible to implement given different and dynamic constraints, such as time, cost, and other limited resources Condori-Fernández and Lago (2015). Prioritization approaches aim to aid the implementation of a software system with preferential requirements of stakeholders Achimugu et al. (2014). However, prioritization is a complex multi-criteria decision making process that stakeholders face in any phase of software development Cheng and Atlee (2007). We consider that using a software sustainability model can be crucial to distinguish the important requirements from the less important ones that contribute to properly address the four sustainability dimensions.
- Design of sustainable software-intensive systems. It is widely known that quality requirements influence on design decisions (e.g. Ameller et al., 2012, Gu et al. (2010), Chitchyan et al. (2016)). As an effective software design should ensure that all the quality requirements of a system of interest are supported Gu et al. (2010), we argue that a software sustainability model can aid designing sustainable software-intensive systems.

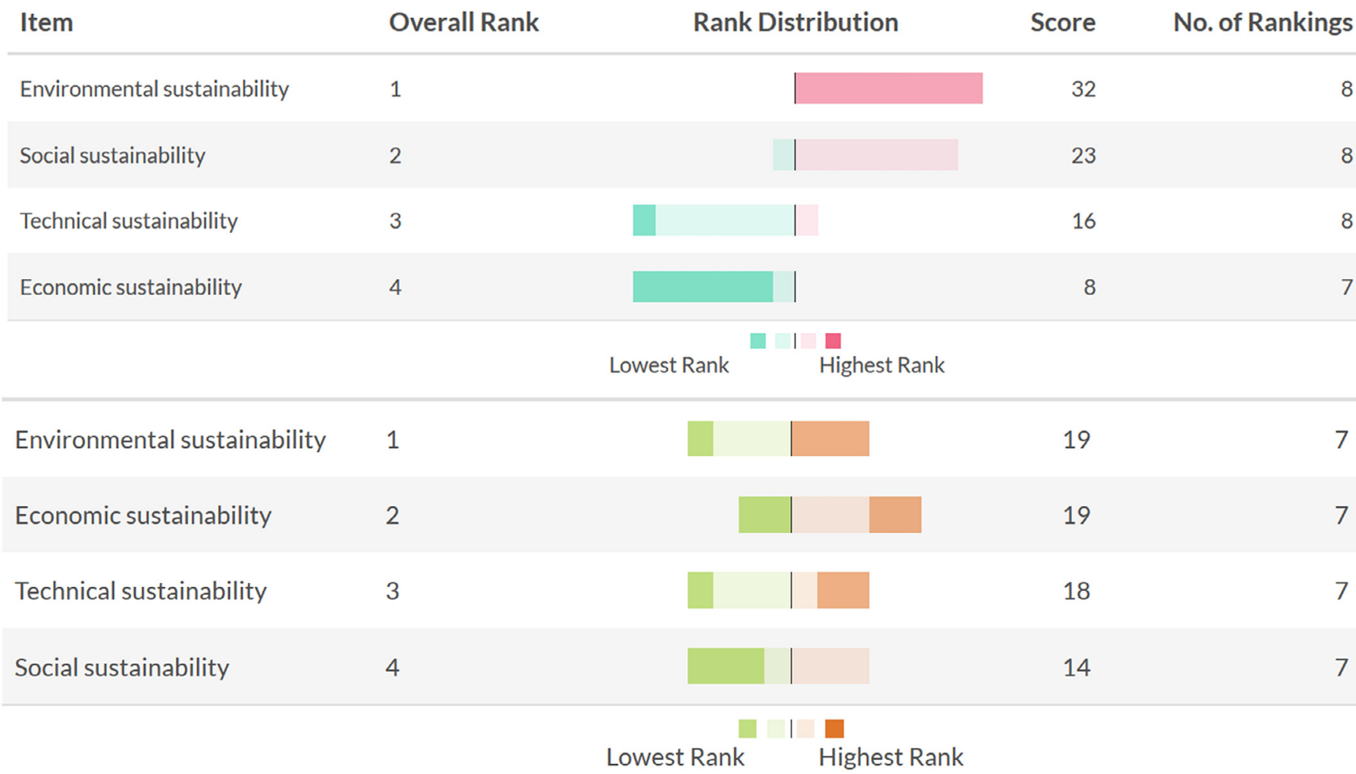

Fig. 7. Overall ranks of software sustainability dimensions by Researchers on ICT Sustainability (top), and Project managers (bottom).

Table 9
Frequency distribution on Perceived usefulness of the sustainability model.

Activity	−2	−1	0	+1	+2
QR prioritization	6.3%	12.5%	25.0%	**40.6%**	15.6%
Design of sustainable sw	0.0%	6.3%	*40.6%*	37.5%	15.6%
Sustainability assessment	3.1%	0.0%	31.3%	**62.5%**	3.1%
Trade- offs analysis	3.1%	3.1%	*43.8%*	34.4%	15.6%

Table 10
Pearson Chi-Square tests: Perceived usefulness of the sustainability model.

Activity	Chi-square	df	Asymp. Sig.
QR prioritization	11.437	4	0.022
Design of sustainable sw	10.750	3	0.013
Sustainability assessment	30.750	3	0.000
Trade- offs analysis	21.750	4	0.000

- Sustainability assessment of software-intensive systems. According to the ISO/IEC 25010 standard ISO/IEC (2010), the characteristics in the product quality model are intended to be used as a set when evaluating software product quality. Similarly, we consider that a software sustainability model can help performing quality assessments of software-intensive systems. In fact, some approaches have been proposed with this purpose (such as those discussed in the related work section).
- Trade-offs analysis among quality requirements. Requirements trade-off analysis is the systematic examination of advantages and disadvantages of requirements as well as the design choices for a system to achieve the right balance among several competing goals Alexander (2002). We argue that a software sustainability model can be help supporting the necessary trade-off decisions, by finding the most appropriate balance of quality requirements contributing to software sustainability.

Table 9 shows the frequency distribution on a 5-point Likert scale of the perceived usefulness for the corresponding activities. We also report the Pearson Chi-Square goodness of fit test in Table 10. Overall, the results show that the 65.6% of the 32 respondents perceived the sustainability model as useful for sustainability assessment and 56.2% of them for quality requirements prioritization as well. Only a minority of the respondents (below 18.8%) disagreed that the sustainability model can be useful for requirements prioritization. Similar results occur for the other activities such as sustainability assessment (3.1%), trade-offs analysis (6.2%) and sustainability design (6.3%).

By looking at the neutral answers, we observed that trade-offs analysis was the activity with the highest proportion of respondents as undecided (43.8%) followed by sustainability design (40.6%). It seems possible that these results are due to an insufficient knowledge/experience of the respondents for them to agree or disagree with the usefulness of the sustainability model in supporting both activities (trade-offs analysis and sustainability design). In order to get a better understanding about these results, a cross analysis was carried out with respect to the sector (academia and industry), and working experience of the respondent. We excluded from our analysis the government sector because we had only 1 respondent. The Pearson's chi-square test was also used to discover if there is a relationship between these two variables(factors).

In Fig. 8 we observed that a high number of respondents from both industry and academia agreed that the sustainability model can be useful for enhancing QR prioritization. However, it is also important to remark that a large proportion of undecided respondents were academics and practitioners with working experience

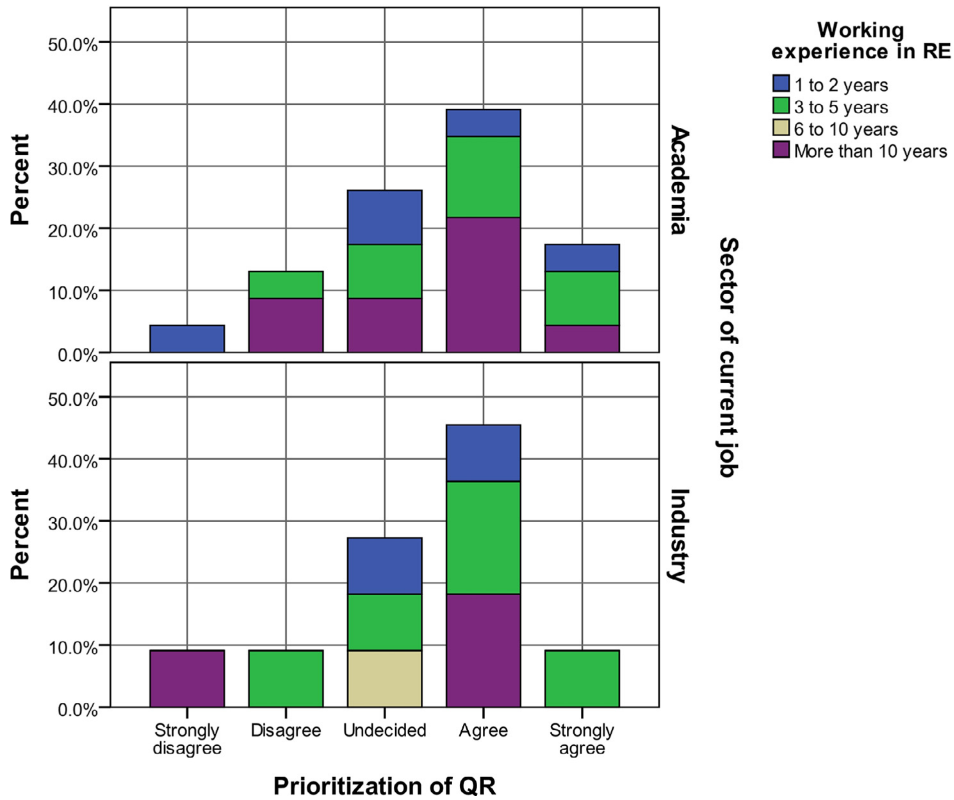

Fig. 8. Perceived usefulness in the Quality requirement prioritization.

between 1 to 5 years. Only four senior researchers (experience with more than 10 years) were undecided or in disagreement, and only 1 senior practitioner (out of 11) was in strong disagreement. Applying Pearson Chi-Square ((12) = 8.929, p = 0.709), we found there is no significant association between working experience and perceived usefulness. This means that all respondents, independently on their working experience, equally agree/disagree with the usefulness of sustainability model for improving QR prioritization.

As shown in Fig. 9, the overall response to the second question was not so positive since 43.5% of 23 respondents from academia was undecided. Although half of them correspond to junior researchers with a working experience between 1 to 5 years, all responses were distributed among almost all the categories of usefulness perceived, without having a clear consensus of agreement. Whereas the 54.5% from 11 respondents from industry agreed on considering the model as useful for designing sustainable software systems. Our results show that practitioners were the most positive. According to the Pearson Chi-square statistic ($\chi(6) = 3.67$, $p = 0.721$) there is not significance association between the job sector and perceived usefulness.

Regarding the third question on usefulness, as shown in Fig. 10, the results show a higher proportion of respondents from academia, 69.6% of 23, who agreed with the usefulness of the sustainability model for assessment purposes. However, the responses proportion from industry were divided between two categories: 45,5% of 11 practitioners were undecided whereas 55,5% showed a positive agreement. Applying the Pearson chi-square test, there is not an association between job sector and perceived usefulness ($\chi(6) = 6.102$, $p = 0.412$). Finally, with respect to usefulness of the model for trade-off analysis, Fig. 11 shows that higher proportion of responses was categorized as undecided. Although most of the undecided respondents correspond to novel researchers (1 to 2 years) and junior practitioners (2–5 years), we observe that there was not a clear consensus having even only two disagreements from the academia sector. Applying the Pearson chi-square test, there is not an association between job sector and perceived usefulness ($\chi(8) = 2.285$, $p = 0.971$).

4. Threats to validity

This section discusses the potential issues that may threaten the conclusion, internal and external validity of our study.

Conclusion validity.

- Reliability of results could be affected due to flaws in the instrument design. To mitigate this threat, our survey was validated with a pilot study, which was conducted with participants of the MEGSUS workshop (5 researchers and 3 practitioners).
- A potential threat due to the heterogeneity of the subjects regards a greater variability in the measures (e.g. participants' background). In order to mitigate this threat, we included demographic questions that we used to improve the interpretation of our results according to the profile of the participants. However, as this heterogeneity might also positively contribute to the external validity of our study, we involved participants from four different communities (software architecture, requirements engineering, ICT sustainability, and project management).

Internal validity.

- Respondents can be affected negatively (tired, bored) by an excessive survey length. We have mitigated this threat by assigning our participants with only the top-2 dimensions they are the most experienced with.

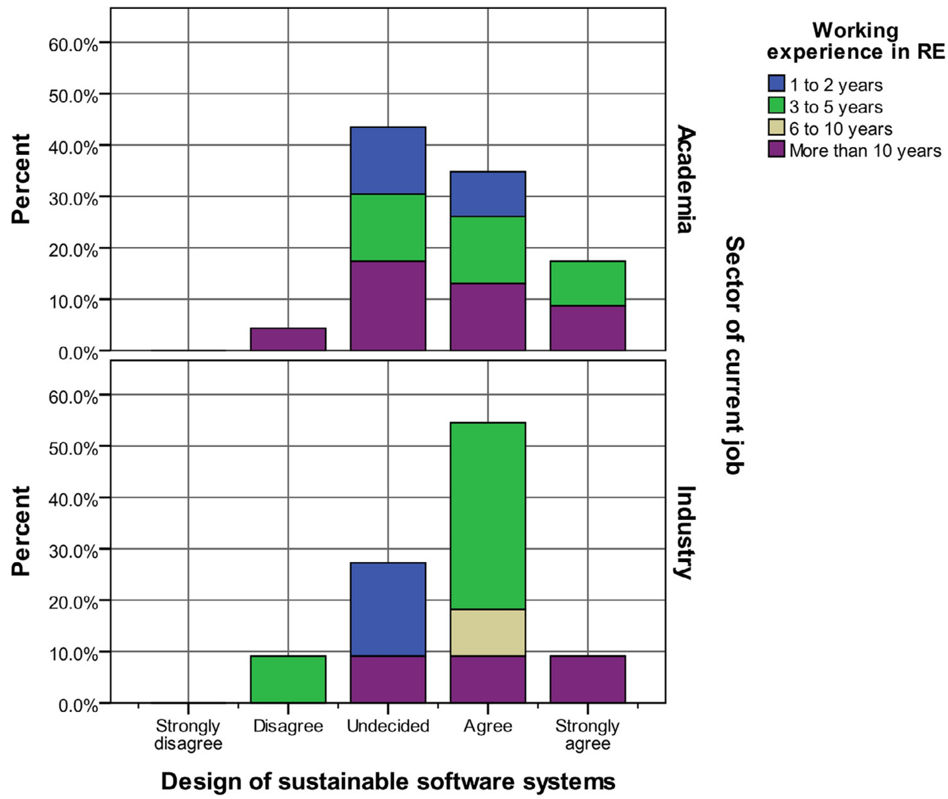

Fig. 9. Perceived usefulness in the design of sustainable software-intensive systems.

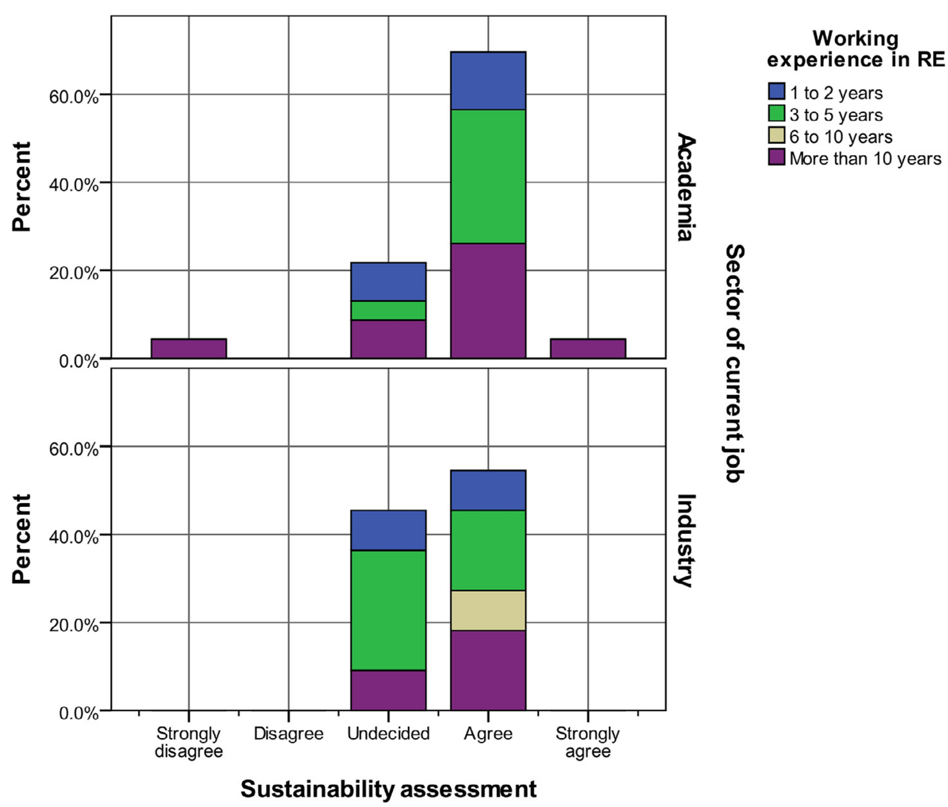

Fig. 10. Perceived usefulness in the assessment of sustainable software-intensive systems.

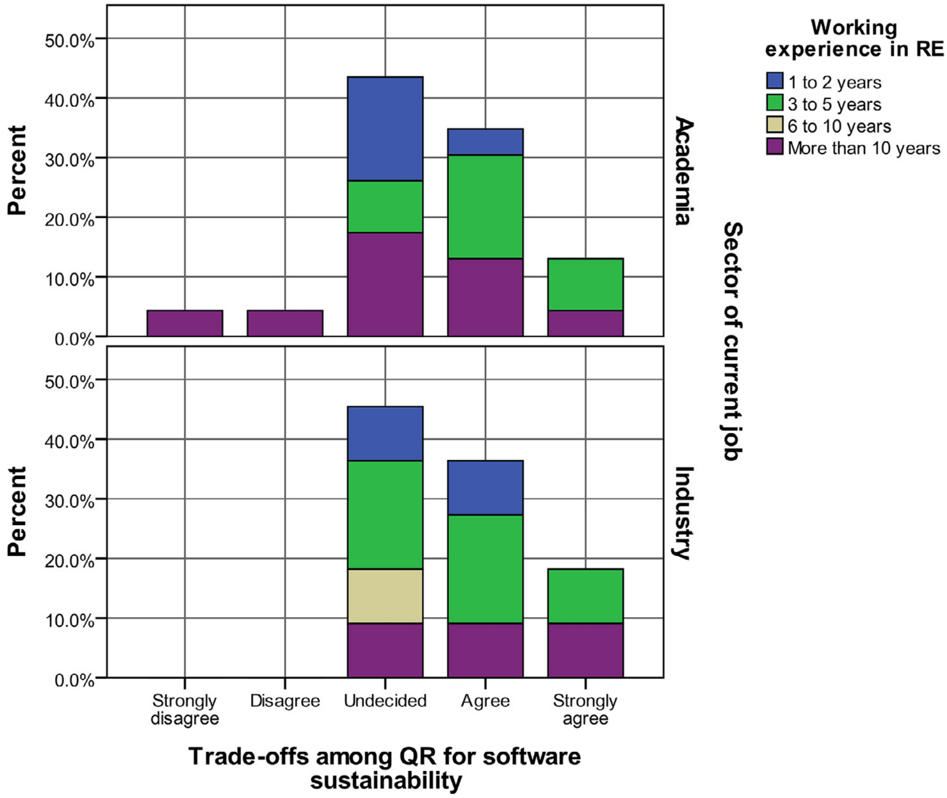

Fig. 11. Perceived usefulness in Trade-offs analysis among QR for sustainable software-intensive systems.

- Intrinsic in this type of study is that the perception of the participants on the usefulness of the sustainability model could be influenced by subjective issues (e.g. personal beliefs). To mitigate this threat, the survey was anonymous just to avoid subjective feelings and biases.
  Another threat to internal validity regards the high number of dropouts that were detected in the category of participants: project managers (9 dropouts among 11 respondents). Only 2 respondents completed the survey about the economic and technical dimensions. Four respondents dropped out after completing the demographic questionnaire, and 5 more after ranking the perceived importance of each sustainability dimension. This high drop out rate might indicate the immaturity of the project management practice in software sustainability, and hence the need for it to invest in developing sustainability competences.

External validity.

- A potential threat regards the sample selection, when unsuitable people participate in the survey. In our study, this threat was mitigated by carefully identifying the target audiences as based on the knowledge competences required for each sustainability dimension. However, after the survey conduction we realized that the respondents (i.e. project managers and software architects) had difficulties in answering questions related to the economic sustainability dimension. In hindsight, this quite reasonably reflects the immaturity of the field.
- There is a threat to external validity due to the low response obtained for the economic- and environmental dimensions. In order to mitigate this threat, we replicated the survey with (i) participants of the EnviroInfo 2017 conference track in energy-aware software engineering and development; and (ii) members of the LinkedIn group in Software Project Risk Management. This replication (third round) was very helpful for verifying the stability of our results obtained from the first and second rounds (see Appendix B).
- Regarding the high response rate for the social- and technical dimensions versus the low response rate for the economic- and environmental dimensions, we again argue that this is reflecting the relative lack of competences of ICT experts in the four dimensions. Understanding the social and technical implications of software-intensive systems is a much more consolidated practice (and hence the higher response rate) than the competences related to economic and environmental sustainability.

5. Related work

Being able to identify the impact of quality requirements on sustainability is the first step towards developing software-intensive systems that fulfill sustainability concerns *by design. Such requirements can be used for architecture design decision making, and ultimately for quality assessment.*

Software quality assessment as such is not new. Assessment based on the notion of sustainability as a software quality property, however, is still emerging and poorly understood Lago (2016). Consequently, how software should be assessed against sustainability concerns is still immature even though it is attracting increasing attention from both research and practice.

Software sustainability intrinsically manifests itself over time. Accordingly, and from a purely technical perspective, it has been linked to the notion of software evolvability, or longevity, e.g. Avgeriou et al. (2013). Of course, software sustainability has a much broader scope.

Venters (2014) discussed the notion of software sustainability based on the analysis of the literature. After debating if it should be considered as a non-functional requirement or an emergent property, the authors conclude it to be a multi-faceted concept and argue for a quantitative approach.

Originated in the GREENS workshop in 2013 Lago et al. (2014); 2015) defined a four-dimensional model that extends the social, environmental and economic dimensions (rooted in the Brundtland report Brundtland et al. (1987)) with a technical dimension. Later on, Lago introduced the Software Sustainability Assessment (SoSA) method Lago (2016), which helps scoping architectural concerns and quality requirements along the four dimensions above. In doing so, architects are supported during design decision making toward smart and sustainable software. Becker et al. (2016) have a similar approach but grounded in requirements engineering instead. In addition to the above four sustainability dimensions, they add the individual as a fifth sustainability dimension. We argue that the social dimension and the individual dimension share the same *social nature*. Differently, the first takes a broader perspective (e.g. organizations, society, stakeholder types). This is especially relevant in software architecture, which aims at capturing “the big picture”. The second dimension, instead, is appropriate whenever the concerns of the individual (e.g. end-user, citizen) should be addressed. This naturally comes forward more frequently in requirements engineering and human-computer interaction.

A much broader perspective has been taken by Hankel et al. (2014): inspired by the CMM, they defined the SURF Green ICT Maturity Model (SGIMM[6]) with the aim to assess the maturity of overall organizations with respect to Green ICT. To this aim, the SGIMM includes criteria in four areas, including *greening of ICT* and *greening of the primary processes*. In terms of software systems, these correspond to software energy efficiency and software energy awareness, respectively.

Based on the ISO/IEC 25010 Standard, Calero et al. (2013) provide a preliminary discussion of which quality characteristics should be considered in addressing software sustainability. As a next step, they propose the definition of a quality model where sustainability is part of the quality of software products. In contrast to our work, Calero et al. defined sustainability only in terms of energy consumption, resource optimization and perdurability (reusability, modifiability, and adaptability).

Finally, Venters et al. (2017) conducted a literature review to understand how the term *sustainability requirement* is used in software-related disciplines. The results reveal that the term is generally used ambiguously, but the sustainability dimensions help understand the level of abstraction of the source of the requirement.

6. Conclusions and future work

In this paper we have presented the main results of a survey that was conducted for empirically investigating how quality requirements may contribute to the sustainability of software-intensive systems.

The research has shown that the qualities we identified as good contributors to technical sustainability (Functional correctness, functional appropriateness, availability, modifiability, interoperability and recoverability) favor positively the endurability of software systems.

Moreover, despite most of the current studies focus on maintainability requirements (in terms of modifiability and modularity) for addressing endurability of software systems (e.g. Avgeriou et al., 2013; Kazman et al., 2015), our respondents considered some of maintainability requirements (in terms of reusability and modifiability) as relevant for addressing environmental sustainability, too. Our research also revealed the contribution of reliability requirements (in terms of availability and recoverability) to the economic- and technical sustainability dimensions.

This study has also identified five requirements, namely security, freedom from risk, satisfaction, accessibility and usability, as potential good contributors to social sustainability.

As satisfaction (in terms of trust and usefulness) is a key technology-acceptance factor that can fluctuate over time Wixom and Todd (2005), it is not surprising that this quality requirement was also considered by the respondents as a good contributor to economic sustainability.

We also identified a set of direct dependencies among the four sustainability dimensions, which were derived from the survey results. For instance quality requirements like availability and efficiency from the technical sustainability dimension are strongly related to the environmental and economic dimensions.

The results of this investigation complement those of earlier studies that also aim to define sustainability in terms of quality requirements (e.g. Calero et al., 2013; Venters et al., 2014; Raturi et al., 2014). In addition, our study provides (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.

Despite its exploratory nature, this study also offers valuable insights into the different potential usages of the envisioned software sustainability model from a requirements engineering viewpoint. We found that industry and academia agreed that a sustainability model can be useful for enhancing quality requirements prioritization and preforming software sustainability assessment. Moreover, in contrast to researchers, practitioners agreed on considering a sustainability model as useful for designing software-intensive systems.

With the purpose of pursuing the definition of our software sustainability model, we plan to interview stakeholders (e.g. software designers/architects) involved in industrial software-intensive systems projects. This will allow us to both improve the identification of the sustainability requirements, and investigate how sensitive are our survey results to specific domains.

Acknowledgment

We thank all the participants who took part in our survey research. We also thank Prof. Coral Calero for her valuable support in conducting the second round of this study. This work has received partial funding from the Netherlands Enterprise Agency (rvo.nl), project GreenServe on energy efficient software and green cloud; and It was partially supported by the Spanish Ministry of Economy, Industry and Competitiveness with the PGE and FEDER Projects: TIN2016-78011-C4-1-R and TIN2013-46238-C4-3-R.

Appendix A. Quality requirements in the survey

[6] https://goo.gl/f0ORLV.

Table A.11
Items of the sustainability survey.

ISO/IEC 25010:2011 Quality model	Qualities	Technical	Economic	Social	Environmental
Product quality model	**Compatibility**				
	Co-existence	x	x	x	x
	Interoperability	x	x	x	x
	Functional suitability				
	Functional appropriateness	x	x		x
	Functional corrected	x	x		x
	Functional completeness	x	x		x
	Maintainability				
	Analysability	x	x		
	Modifiability	x	x		x
	Modularity	x	x		
	Reusability	x	x		x
	Testability	x	x		
	Performance efficiency				
	Capacity	x	x		x
	Resource utilization	x	x		x
	Time behaviour	x	x		x
	Portability				
	Adaptability	x	x		
	Installability	x	x		
	Replaceability	x	x		
	Reliability				
	Availability	x	x		x
	Fault tolerance	x	x		x
	Maturity	x	x		x
	Recoverability	x	x		x
	Security				
	Accountability			x	
	Authenticity			x	
	Confidentiality			x	
	Integrity			x	
	Non-repudiation			x	
	Usability				
	Accessibility		x	x	
	Appropriateness recognizability		x	x	
	Learnability		x		
	Operability		x		
	User error protection		x		
	User interface aesthetics		x		
	Context coverage				
	Context completeness	x	x		x
	Flexibility	x	x		x
Quality in use model	**Effectiveness**	x	x	x	x
	Efficiency	x	x	x	x
	Freedom from risk				
	Economic risk mitigation		x		
	Environmental risk mitigation			x	x
	Health and safety risk mitigation			x	
	Satisfaction				
	Comfort	x	x	x	
	Pleasure	x	x	x	
	Trust	x	x	x	
	Usefulness	x	x	x	

Appendix B. Stability analysis

The following tables report our stability analysis carried out by comparing the contribution levels obtained from the first and second rounds against the third round. These contribution levels are expressed in a 4-points scale: 1 (highly contributing), 2 (contributing), 3 (somewhat contributing) and 4 (not contributing).

Then we scored each QR (last column), by assigning 2 points if there was no difference between the corresponding contributions levels; 1 point if the difference was minimal (1); and 0 points if the difference was greater than 2.

Table B.12
Stability analysis for social sustainability.

Characteristics	Quality attributes	Round 1&2	Round 3	Score
Security	Confidentiality	1	1	2
Security	Authenticity	1	2	1
Security	Accountability	1	4	0
Satisfaction	Trust	1	2	1
Freedom from risk	Health and safety risk mitigation	2	1	1
Security	Integrity	2	1	1
Effectiveness	Effectiveness	2	1	1
Satisfaction	Usefulness	2	2	2
Usability	Operability	2	4	0
Compatibility	Interoperability	2	2	2
Freedom from risk	Environmental risk mitigation	2	3	1
Usability	User error protection	2	4	0
Usability	Learnability	2	3	1
Accessibility	Accessibility	2	1	1
Usability	Appropriateness recognizability	3	4	1
Compatibility	Co-existence	3	2	1

Table B.13
Stability analysis for environmental sustainability.

Characteristics	Quality attributes	Round 1&2	Round 3	Score
Maintainability	Reusability	1	1	2
Maintainability	Modifiability	1	4	0
Performance efficiency	Resource utilization	2	1	1
Freedom from risk	Environmental risk mitigation	2	3	1
Performance efficiency	Time behaviour	2	3	1
Reliability	Availability	2	2	2
Efficiency	Efficiency	3	1	0
Compatibility	Co-existence	3	1	0

Table B.14
Stability analysis for economic sustainability.

Characteristics	Quality attributes	Round 1&2	Round 3	Score
Effectiveness	Effectiveness	1	1	2
Reliability	Availability	1	2	1
Satisfaction	Trust	2	2	2
Satisfaction	Usefulness	2	1	1
Freedom from risk	Economic risk mitigation	2	1	1
Context coverage	Context completeness	2	3	1
Context coverage	Flexibility	2	2	2
Functional suitability	Functional approriateness	2	1	1
Functional suitability	Functional correctness	2	2	2
Reliability	Recoverability	2	1	1
Efficiency	Efficiency	3	2	1
Functional suitability	Functional completeness	3	2	1

Table B.15
Stability analysis for technical sustainability.

Characteristics	Quality attributes	Round 1&2	Round 3	Score
Functional suitability	Functional correctness	1	2	1
Compatibility	Interoperability	1	1	2
Reliability	Availability	1	4	0
Functional suitability	Functional appropriateness	1	2	1
Satisfaction	Usefulness	2	4	0
Reliability	Fault tolerance	2	1	1
Maintainability	Modifiability	2	1	1
Satisfaction	Trust	2	1	1
Context coverage	Context completeness	2	4	0
Effectiveness	Effectiveness	2	2	2
Robustness	Robutsness	2	2	2
Portability	Adaptability	2	1	1
Performance efficiency	Time behaviour	2	4	2
Maintainability	Modularity	2	1	1
Maintainability	Testability	2	2	2
Reliability	Recoverability	2	1	1
Compatibility	Coexistence	2	2	2
Reliability	Maturity	3	3	2
Efficiency	Efficiency	3	3	2
Survivability	Survivability	3	2	1
Performance efficiency	Capacity	3	2	1

References

Abran, A., Khelifi, A., Suryn, W., Seffah, A., 2003. Usability meanings and interpretations in ISO standards. Softw. Quality J. 11 (4), 325–338. doi:10.1023/A:1025869312943.

Achimugu, P., Selamat, A., Ibrahim, R., Mahrin, M.N., 2014. A systematic literature review of software requirements prioritization research. Inf. Softw. Technol. 56 (6), 568–585. https://doi.org/10.1016/j.infsof0.2014.02.001.

Albertao, F., Xiao, J., Tian, C., Lu, Y., Zhang, K.Q., Liu, C., 2010. Measuring the sustainability performance of software projects. In: e-Business Engineering (ICEBE), 2010 IEEE 7th International Conference on, pp. 369–373. doi:10.1109/ICEBE0.2010.26.

Alexander, I., 2002. Initial industrial experience of misuse cases in trade-off analysis. In: Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on, pp. 61–68. doi:10.1109/ICRE0.2002.1048506.

Ameller, D., Ayala, C., Cabot, J., Franch, X., 2012. How do software architects consider non-functional requirements: an exploratory study. In: 2012 20th IEEE International Requirements Engineering Conference (RE), pp. 41–50. doi:10.1109/RE0.2012.6345838.

Assefa, G., Frostell, B., 2007. Social sustainability and social acceptance in technology assessment: a case study of energy technologies. Technol. Soc. 29 (1), 63–78. https://doi.org/10.1016/j.techsoc0.2006.10.007.

Avgeriou, P., Stal, M., Hilliard, R., 2013. Architecture sustainability [guest editors' introduction]. Softw. IEEE 30 (6), 40–44.

Becker, C., Betz, S., Chitchyan, R., Duboc, L., Easterbrook, S., Penzenstadler, B., Seyff, N., Venters, C., 2016. Requirements: the key to sustainability. IEEE Softw. 33 (1), 56–65.

Becker, C., Chitchyan, R., Duboc, L., Easterbrook, S., Penzenstadler, B., Seyff, N., Venters, C.C., 2015. Sustainability design and software: the karlskrona manifesto. In: Proceedings of the 37th International Conference on Software Engineering - Volume 2. IEEE Press, Piscataway, NJ, USA, pp. 467–476.

Betz, S., Becker, C., Chitchyan, R., Duboc, L., Easterbrook, S.M., Penzenstadler, B., Seyff, N., Venters, C.C., 2015. Sustainability debt: a metaphor to support sustainability design decisions. In: RE.

Brundtland, G., Khalid, M., Agnelli, S., Al-Athel, S., Chidzero, B., Fadika, L., Hauff, V., Lang, I., Shijun, M., Morino de Botero, M., Singh, M., Okita, S., 1987. Our Common Future (Brundtland report). Technical Report. World Commission on Environment and Development.

Calero, C., Moraga, M.Á., Bertoa, M.F., 2013. Towards a software product sustainability model. CoRR abs/1309.1640.

Cheng, B.H.C., Atlee, J.M., 2007. Research directions in requirements engineering. In: Future of Software Engineering, 2007. FOSE '07, pp. 285–303. doi:10.1109/FOSE0.2007.17.

Chitchyan, R., Becker, C., Betz, S., Duboc, L., Penzenstadler, B., Seyff, N., Venters, C.C., 2016. Sustainability design in requirements engineering: state of practice. In: Proceedings of the 38th International Conference on Software Engineering Companion. ACM, New York, NY, USA, pp. 533–542. doi:10.1145/2889160.2889217.

Condori-Fernández, N., Lago, P., 2015. Can we know upfront how to prioritize quality requirements? In: 2015 IEEE Fifth International Workshop on Empirical Requirements Engineering, EmpiRE 2015, Ottawa, ON, Canada, August 24, 2015, pp. 33–40. doi:10.1109/EmpiRE0.2015.7431305.

Gerostathopoulos, I., Keznikl, J., Bures, T., Kit, M., Plasil, F., Namesti, M., 2014. Software engineering for software-intensive cyber-physical systems.. In: GI-Jahrestagung. Citeseer, pp. 1179–1190.

Gu, Q., Lago, P., van Vliet, H., 2010. A template for soa design decision making in an educational setting. In: 2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications, pp. 175–182. doi:10.1109/SEAA0.2010.9.

Hankel, A., Oud, L., Saan, M., Lago, P., 2014. A maturity model for green ICT: The case of the SURF green ICT maturity model. In: 28th International Conference on Informatics for Environmental Protection (EnviroInfo). BIS Verlag, pp. 33–40.

Hölzl, M., Rauschmayer, A., Wirsing, M., 2008. Software-intensive Systems and New Computing Paradigms. Springer-Verlag, Berlin, Heidelberg, pp. 1–44. chapter Engineering of Software-Intensive Systems: State of the Art and Research Challenges.

ISO/IEC, 2010. ISO/IEC 25010 - Systems and Software engineering - Systems and Software Quality Requirements and Evaluation (SQuaRE) - System and Software Quality Models. Technical Report.

Jirotka, M., Goguen, J.A. (Eds.), 1994, Requirements Engineering: Social and Technical Issues. Academic Press Professional, Inc., San Diego, CA, USA.

Kazman, R., Cai, Y., Mo, R., Feng, Q., Xiao, L., Haziyev, S., Fedak, V., Shapochka, A., 2015. A case study in locating the architectural roots of technical debt. In: Proceedings of the 37th International Conference on Software Engineering - Volume 2. IEEE Press, Piscataway, NJ, USA, pp. 179–188.

Khan, H.S., 2006. Achieving reusability through interoperability. In: 13th Annual IEEE International Symposium and Workshop on Engineering of Computer-Based Systems (ECBS'06), pp. 2 pp.–484. doi:10.1109/ECBS0.2006.19.

Koziolek, H., 2011. Sustainability evaluation of software architectures: a systematic review. In: Proceedings of the Joint ACM SIGSOFT Conference – QoSA and ACM SIGSOFT Symposium – ISARCS on Quality of Software Architectures – QoSA and Architecting Critical Systems – ISARCS. ACM, New York, NY, USA, pp. 3–12. doi:10.1145/2000259.2000263.

Kruchten, P., Nord, R.L., Ozkaya, I., 2012. Technical debt: from metaphor to theory and practice. IEEE Softw. 29 (6).

Lago, P., 2016. Software and Sustainability [inaugural lecture]. http://dare.ubvu.vu.nl

Lago, P., Koçak, S.A., Crnkovic, I., Penzenstadler, B., 2015. Framing sustainability as a property of software quality. Commun. ACM 58 (10), 70–78.

Lago, P., Meyer, N., Morisio, M., Müller, H.A., others, 2014. Leveraging energy efficiency to software users: summary of the second GREENS workshop, at ICSE 2013. ACM SIGSOFT Software.

de Mello, R.M., Travassos, G.H., 2016. Surveys in software engineering: identifying representative samples. In: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. ACM, New York, NY, USA, pp. 55:1–55:6. doi:10.1145/2961111.2962632.

Nayebi, F., Desharnais, J., Abran, A., 2013. An expert-based framework for evaluating ios application usability. In: 2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement, Ankara, Turkey, October 23–26, 2013, pp. 147–155. doi:10.1109/IWSM-Mensura0.2013.30.

Nuseibeh, B., Easterbrook, S., 2000. Requirements engineering: a roadmap. In: Proceedings of the Conference on The Future of Software Engineering. ACM, New York, NY, USA, pp. 35–46. doi:10.1145/336512.336523.

Ormeo, Y.I., Panach, J.I., Condori-Fernandez, N., Pastor, 2013. Towards a proposal to capture usability requirements through guidelines. In: IEEE 7th International Conference on Research Challenges in Information Science (RCIS), pp. 1–12. doi:10.1109/RCIS0.2013.6577677.

Panach, J., Condori-Fernandez, N., Valverde, F., Aquino, N., Pastor, O., 2008. Understandability measurement in an early usability evaluation for model-driven development: An empirical study. In: Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. ACM, New York, NY, USA, pp. 354–356. doi:10.1145/1414004.1414080.

Pfleeger, S.L., Kitchenham, B.A., 2001. Principles of survey research: part 1: turning lemons into lemonade. ACM SIGSOFT Softw. Eng. Notes 26 (6), 16–18. doi:10.1145/505532.505535.

Raturi, A., Penzenstadler, B., Tomlinson, B., Richardson, D., 2014. Developing a sustainability non-functional requirements framework. In: Proceedings of the 3rd International Workshop on Green and Sustainable Software. ACM, New York, NY, USA, pp. 1–8. doi:10.1145/2593743.2593744.

Razavian, M., Procaccianti, G., Tamburri, D.A., 2014. Four-dimensional sustainable e-services. In: Gómez, J.M., Sonnenschein, M., Vogel, U., Winter, A., Rapp, B., Giesen, N. (Eds.), BIS-Verlag, pp. 221–228. ISBN 978-3-8142-2317-9.

Venters, C., Jay, C., Lau, L., Griffiths, M., Holmes, V., Ward, R., Austin, J., Dibsdale, C., Xu, J., 2014. Software sustainability: the modern tower of babel. RE4SuSy: Third International Workshop on Requirements Engineering for Sustainable Systems.

Venters, C., Lau, L., Griffiths, M., Holmes, V., Ward, R., Jay, C., Dibsdale, C., Xu, J., 2014. The blind men and the elephant: towards an empirical evaluation framework for software sustainability. J. Open Res. Softw. 2 (1).

Venters, C.C., Seyff, N., Becker, C., Betz, S., Chitchyan, R., Duboc, L., McIntyre, D., Penzenstadler, B., 2017. Characterising sustainability requirements: a new species, red herring, or just an odd fish? In: Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Society Track. IEEE Press, Piscataway, NJ, USA, pp. 3–12. doi:10.1109/ICSE-SEIS0.2017.2.

Wixom, B.H., Todd, P.A., 2005. A theoretical integration of user satisfaction and technology acceptance. Inf. Syst. Res. 16 (1), 85–102. doi:10.1287/isre0.1050.0042.