The Influence of Green Strategies Design onto Quality Requirements Prioritization

Nelly Condori Fernandez[1,2] and Patricia Lago[2]

[1]Universidade da Coruna, Spain,
[2]Vrije Universitiet Amsterdam, The Netherlands,
n.condori.fernandez@udc.es
n.condori-fernandez@vu.nl, p.lago@vu.nl

Abstract. [**Context and Motivation**] Modern society is facing important challenges that are critical to improve its environmental performance. The literature reports on many green strategies aimed at reducing energy consumption. However, little research has been carried out so far on including green strategies in software design.

[**Question/problem**] In this paper, we investigate how green software strategies can contribute to, and influence, quality requirements prioritization performed iteratively throughout a service-oriented software design process.

[**Methodology**] In collaboration with a Dutch industry partner, an empirical study was carried out with 19 student teams playing the role of software designers, who completed the design of a real-life project through 7 weekly deliverables.

[**Principle ideas/results**] We identified a list of quality requirements (QRs) that were considered by the teams as part of their architectural decisions when green strategies were introduced. By analyzing relations between QRs and green strategies, our study confirms usability as the most used QR for addressing green strategies that allow to create people awareness. Qualities like reliability, performance, interoperability, scalability and availability emerged as the most relevant for addressing service-awareness green strategies.

[**Contribution**] If used at the beginning of a green software project, our results help including the most relevant QRs for addressing those green software strategies that are e.g. the most domain-generic (like increase carbon footprint awareness, paperless service provisioning, virtualization).

Keywords: green software design, quality requirements, prioritization

1 Introduction

In the last decade, a number of green strategies have been discussed in the literature to achieve the goal of reducing energy consumption (e.g. [1,2]). However, few are the approaches that assist the inclusion of green strategies into the software development process. This has been corroborated by a recent qualitative

study [3], where most of the interviewed practitioners confirmed their inability to practice sustainability design within software engineering due to the lack of methodological- and tool support. In requirements engineering it is generally acknowledged that Quality Requirement (QR) prioritization is an important and difficult activity of the requirements management process [4]. The situation is even more complex in a services-oriented development environment, because software services are engineered with multiple sets of functional requirements to fulfill different groups of potential consumers with different QRs. Moreover, if environmental sustainability is targeted, new trade-offs emerge between environmental sustainability criteria (e.g.energy efficiency) and traditional quality requirements (e.g. usability, maintainability).
In an empirical study investigating the evolution of quality requirements prioritization during a service design process, Condori-Fernandez and Lago [5] identified *stable* quality requirements that can be considered as such from the beginning of a project, and *unstable* quality requirements that hence demand special attention throughout the project. In this paper, we investigate how the inclusion of green software strategies into a service design process can influence quality requirements prioritization. In order to assist the inclusion of green strategies into the design activities of the service-oriented design method proposed by Lago et al [6], we adapted the service design process by adding the "design space refinement" activity. To this aim, our research was conducted in the context of the Computer Science Master Track in 'Software Engineering and Green IT' [7] at the Vrije Universitiet Amsterdam in close collaboration with an IT company in the Netherlands. The study involved 19 teams of master students playing the role of software designers/architects, and who completed the design of a real-life green software project.

In this paper, we focus on green strategies that address the two targets of green software engineering: (i) *Green in software*, where the goal is to reduce the energy consumption and the resources used by the software itself, and (ii) *Green by software*, where the goal is on using software to deliver environmental-friendly systems in other domains.

The rest of this paper provides a detailed account of our study. Section 2 describes the extended green software model and service design process on which our work is based. Section 3 presents design of the empirical study. Then, data collection and preparation is described in section 4. Section 5 reports our analysis on influence of green strategies on QR prioritization. Sections 6 and 7 discuss the validity threats and conclude the paper, respectively.

2 Background

2.1 Extended Green Strategy Model

Gu et al. [8] proposed a green strategy model, which consists of a green goal that is realized by a number of green actions (see Fig. 1). In turn, each green action has a description explaining what the green action entails. Further, a green action typically belongs to one sub-category, which is a sub-set of category. A green action leads to at least one action effect, which causes a certain impact. A green strategy is a plan of green actions intended to accomplish a specific green goal. In order to assist the inclusion of green strategies into the design space specification, we have extended the green strategy model. As shown in Figure 1, the original model was extended by (i) mapping a green strategy (made of a green goal and a set of green actions) on the main elements of our design space (i.e. design concern, design option); (ii) adding the entity "quality requirement", which is used in the selection of design options; iii) considering not only an economic impact, but also social and environmental impacts; (iv) making explicit the fact that a green goal can regard one or more of the following three types of green problems in service-oriented software engineering [9]:

– People awareness: the use of the services and/or Service-Based Applications (SBAs) makes the users realize their sustainability footprint. The services/SBAs may further propose the users tips/alternatives that help them comply with some green strategy.
– Service awareness: the services/SBAs are designed in such way that their execution is environmental friendly (e.g. energy efficient). These types of systems implement strategies that enable their green execution.
– Process awareness: the services/SBAs are especially conceived to create or support a more sustainable development process.

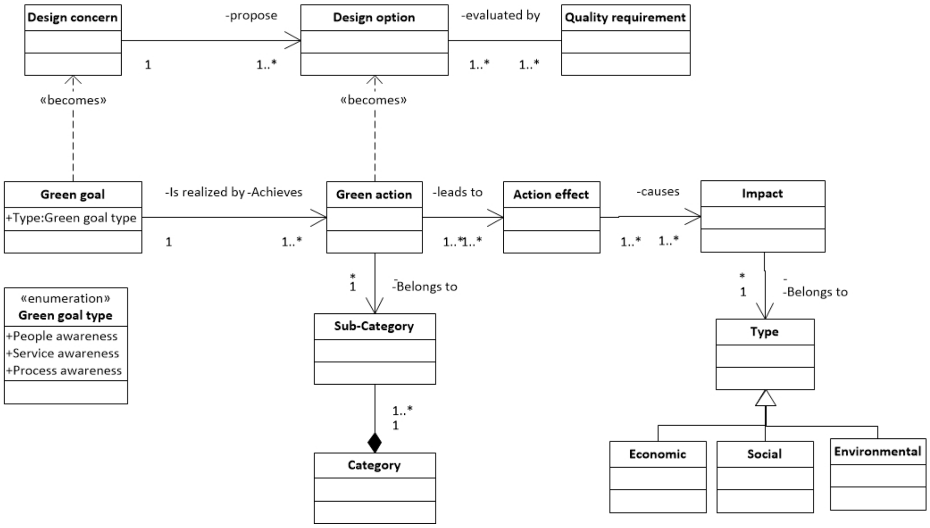


Fig. 1. Extended green strategy model, adapted from Gu et al.[8]

2.2 A Service-oriented Design Process

In service-oriented design, the delivered software may come in two flavors: (i) as an inventory of services that deliver independent functionality (in the form of software services) of some business value (e.g. for internal reuse or for provisioning to third-parties); (ii) as a SBA reusing (i.e. composing) software services.

Moreover, while in the first case the developer makes service inventories available after they have been developed, in the second case developing the SBA demands for composed services to be already available for the application to be specified, designed, and tested. If such services are not there, assumptions about their interfaces and delivered functionality are to be made. While designing SBAs resembles in many ways traditional software development where e.g. components are being reused, designing service inventories is much more challenging. This is why we focused for many years on teaching this second aspect of service orientation. Related challenges include identifying those functionalities that deliver stand-alone business-relevant services; and evaluating designed services against QRs that must be ensured once the services are composed in a third-party software system or SBA. To address these challenges and provide our students with the necessary competencies in service-oriented software design, we defined the service-oriented design method explained in [6] and [10]. It consists of the following four phases (see also lower part of Figure 2)

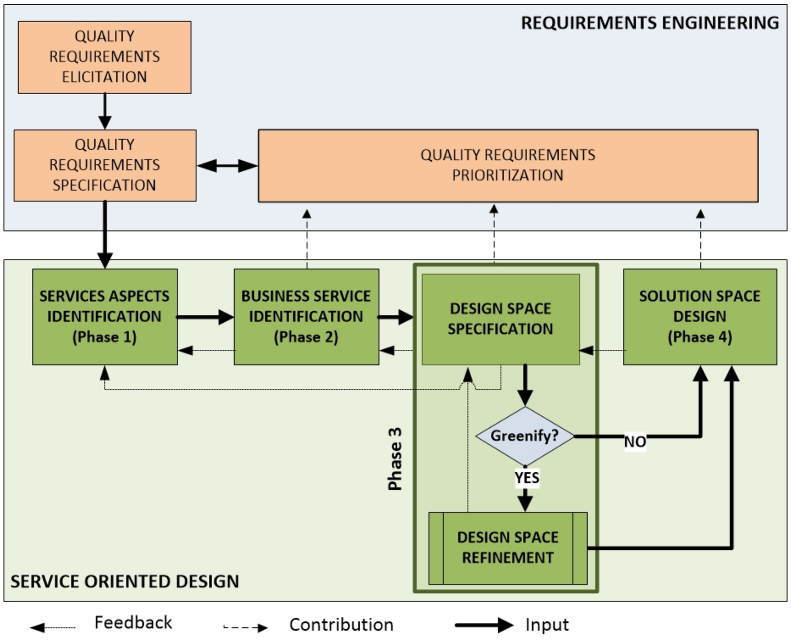

REQUIREMENTS ENGINEERING
QUALITY REQUIREMENTS ELICITATION
QUALITY REQUIREMENTS SPECIFICATION
QUALITY REQUIREMENTS PRIORITIZATION
SERVICES ASPECTS IDENTIFICATION (Phase 1)
BUSINESS SERVICE IDENTIFICATION (Phase 2)
DESIGN SPACE SPECIFICATION
SOLUTION SPACE DESIGN (Phase 4)
Greenify?
NO
YES
Phase 3
DESIGN SPACE REFINEMENT
SERVICE ORIENTED DESIGN
Feedback
Contribution
Input

Fig. 2. QR prioritization supported by the service-oriented design method adapted from [5]

Services aspects identification is grounded in the definition of service aspects as "those QRs that are especially relevant for service-oriented software" and that must be therefore addressed in a service-oriented software design [10]. Starting from the application domain of a selected software project, service aspects identification aims at selecting the QRs that are especially critical to achieve via services and for that domain.

Business service identification. Starting from the description of usage scenarios, functional- and quality requirements, business process models and conceptual data models are used to identify business services by clustering service-relevant functionality within a business process. The elements of the business process models (e.g. activities and decision points) are examined as candidate business services. In turn, the elements of the conceptual data models (e.g. data entities and their dependencies) are examined as candidate business services that will complement or complete the business services previously identified with the necessary data management features.

Design space specification explores the most important issues influencing the design of a software solution (e.g. in services for charging electric cars, an issue is "How to minimize idle time of charging points over night when demand is low?"). Possible design solutions for the identified issues (e.g. "allowing variable pricing", or "elastically shutting down charging points") are evaluated against relevant QRs (e.g. availability vs. energy efficiency), and design decisions are made by means of trade-offs. This phase is supported by a template capturing quality-driven design decisions and developed in [11] (see Table 1). Design space specification hence addresses issues, possible alternative design solutions, and (among them) the decided-upon solutions (or design decisions).

Design space refinement. To include the green strategies, the specification of the design space must be refined. This refinement can take two main forms:

- Extending the design space: the green strategies may lead to new design issues, options or criteria. Here the green goal is mapped onto the corresponding design concern; and the green actions are mapped onto design options, or new design concerns if large in scope.
- Challenging the already existing design decisions: in this case, introducing a green strategy causes changes in the decisions addressing pre-existing design issues. E.g., the green strategy might introduce a new green-related criterion, like energy efficiency, that in turn might lead to new trade-offs in favor of alternative design decisions.

  The redefined design space is captured by using the template for a green strategy description [12], the template shown in Table 1, and the Question-Options-Criteria (QOC) notation [13].

Solution space design takes as input the set of candidate business services and design decisions made during design space exploration. It identifies which candidate services must be designed as software services to support the initial usage scenarios. Then, it defines views that show how they can be composed in SBAs and how they should interact to deliver the target functionality and QRs.

Table 1. Capturing the knowledge about Quality-driven Design Decisions [11]

Concern (Identifier: Description)		*Con#<number>: What is the concern that needs to be solved (by taking a decision)? The description has to be in a form of a question)*
Ranking criteria (Identification: Name)		*What quality requirements have been used as criteria to take the decision based on the available options? (List the ranking criteria)* *Cr#<number>:*
Options (Repeat for each option)	**Identifier: Name**	*Con#<number>-Opt#<number> : Name of the option*
	Description	*Short description of the option*
	Status	*Has this option been* ***selected*** *or* ***rejected****?*
	Relationship(s)	*Indicate relationship with other options (by using their Identifiers): {forbids, conflicts with, enables, subsumes, is related to}*
	Evaluation	*Cr#<number N>: To which extent does this option support ranking criterion* *Cr#<number N>?*
	Rationale of decision	*Why has this option been selected or rejected? (use the ranking criteria identifiers in the argumentation)*

As shown Fig. 2, QR prioritization is an activity that is carried out throughout the whole service design process. Naturally, the outcomes of this process (e.g. identified design issues, design alternatives, trade-offs) contribute to a better requirements prioritization, too [5]. In this paper, our study focuses mainly on the outcomes of the design space specification and refinement, as well as the solution space design.

3 Empirical study design

This research is carried out with the purpose of empirically investigating how Green Strategies influence on Quality Requirements prioritization in the domain of smart transportation from the viewpoint of a software designer. The following specific research questions were formulated:

RQ1: Which are the most Quality Requirements used by designers when green strategies are included into the design process?
RQ2: How do Green Strategies influence on Quality Requirements prioritization?

3.1 Research Context

The study has been carried out in the context of a master-level course in Service Oriented Design (SOD) taught for over a decade [14]. It requires modeling and reasoning competencies. The course deals with how to analyze business and domain requirements to identify and design a satisfactory software services' offer for the identified requirements. Contents revolve around the service-oriented design methodology described in section 2.2.

About 20 percent of the 95 students participating in this course come from the Computer Science Master program with specialization in Software Engineering and Green IT [7], and 80 percent from Information Sciences Master program with specialization in Business Information Systems.

Table 2. Assignments of the EV-mobility project

Week	Phase	Assignment goal
1	Phase 1	A1: Identify quality requirements that are especially relevant for service-oriented applications
2	Phase 2	A2: Identify a set of relevant business services
3	Phase 3	A3: Document the design space according to the design space template shown in Table 1
4	Phase 3	A4: Reflect on and identify green strategies that facilitate accomplishing a specific environmental goal
5	Phase 3	A5: Refine the design space by adding the set of green strategies
6 ,7	Phase 4	A6: Document the design solution for your business services and address your design decisions

In the reported study, all participants were randomly grouped in 5-member teams, resulting in 19 groups. Each group was assigned to one of the 3 tutors in the course. All teams address the same real-life case project in parallel and independently. Students put theory into practice by collaboratively working in the project. As shown in Table 2, several assignments were distributed along 7 weeks. Each week all teams receive a review and feedback by the respective tutor. Moreover, an industrial stakeholder is invited to the students' progress presentations and gives feedback based on his or her expectations on the final product. Students are encouraged to present the least understood requirements and/or discuss issues encountered during the week.

3.2 Case Project: EV-mobility

The EV-mobility project (the case used in this study) focused on designing new software services on top of an existing charging point management platform, with the goal to facilitate the adoption of electric vehicles (EVs) in the Dutch private market. The project is proposed by one of our Green IT industrial partners.

Project-related concerns included: how electric cars can harmonize with people lifestyle, and how software services can create incentives for individuals to switch to electric cars, hence lowering their carbon footprint.

The main stakeholders of the EV-mobility project were:

EV Driver: early adopter driving an EV. Different motivation to do this: sustainability, new technology, marketing/image, low cost driving, etc.

Fleet owner: The owner of the car can be the EV driver, a Lease company or the driver's Employer. Risk for this stakeholder is battery lifetime, which is currently unpredictable. To monitor this risk, Fleet owners are interested in gathering battery-related statistics.

Charging point service provider (CPSP): It is operating a charging point network and providing services to EV drivers to increase utilization of the charging points and make life easier for EV drivers.

Distribution Service Operator (DSO): It is responsible to connect charging points to the electricity grid and may have a challenge, in the future, when EV's

use will increase and peaks will grow.
Energy Suppliers: supply energy to charging points.

4 Data Collection and Preparation

In order to answer our two research questions, we carried out a manual text analysis of the deliverables produced in the third and fourth phases of the service-oriented design process, namely Design space and solution space, which correspond to deliverables of the assignments A3, A4, A5 and A6. Table 3 shows the average page length per deliverable as well as the areas of interest (AoI) considered as relevant for our study. The data collection was carried out in two steps: extracting Green software strategies, and identifying Quality Requirements.
Regarding the first step, 16 clusters of green strategies were identified from all teams' Deliverables D4 and D5. The type of green strategies that were considered as relevant for the EV-Mobility project are shown in the Appendix A (Table 6). Ten clusters were identified as domain generic strategies, whereas 6 clusters were considered as domain-specific strategies. More details of the green strategies extraction can be found in [12].

Regarding the Quality Requirements Identification, in this step we proceeded to identify: (1) QRs that were kept or added when a green strategy was integrated in the design space phase (Deliverables D3 and D5); and (2) QRs that were added at the solution space design phase after including the green strategy (Final report).

Firstly, we analyzed deliverable D3 (i.e. design decisions tables, QOC) in order to identify which quality requirements (service aspect) were considered as relevant for deciding on any of the options identified for each concern specified in the design space. Then, by analyzing the deliverable D5 (mappings between design space and green strategies), we located the new design concerns or existing ones that could have been added or modified as consequence of including a green strategy. Their respective design decision tables and QOCs were then analyzed to extract the corresponding QRs that were used as ranking criteria for evaluating the respective design options. During this extraction, synonyms were verified by

Table 3. Data sources: Areas of Interest per deliverable. $^{(*)}$Design Decision Tables follow the template presented in Table. 1.

Deliverable	**Area of Interest**	**Average page length**
D3: Design space	Design Decisions Tables$^{(*)}$, QOC	15
D4: Green strategies	Strategies descriptions, graphical representation	10
D5: Design space with Green Strategies	Design Decisions Tables$^{(*)}$, QOC, Mappings	35
Final report	Quality Requirements list	4

reviewing the QR definitions and respective justifications given by the teams. By means of this analysis, we discarded two deliverables. This is because one of the teams did not consider QRs as design criteria (e.g. cost, behavioral impact). A second team focused only on describing green strategies without making a mapping to design concerns of sufficient quality. In the similar way, we proceed with the last deliverable (final report) to extract the corresponding new QRs that were considered at the design solution phase.

Nominal data was collected (yes=1; no=0), by assigning 1 when i) the QR was considered as a ranking criterion for the selection of design options introduced by the green strategies; ii) a new QR was added in the solution space design phase due to the introduction of a green strategy. In order to collect this data from the projects developed by the 19 teams, we created an Excel sheet template as shown in Appendix B (Figure 4).

5 Results and Discussion

5.1 Most used QRs when green strategies are included into the design process (RQ1)

With the purpose of identifying first which QRs were considered by design teams when green strategies were involved in the service design process,we calculated the frequency distribution of all QRs used in design space specification, design space refinement, and solution space design across all the projects. As shown in Fig. 3, 20 different QRs were used for designing the EV-mobility project. We observed that some QRs, like performance, privacy, accuracy, and efficiency, were consistently used before-, during- and after including a green strategy. The stability of these QRs can be explained due to the functionality required by the project (e.g. EV battery should be charged every time the service is requested). However, some others QRs like usability, reliability, availability, security, interoperability, and scalability were also considered as relevant during the design process but with lower stability. We also identified some QRs that were less frequently used by the teams throughout the design process. These QRs are auditability, portability, flexibility, maintainability, and testability. What is surprising is that according to our sustainability definition (cf. [15,16]), most of these QRs contribute to software longevity (i.e. the technical sustainability dimension). This result can be due to the nature of the specific green software project, which aims to primarily address social and environmental issues.

In order to understand more clearly how the most used QRs can contribute to prioritization when green strategies were included into the design process, our analysis focused on the QRs that were considered in the design space refinement at least by 3 teams. Table 4 shows the number of QRs that were added during the refinement or after (solution space design phase); and QRs that were already identified earlier (design space specification), but kept still as important during the refinement as a consequence of including green strategies. From this data, we found that the most-used QRs, except environmental sustainability and energy efficiency, were kept as relevant from the design space specification. These

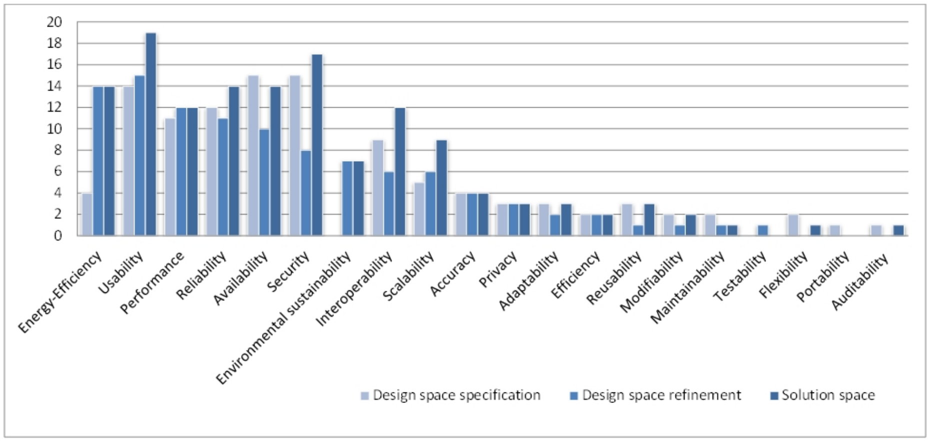


Fig. 3. Frequency distribution of influenced quality requirements used at design space specification, Design space refinement and solution space design

results could have important positive implications for the development of green software projects. For example, reusing these technical QRs, that had been already considered as relevant before including green strategies at design space, can be economically beneficial for the software project. Only security and interoperability (see grayed rows) were added after the refinement by most of the design teams, followed by scalability.

Next, we analyze and discuss main results on the influence relation between green strategies and QRs considered as the most important in the design of a green software project.

Table 4. Quality requirements added and/or kept during and after design refinement

	During design space refinement					
	Kept from		**Added at**		**Added at**	
	Design space specification		**Design space refinement**		**Solution space design**	
Energy-Efficiency	4	29%	10	71%	0	%
Usability	10	67%	5	33%	4	21%
Performance	10	83%	2	17%	0	0%
Reliability	9	82%	2	18%	3	21%
Availability	11	100%	0	0%	4	29%
Security	6	75%	2	25%	10	59%
Env Sustainability	0	0%	7	100%	0	0%
Interoperability	6	100%	0	0%	6	50%
Scalability	4	67%	2	33%	4	44%
Accuracy	4	100%	0	0%	0	0%
Privacy	2	67%	1	33%	1	33%

5.2 Green Strategies influence on Quality Requirements Prioritization (RQ2)

We considered the first thirteen clusters of green strategies shown in Table 6. The last three clusters were excluded because of their size (1 strategy only). As a green strategy can have an influence on a QR only if such QR is used for the selection of a design option that introduces a green action, we first calculated the frequency distribution of the most used QRs per green strategy. In order to answer RQ2, we calculated i) the total number of influenced QRs by the respective cluster of strategies (shown in the last row of Table 5), and ii) the total number of clusters that influence on a QR (last column of Table 5).

From this data we can observe that all clusters of **people-awareness strategies** influence on Usability (e.g. G1, G4, G5, G8, G11). Due to the type of green problem that the project aims to solve (making users become more aware of their own sustainability footprint), this result confirms that designers took on the challenge of designing usable services not only to improve user experience and user acceptance (e.g. [17]), but also to foster awareness among potential service consumers (e.g. non EV-drivers). However, it is surprising that none of the teams considered persuasiveness as a ranking criteria of the design concerns, despite its importance in the design of interactive systems [18]. This can be due to the traditional software quality models (e.g., [19,20,21]) used by the teams, which did not include persuasion as QR. Another important observation is that people-awareness clusters G1 and G4 yield the highest number of dependencies with QRs. For instance, for G1 (Raise carbon footprint awareness), besides usability and environmental sustainability, designers considered also security, reliability, privacy, accuracy and availability as relevant qualities for evaluating different design options (e.g. rank users on carbon footprint, display tips). Moreover, as cluster G1 contains domain-generic green strategies, the probability of reusing these influenced QRs at any other domain can be high. In a similar way, this capability of reuse applies also to the five QRs influenced (i.e. usability, enviromental sustainability, performance, interoperability and scalability) by the cluster G8 (Paperless service).

Regarding **service-awareness strategies**, our results show that availability was the quality mostly influenced by green strategies of clusters G2, G3 and G7. Moreover, for these three clusters we observed a good number of dependencies with QRs (last row). For instance cluster G7 (reduce carbon footprint caused by databases) influence reliability, performance, availability, energy efficiency and scalability. The importance of some of these QRs (reliability, performance, availability) was also confirmed by other surveys-based studies on QR prioritization (e.g. [22]). Finally, from the last column of Table 5, we corroborated that energy efficiency and environmental sustainability were influenced by most of the green strategies (11 out of 13 clusters). Few clusters like G13 (Create a green cloud of energy) did not have any dependencies with these two QRs. However this observation may be limited by the design teams who proposed such strategies in cluster G13.

Table 5. Green strategies and Quality requirements

							Service awareness							
			Process awareness											
	People awareness											People awareness		
Qualities	**G1**	**G5**	**G4**	**G9**	**G12**	**G13**	**G2**	**G6**	**G10**	**G3**	**G7**	**G8**	**G11**	**Count**
Usability	11	3	3		1		3	2	1	1	1	2	2	11
Energy efficiency	4	2	2	2	1		5	2	1	2	3		1	11
Env Sustainability	5	2	2	1			1	1		3	1	2	1	10
Reliability	4		1	2			2	1		4	4		1	8
Performance	2		1	3		1	2		1	3	3	1		9
Interoperability	2		1				3		1	2	1	1		7
Availability	3		2	1			5		1	4	3			7
Scalability	1				1		2			1	2	1		6
Accuracy	3	1					2	1	1					5
Security	4		2	1						1	1			5
Privacy	3		2				1	1						4
Efficiency		1				1	1				1			4
Count	11	5	9	6	3	2	11	6	6	9	10	5	4	

We also observed that for cluster G8 (Paperless service), designers considered only environmental sustainability as ranking criterion for their design decisions. This result may be explained by the fact that environmental sustainability is a broader topic that covers also energy efficiency.

Looking at the last row of Table 5 and considering only domain-generic strategies (text highlighted in blue), we identify: *Raise Carbon Footprint awareness (G1)* as the most influential people-awareness green strategy of the EV mobility project and *Reduce Carbon Footprint caused by DB (G7)* as the most influential service-awareness green strategy. This is because most of the qualities (11 QRs and 10 QRs respectively) were used during or after the design refinement as as a result of implementing both strategies.

Our results differ from our first empirical study [5] in the following way: i) New requirements like energy-efficiency, environmental sustainability, and privacy emerged as important/relevant requirements during design space refinement and solution space design. This was a consequence of including green strategies (e.g. use most efficient charging points) into the design process. ii) Security, scalability and interoperability are requirements that were considered as important/relevant by almost half of the design teams *after* the inclusion of green strategies (e.g. Reduce Carbon Footprint in EV)

On the other hand, requirements like usability, performance, reliability, and availability were considered as relevant since the design specification of the EV project. These results are consistent not only with the data obtained in our first study [5], but also with the top-five list identified by Ameller et al.[22]. This top list was derived from an empirical study (interviews with software architects) and consisted of the following quality requirements: Performance, usability, security, availablity, interoperability.

6 Threats to Validity

The following discusses the threats to the validity of our results and provides rationale for our related design decisions.

Internal validity. As software design is a socially intensive activity, the study was organized in 19 teams clustered in 3 working groups with 3 different tutors. In this way, our threat of selection bias was partially mitigated by assigning the teams to the working groups randomly. Expertise level of tutors was similar. Maturation was another potential threat in our study because the data was collected in different phases. However, the effect that the subjects can react differently as time passes was mitigated by various mechanisms meant to keep the students motivated: (i) weekly feedback from industrial stakeholders, (ii) weekly competition among working groups, and (iii) a prize competition for the best project selected by the company that proposed the EV-mobility project. It is also important to notice that weekly competition helped avoiding plagiarism among teams. Moreover, with the purpose of not limiting the service aspects identification to a pre-defined quality standard model, the design teams freely choose QRs not only from a set of selected articles on the topic of quality attributes for SOA were used by the students (e.g., [19,20,21]) but also from "Web searched" electronically available papers. Although we consider this flexibility as very positive for stimulating design creativity in our students, the list of QRs that were identified in this study is limited to these different sources of qualities. Moreover, given some qualities might be named in different ways, synonyms were verified during data collection. In a possible future replication it would be interesting to use a reference quality model for services, like S-Cube[1].

Construct validity. Researchers can bias the results of a study both consciously and unconsciously. This threat was mitigated since tutors' feedback focused specifically on methodology issues (e.g. correct use of the template for capturing design decisions) without influencing on the decisions themselves required by the EV-mobility project. Our analysis for identifying QRs affected by the inclusion of green strategies may be threatened due to QRs removed during or after the design refinement were not considered in this study.

External validity. Although master students participated in this study, we think that our setting was representative since students had an industrial project featuring a real case. Besides, the project involved also a high number of domain-generic strategies. Regarding sample size, we had "only" 19 teams. This could have been easily mitigated by reducing the size of teams (e.g. 3 instead of 5 students per team). However, in our experience 5-person teams is more realistic yet necessary to deliver quality results in such large-scale industrial cases. Besides, it would have been unfeasible to tutor more teams with the three available tutors.

7 Conclusions and Further Work

The main goal of this study was to determine how green strategies can contribute to, and influence, QR prioritization performed iteratively throughout a service-oriented software design process.

[1] http://www.s-cube-network.eu/km/qrm

Our study was conducted in an academic setting but with a real-life industrial project.

The study has shown that technical QRs like usability, performance, reliability, availability, interoperability, and scalability were identified before the refinement but considered still as important for designing green strategies. Instead, some other QRs like security and interoperability emerged after the refinement. This suggests that both QRs might require deeper understanding of the solution space before being able to determine their relevance in the green software design. Only green-related QRs (i.e. energy efficiency, environmental sustainability) were added during the design space refinement by most of the teams.

Another interesting result from our analysis of QRs and the most used green strategies is that usability emerged as the most relevant QR for addressing people-awareness green strategies, followed by reliability and security. Other QRs like privacy, availability and accuracy were also considered but with lower intensity.

For service-awareness green strategies, instead, availability was the most relevant QR. Our study has also found that certain clusters of service-awareness green strategies (e.g. reduce carbon footprint caused by databases) had a higher number of influenced QRs than others (e.g. reduce carbon footprint of charging points). This result implies that the implementation of such green strategies could become more complex due to the higher number of QR dependencies.

Our results help with QR prioritization by including since a project's inception, the most relevant QRs for addressing those green strategies that are domain-generic (e.g. reduce services' carbon footprint). Moreover, this research can serve as a basis for future replications in different domains: this would be beneficial to build empirical knowledge that can be reused for both developing green software, and modernizing legacy software to address environmental issues.

Acknowledgment

This work has received partial funding from the Netherlands Enterprise Agency, with the project GreenServe. It was also partially supported by the Spanish Ministry of Economy, Industry and Competitiveness with the Project: TIN2016-78011-C4-1-R, and Galician Government with the project: ED431C 2017/58.

References

1. J. John, Green Computing Strategies for Improving Energy Efficiency in IT Systems, International Journal of Scientific Engineering and Technology 717 (36) (2014) 2277–1581.
2. S. Murugesan, Harnessing green IT: Principles and practices, IT Professional 10 (1) (2008) 24–33. doi:10.1109/MITP.2008.10.
3. R. Chitchyan, C. Becker, S. Betz, L. Duboc, B. Penzenstadler, N. Seyff, C. C. Venters, Sustainability design in requirements engineering: State of practice, in: Proceedings of the 38th International Conference on Software Engineering Companion, ICSE '16, ACM, New York, NY, USA, 2016, pp. 533–542.

4. B. H. C. Cheng, J. M. Atlee, Research directions in requirements engineering, in: Future of Software Engineering, 2007. FOSE '07, 2007, pp. 285–303.
5. N. Condori-Fernández, P. Lago, Can we know upfront how to prioritize quality requirements?, in: IEEE Fifth International Workshop on Empirical Requirements Engineering, EmpiRE 2015, Ottawa, ON, Canada, August 24, 2015, pp. 33–40.
6. P. Lago, M. Razavian, A pragmatic approach for analysis and design of service inventories, in: Proceedings of the 2011 International Conference on Service-Oriented Computing, ICSOC'11, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 44–53.
7. P. Lago, A master program on engineering energy-aware software, in: 28th International Conference on Informatics for Environmental Protection: ICT for Energy Efficiency (EnviroInfo), 2014, pp. 469–476.
8. Q. Gu, P. Lago, S. Potenza, Aligning economic impact with environmental benefits: A green strategy model, in: First International Workshop on Green and Sustainable Software (GREENS), 2012, pp. 62–68. doi:10.1109/GREENS.2012.6224258.
9. P. Lago, T. Jansen, Creating environmental awareness in service oriented software engineering, in: Proceedings of the 2010 International Conference on Service-oriented Computing, ICSOC'10, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 181–186.
10. Q. Gu, P. Lago, E. Di Nitto, Guiding the Service Engineering Process: The Importance of Service Aspects, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 80–93.
11. Q. Gu, P. Lago, H. van Vliet, A template for SOA design decision making in an educational setting, in: Proceedings of the 2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA, IEEE Computer Society, Washington, DC, USA, 2010, pp. 175–182.
12. N. Condori-Fernandez, P. Lago, Analyzing Green Software Strategies within a Service Design Process, 31st Edition, Shaker Verlag, 2017, pp. 101–110.
13. T. P. Moran, Design Rationale: Concepts, Techniques, and Use, L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1996.
14. P. Lago, H. Muccini, M. A. Babar, Developing a course on designing software in globally distributed teams, in: International Conference on Global Software Engineering, IEEE Computer Society, Los Alamitos, CA, USA, 2008, pp. 249–253.
15. P. Lago, S. A. Koçak, I. Crnkovic, B. Penzenstadler, Framing sustainability as a property of software quality, Communications of the ACM 58 (10) (2015) 70–78.
16. N. Condori-Fernandez, P. Lago, Characterizing the Contribution of Quality Requirements to Software Sustainability, Journal of Systems and Software-doi:10.1016/j.jss.2017.12.005.
17. N. Condori-Fernandez, Happyness: An emotion-aware QoS assurance framework for enhancing user experience, in: Proceedings of the 39th International Conference on Software Engineering Companion, ICSE-C '17, IEEE Press, Piscataway, NJ, USA, 2017, pp. 235–237.
18. D. Freeney, Usability versus Persuasion in an Application Interface Design, Mälardalen University, 2014.
19. L. O'Brien, P. Merson, L. Bass, Quality attributes for service-oriented architectures, in: Proceedings of the International Workshop on Systems Development in SOA Environments, SDSOA '07, IEEE Computer Society, Washington, DC, USA, 2007, pp. 3–. doi:10.1109/SDSOA.2007.10.
20. S. Kounev, F. Brosig, N. Huber, R. H. Reussner, Towards self-aware performance and resource management in modern service-oriented systems, in: 2010 IEEE International Conference on Services Computing, SCC 2010, Miami, Florida, USA, July 5-10, 2010, 2010, pp. 621–624. doi:10.1109/SCC.2010.94.

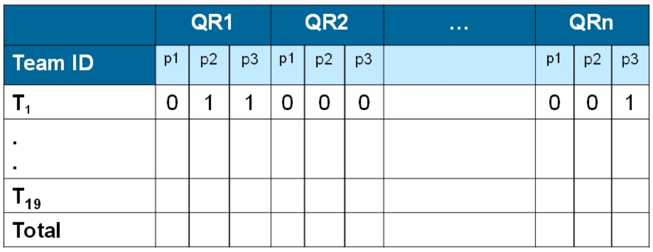
21. G. A. Lewis, E. Morris, S. Simanta, L. Wrage, Common misconceptions about service-oriented architecture, in: Proceedings of the Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, ICCBSS '07, IEEE Computer Society, Washington, DC, USA, 2007, pp. 123–130.
22. D. Ameller, C. Ayala, J. Cabot, X. Franch, How do software architects consider non-functional requirements: An exploratory study, in: 2012 20th IEEE International Requirements Engineering Conference (RE), 2012, pp. 41–50.

Appendix A: Green Strategies

Table 6. Software green strategies for the EV-mobility project. (Gray rows=Domain-generic strategies; rows=Domain-specific strategies) [12]

Green problem type	Green Software Strategy	Code	Size
People awareness	Raise Carbon Footprint awareness	G1	17
Service/Process awareness	Use most efficient charging points	G2	10
Service awareness	Reduce Service's Carbon Footprint	G3	9
People awareness	Eco-friendly driving awareness	G5	8
People/process awareness	Reduce Carbon Footprint in EV	G4	7
People/service awareness	Paperless service	G8	4
Service awareness	Reduce Carbon Footprint caused by DB	G7	4
Service/process awareness	Reduce Carbon Footprint of charging points	G6	4
Process awareness	Virtualization	G9	3
Process awareness	Renewable energy resource	G12	2
Process awareness	Create a green cloud of energy	G13	2
Service/people awareness	Battery Sustainability	G11	2
Service/process awareness	Smart battery charger	G10	2
Service awareness	Optimizing network traffic	G14	1
Service awareness	Green Computing	G15	1
Service awareness	Data monitoring	G16	1

Appendix B: Data collection template



	QR1			QR2			…	QRn		
Team ID	p1	p2	p3	p1	p2	p3		p1	p2	p3
T_1	0	1	1	0	0	0		0	0	1
. .										
T_{19}										
Total										

Fig. 4. Excel template for data collection (p1=design space specification; p2=design space refinement; p3=solution space design)