# A survey on quality attributes in service-based systems

David Ameller[1*], Matthias Galster[2], Paris Avgeriou[3], Xavier Franch[1]
[1]Department of Service Engineering and Information Systems
Universitat Politècnica de Catalunya, Spain
dameller@essi.upc.edu, franch@essi.upc.edu
[2]Department of Computer Science and Software Engineering
University of Canterbury, New Zealand
mgalster@ieee.org
[3]Department of Mathematics and Computer Science
University of Groningen, The Netherlands
paris@cs.rug.nl
* Corresponding author. Tel.: +34 934 137 174; fax: +34 934 139 833

## Abstract

*Context*: Service-based systems have become popular in the software industry. In software engineering, it is widely acknowledged that requirements on quality attributes (e.g., performance, security, reliability) significantly impact the design of software systems.

*Objective*: This study explores the role of quality attributes during the design of service-based systems. We investigate the significance of quality attributes when designing service-based systems and how quality attributes are addressed through design decisions, across application domains, and related to other aspects of software development, e.g., architecture documentation.

*Method*: We conducted a descriptive survey. The survey was done as an online questionnaire targeting practitioners. Furthermore, we included researchers with practical design experience. We obtained 56 valid responses.

*Results*: Most survey participants consider quality attributes and functionality as equally important and treat quality attributes explicitly rather than implicitly. Furthermore, dependability is the most relevant quality attribute in service-based systems; we do not find quality attributes that are particularly important in specific application domains. Most quality attributes are addressed by ad-hoc decisions, rather than established architecture or design patterns or

technologies. Only few decision alternatives are considered when making architectural decisions to address quality attributes.

*Conclusion*: Our results partially confirm anecdotal evidence from current literature, but also strengthen previous claims by providing empirical evidence. Our results point to future research directions (e.g., exploring the impact of decision types on how well quality attributes can be achieved) and implications for practitioners (e.g., training makes a difference to how quality attributes are treated).

## Keywords

Quality attributes, service-based systems, software design, survey, empirical study

## 1. Introduction

Quality attributes (QAs) are characteristics that affect the quality of software systems [25]. Bass et al. differentiate qualities of the system (e.g., availability, modifiability), business qualities (e.g., time to market) and qualities that are about the architecture (e.g., correctness, consistency, conceptual integrity) [9]. Considering QAs throughout software development, and especially during the design stage, is crucial to produce systems that meet their quality requirements (i.e., requirements on quality attributes; for example, "The system shall provide a response time of 1 second to queries that do not involve images" is a quality requirement for quality attribute "performance").

Service-based systems (SBS, also called service-oriented systems) have become popular in the software industry. The underlying architecture paradigm of SBS is Service-Oriented Architecture (SOA). In SBS design (as in software engineering in general), achieving quality and QAs is a top challenge [23]. However, the role of quality attributes in SBS design has not been studied extensively. To address this challenge and to contribute to better understanding QAs of SBS in practice, this article studies the role of QAs in SBS design.

### 1.1. Problem statement and motivation

QAs play a significant role when designing software systems. Some design methodologies, especially the ones for architecture design, treat QAs as drivers [4, 6, 17, 31, 61]. Also, QAs may be used to select specific software components [20] as part of the detailed design, or to evaluate software architectures [7]. According to O'Brien et al. [45], choosing an architecture that satisfies the QAs is vital to the success of a system. Furthermore, QAs shape the architecture design. To create successful SBS designs, it is important to understand how SBS supports different QAs [45]. According to O'Brien et al., this has not been thoroughly researched. Instead, current research has mostly investigated quality metrics for SBS. For example, Sindhgatta et al. focused on metrics for intrinsic qualities (cohesion, coupling, reusability, composability, granularity) but ignore QAs visible to the end user (e.g., performance) [51]. Furthermore, Voelz and Goeb argue that QAs such as interoperability and changeability are supported in SBS but implications of SBS design on performance and security are not well understood [58]. This means, there is currently a lack of understanding the role of QAs in SBSs in industrial practice. Also, to our knowledge there are no thorough empirical studies on the role of QAs in SBS. Thus, we present a survey to understand the role of QAs in the design of SBSs.

### 1.2. Paper goal and contribution

To get an in-depth understanding of the role of QAs in SBS we need to get insights from industrial practice. Therefore, the objective of this study is to collect and report information, from practitioners as well as researchers with practical experience, about what QAs are considered important and how these QAs are addressed during SBS design. Following the four perspectives for describing a goal in the Goal-Question-Metric (GQM) approach (purpose, issue, object, viewpoint) [8], the goal of our study is defined as follows:

- Purpose: Analyze and characterize
- Issue: Role of quality attributes
- Object: In service-based systems design
- Viewpoint: From the viewpoint of practitioners, as well as researchers with practical experience

By addressing this goal, we aim to:

a) Identify the influence of QAs on the design of SBSs (e.g., how QAs are treated compared to functional requirements),

b) Identify how practitioners and researchers with practical experience handle QAs (e.g., what kinds of decisions are used to accommodate QAs).

Detailed research questions are provided in Section 3.1.

The target audience of our study is two-fold: First, we aim at researchers who would like to get insights into how QAs are treated in SBSs design in industry, and who would like to get directions for improvement of current practices and future research. Our results help researchers align their work with industrial needs. Second, we aim at practitioners who would like to find out what QAs are important and how these can be addressed.

Even though some of our results confirm what other research has claimed in an anecdotal manner, our study provides empirical evidence for the importance and treatment of quality attributes in SBS. For example, a design quality model for SOA was proposed by Shim et al. [50]. However, this model has not been empirically validated nor has it been linked to industrial practice.

The work presented in this paper is an extension of our previous work ("The Role of Quality Attributes in Service-based Systems Architecting: A Survey") [2]. The work presented in this paper differs from our previous work in that we a) include additional research questions, b) include more data and more survey responses, and c) perform additional analyses of the data.

### 1.3. Paper structure

In Section 2 we discuss SBS, QAs and architectural decisions. Section 3 discusses the research method and introduces detailed research questions. In Section 4 we present the results of our study. These results are interpreted in Section 5. Section 6 discusses limitations of our study. Section 7 concludes the paper and evaluates the quality of the survey based on a predefined checklist.

## 2. Background

### 2.1. Service-based systems

Service-orientation is a standard-based, technology-independent computing paradigm for distributed systems. As there is no universal definition for service-based design [44], we utilize a broad definition: We consider service-based design as the design of a system which is assembled

from individual services that are invoked using standardized communication models [39, 46]. Service-based design produces service-based systems (SBS). The two important principles of SBS are: a) the identification of services aligned with business drivers, and b) the ability to address multiple execution environments by separating the service description (i.e., interface) from its implementation [15].

Designing SBS requires well documented interfaces for all conceptual services identified during analysis, before constructing services. Service design is based on the logical part of a system (e.g., business logic) and on the physical part of the SOA abstraction layers (e.g., actual software services). The purpose of logical service design is to define single services and to compose services. The physical design part focuses on how to design component implementations that implement services at an acceptable level of granularity, often following component-based development techniques [47].

### 2.2. Quality attributes

For QAs, we adapt the definition proposed by ISO standards for software quality [26-29]: A QA is a measurable physical or abstract property of software product that bears on its ability to satisfy stated and implied needs. According to Balasubramaniam et al. [5], achieving QAs in SBS is critical due to the following reasons:

- Application developers need to be confident that the services (and compositions of them) will meet end user quality requirements.
- Application developers need to understand the cost and risk of achieving quality requirements, given that system QAs often must be traded off or built in [45].
- Application developers require information for selecting between alternate services with similar functional capability.
- Application developers require information about Quality of Service (QoS) to monitor and enforce service level agreements (SLAs).

This is because SBS lack central control and authority, and impose limited end-to-end visibility of services, unpredictable usage patterns and dynamic composition [5]. Desired QoS goals cannot be achieved by tuning the system after the SBS is implemented. Instead, according to Balasubramaniam et al., achieving desired QoS goals in SOA environments will require a life-cycle approach to verification and the incorporation of techniques that focus on architecture, source code, and runtime monitoring [5].

The S-Cube network defines quality as the degree to which a set of characteristics described as QAs fulfills a set of requirements [21]. At least three different views on quality can be distinguished: process quality, product quality and quality in use. Using the definitions in the S-Cube network [21] and similar to [26] we can differentiate quality as follows:

- Process quality is the quality of the production process of a product.
- Product quality refers to the degree to which a product fulfills its requirements.
- Quality in use describes the quality of a product evaluated in specific usage contexts and for specific tasks.

Our work focuses on product quality (design time and runtime). We do not address the quality of the process of developing SBS. Furthermore, quality in use is indirectly addressed as product quality affects the quality in use.

Gu and Lago found more than 50 quality challenges in SBS related to QAs, such as security, reusability, flexibility, interpretability, and performance [23]. Furthermore, O'Brien and et al. discuss how QAs are affected by service-based design [45]. The S-Cube network mentioned before also proposed a quality model for SBS [21]. Our survey design (Section 3.2) uses this quality model as reference in the questionnaire. As S-Cube classifies more than 60 QAs, we do not include the S-Cube quality model in the paper.

### 2.3. Architectural design and design decisions

An architectural decision (AD) is a decision that affects the architecture of a system [24]. A decision addresses QAs and functional requirements through a justified solution. Each decision can address some requirements, but leave others unresolved [12]. ADs might cause the creation of architecture components, assign functionality to existing components, add requirements to a components' expected behavior, or add constraints on the software architecture [12].

Kruchten describes several types of ADs [41]: a) existence decisions relate to the behavior or structure in the system's design or implementation; b) non-existence decisions describe behavior that is excluded from the system; c) property decisions state an enduring, overarching system trait or quality, which might include design guidelines or constraints; d) executive decisions are those driven by external forces, such as financial constraints.

For documenting decisions, it is crucial to decide what to document. A decision should include a problem, motivation, cause, context, potential solutions (alternatives) and decision [30]. Tyree et

al. provide a template to record AD [55]. In our survey design (Section 3.2) we will use this template to derive questions for the survey questionnaire to obtain decisions.

## 3. Research method

Surveys collect qualitative and quantitative information to provide a "snapshot" of the current status related to a phenomenon [60]. This information is used to describe, compare, or explain knowledge, attitudes or behavior [22, 48]. To ensure rigor and repeatability of our study, and to reduce researcher bias while conducting the survey, we designed a survey protocol. The protocol followed the template for surveys proposed for evidence-based software engineering[1]. Furthermore, the study itself followed the six-step survey process proposed by Ciolkowski et al. [14] (Figure 1, including the iterations in the process) and used activities of a survey process as described by Pfleeger and Kitchenham [48]:

1. Survey definition: We determined the goal of the study in terms of a specific objective: Investigating the role of QAs in SBS design (see also Section 1.2).
2. Survey design: We operationalized the survey goal into questions. We defined research questions and designed an online questionnaire as data collection instrument. Furthermore, we validated the data collection instrument. Details of the design are discussed in the following sub-sections.
3. Survey implementation: We operationalized the study design to make the survey executable. This included scheduling the survey and setting up an online infrastructure to collect the data.
4. Survey execution: We collected and processed data.
5. Survey analysis: We interpreted data collected during study execution.
6. Survey packaging: We report the results of the survey in this article.

[1] http://www.dur.ac.uk/ebse/resources/templates/SurveyTemplate.pdf

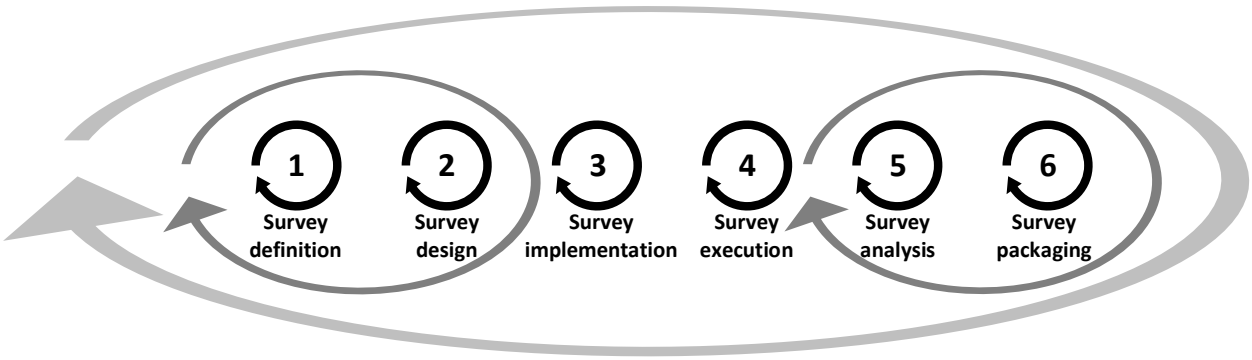


**Figure 1. Survey process**

### 3.1. Research questions

Based on the goal of the study we defined two research questions, each divided into sub-questions with an explicit rationale:

- RQ1: How important are quality attributes in the context of service-based systems design?
  - RQ1.1: How important are quality attributes in comparison to functionality?
  - RQ1.2: Are certain quality attributes more important in particular application domains?
- RQ2: How are quality attributes addressed in the context of service-based systems design?
  - RQ2.1: What kinds of decisions are used to address quality attributes?
  - RQ2.2: What is the impact of these decisions on other quality attributes?
  - RQ2.3: What rationale is behind the architectural decisions?
  - RQ2.4: After the decision was made, were quality attributes satisfied?

RQ1 provides us with an overview of the role of QAs in industrial practice when designing SBS. Here, we first compare how QAs are treated compared to functionality (RQ1.1). Current literature (see Section 5.1) suggests that even though QAs and functionality are equally important, QAs drive the architecture process of software. We are interested in finding out if this is also the case in SBS, or if QAs in SBS are mainly treated as factors that suggest the use of a service-based solution: SBS design has been claimed as an approach to achieve "qualities" such as interoperability, flexibility or reusability [18]. Second, we are interested in finding out if there are QAs that are more common in specific application domains, such as telecommunication or e-government (RQ1.2). This will provide us with information that can be used to provide guidance for architects through the design process by focusing on QAs that are important for a certain domain.

RQ2 aims at obtaining insights into how practitioners address QAs when designing SBS. In detail, we are interested in the kind of decisions that practitioners use (RQ2.1). We used Kruchten's taxonomy of decisions [41]:

- *Property decision*: "A property decision states an enduring, overarching trait or quality of the system. Property decisions can be design rules or guidelines (when expressed positively) or design constraints (when expressed negatively), as some trait that the system will not exhibit", e.g., "all domain-related classes are defined in the domain layer."
- *Existence decision*: "An existence decision states that some element / artifact will positively show up, i.e., will exist in the systems' design or implementation", e.g., "the logical view is organized in 3 layers."
- *Executive decision*: "These are the decisions that do not relate directly to the design elements or their qualities, but are driven more by the business environment (financial), and affect the development process (methodological), the people (education and training), the organization, and to a large extend the choices of technologies and tools", e.g., "the system is developed using J2EE."

Also, we are interested in understanding how decisions made to accommodate one QA would affect other QAs (RQ2.2). This information provides us with impacts of decisions on QAs and associated tradeoffs. This information can be used as reusable architectural knowledge and as starting point for formulating best practices. For example, we could recommend decisions to accommodate certain QAs and at the same time make architects aware of possible negative impacts of such decisions on other QAs. RQ2.3 is about finding out what rationales exist behind decisions. We are interested in why practitioners would choose a decision alternative over other alternatives. RQ2.4 is about finding out if and how QAs are verified after decisions have been made to ensure that they satisfy stakeholders. A significant part of systems and software architecting is the evaluation of systems. Often, QAs are verified during testing, after a system has been implemented. However, this often leads to expensive changes in the architecture if quality requirements cannot be met at a later stage of development.

Answering all these research questions provides us with an empirical foundation of the importance of QAs and how QAs are treated during SBSs design. This will help researchers and practitioners developing new methods for handling QAs in SBS based on solid theories rather

than anecdotal evidence. Furthermore, it will help to identify problems that architects face when making decisions about how to address QAs when designing SBS.

## 3.2. Survey design

### *3.2.1. Form of the survey*

There are three types of surveys: 1) descriptive surveys enable assertions about some population and measure what occurred, rather than why [22]; 2) explanatory surveys make explanatory claims about a population; 3) exploratory surveys are used as a pre-study to a more thorough investigation and in cases where "research looks for patterns, ideas, or hypotheses rather than research that tries to test or confirm hypotheses" [59]. We conducted a descriptive survey [33] to study *how* QAs are treated during SBSs design, rather than *why*.

### *3.2.2. Population, sampling technique, sample size, and participant recruitment and selection*

Our population was the global community of software engineering practitioners, as well as researchers that have practical experience with, and knowledge about, designing SBS. We did not restrict the population with regard to the number of years of practical experience as long as participants had experience from real-world projects.

To find participants we used purposive sampling [16] because all participants needed to have practical experience in SBS design. Other sampling techniques, such as random sampling, may have resulted in many invalid responses as most likely not all randomly picked participants would have had the required background to properly answer the survey questions. To recruit participants, we advertised the survey in around 20 LinkedIn groups (e.g., the Service-Oriented Architecture Special Interest group, the SOA Professionals Worldwide group, the Software Architecture group) and several mailing lists (e.g., ISO / IEC / IEEE 42010 user group, re-online). Furthermore, we advertised through online communities and blogs (e.g., Iniciativa Española de Software y Servicios, ModelingLanguages). We also advertised the survey at premium conferences and workshops (ICSE, ECSA, WICSA, CAiSE) and professional meetings (e.g., meeting of the S-Cube consortium). Finally, we asked individuals in our personal network to participate as well as to spread the survey to other individuals that might be interested in participating in the survey (chain referral sampling) [42]. We did not announce the survey in all venues and through all channels at the same time, but spread the announcements over a longer period of time. The reason is that potential respondents might be members of several online

groups, or have subscriptions for several mailing lists. Sending announcements to different groups and mailing lists at different times acted as reminder to members of the population who had already received the announcement before through other channels.

The sample size was restricted with regard to the responses we could obtain. Our survey was very focused as it required experience with QAs and SBS design in a practical context. Thus, we had limited impact on the sample size. Consequently, we also could not apply power analysis to determine the sample size needed to achieve statistically significant results [36].

### 3.3. Data preparation and collection – questionnaire

We used a self-administered online questionnaire for data collection [22]. The questionnaire was published at a dedicated URL (www.soasurvey.com; note that the survey has been deactivated but we include the list of questions used in the survey in the appendix). The reasons for using an online questionnaire rather than interviews or paper-based questionnaires were as follows:

- Survey participants and researchers did not need to synchronize time and place to collect data (participants filled in the questionnaire independent from researchers).
- It allowed us to collect data from participants from all over the world.
- An electronic questionnaire avoided introducing errors in data when manually entering data into computer systems from paper-based questionnaires.

We acknowledge that interviews would have allowed us to gain in-depth information about QAs in SBS design. Furthermore, interviews would have allowed us to target questions towards interviewees and to explain misunderstandings in questions. However, as we conducted a descriptive survey and aimed at a larger sample, we used a questionnaire.

All questions in the questionnaire referred to one particular project that participants worked in. Based on our experience with previous surveys it is easier for participants from industry to answer questions if they can relate to a particular project [56]. We also acknowledge that results can be biased by the participants' perceptions of what they believe their company do, but since the participants have actively participated in the project referred in the questionnaire we believe that this risk is minimized. Some questions were optional and some mandatory. Structured questions could be answered using Likert-scale or pre-defined answer options [34], unstructured questions allowed numeric answers or free text. For some questions, more than one answer was applicable (e.g., the role of participants). In this case, participants chose the answer that describes

them best. Furthermore, in most questions we allowed participants to provide additional comments to complement their answer. We included different types of questions:

1. Questions related to the profile of participants were used to ensure reliability and documentation of the survey as well as to filter and group participants.
2. We asked questions about the project that participants had in mind when answering the questionnaire. These questions did not address a particular research question but helped compare if project specifics have an impact on how QAs are treated. Before answering project-related questions we included a check question to ensure that participants had design responsibility in the project. Participants could only proceed if this question was answered with "yes".
3. Another set of questions elicited the most relevant QAs in the project. These QAs were specified as scenarios because different participants might have had different understandings of a QA. Scenarios helped "operationalize" QAs. The questions to specify scenarios for QAs were based on the QA scenarios introduced by Bass et al. [9]. Here, we did not use all parts of a QA scenario but only stimulus, artifact (system) and response / measure. These questions aim at answering RQ1.
4. We had questions that describe ADs made to accommodate QAs, and the relationship of these decisions to other QAs. These questions aimed at answering RQ2. They were derived from the template to describe ADs as proposed in [55] and used in [24] (i.e., this template requires that decisions were described in terms of the issue they address, alternatives considered for that decision, the rationale for the decision, etc.). Referring to this template, the "issue" to be addressed by a decision would be the QA elicited before. Participants were asked to provide at least one decision but could provide up to three. We also asked participants for comments on QAs or QAs that may have not been listed in the S-Cube model.
5. Finally, we asked three open questions to get further details about the context and the respondent: "In your projects, do you usually document information about design decisions?", "What problems do you think occur when you try to satisfy quality attributes in the context of service-based systems (if any)?", and "Upon reflection of answering the questions, is there anything you can add and that you feel is relevant in the context of this questionnaire?".

The questionnaire was made available in two iterations from May 2011 to December 2012. The first iteration ended in September 2011 with 31 responses; the second iteration ended in December 2012 with 25 additional responses. On average, the questionnaire took around 20 minutes to be completed. In total, 235 potential respondents started the survey, but only 56 completed it.

### 3.4. Data analysis

To ensure the quality of the data obtained from the questionnaire, we applied sanity checks. These sanity checks aimed at finding obvious errors in data. Sanity checks also helped ensure that responses expressed what we interpret as decisions or QAs. Furthermore, the quality of input data was ensured by restricting possible input values through predefined options for some questions (see appendix). Table 1 shows the mapping between the research questions and the questions in the questionnaire as outlined in the appendix. Note that not all questions from the questionnaire were used for data analysis.

Table 1: Mapping of research questions to questions in the questionnaire

| Research question | Questions in questionnaire |
|---|---|
| RQ1 | |
| RQ1.1 | PS7, PS8 |
| RQ1.2 | PS2, QA1, QA2, QA3, QA4, QA5 |
| RQ2 | |
| RQ2.1 | QA1, AD1, AD2, AD4 |
| RQ2.2 | AD5 |
| RQ2.3 | AD3 |
| RQ2.4 | QA1, QA5 |

We applied descriptive statistics, qualitative analysis and content analysis to analyze the data [37]. In particular, we analyzed the variables using frequency analysis and correlation analysis with cross-tabulation and Fisher's exact test. Questions which resulted in free text were coded [43] and underwent content analysis [40]. Each content analysis involved three of the authors (two plus one to resolve the conflicts), in varying combinations. For example, to identify decision types, the researchers independently judged each decision and a final judgment was achieved through consolidation of the different judgments. We determined consistency among the involved researches using Cohen's kappa statistic.

### 3.5. Protocol review

The survey was validated as follows [35]:

1. The protocol was reviewed by external reviewers. The external reviewers were researchers experienced in empirical studies, and in particular in the execution of surveys.
2. The survey procedure was piloted.
    a. Researchers other than the authors piloted the data collection instrument and data analysis. Problems were identified and revisions made accordingly [35].
    b. An initial test of the questionnaire was performed with participants from the target population. The questionnaire was redesigned accordingly. Furthermore, as suggested by [56], participants were asked to explain questions to ensure that questions were well understood.

# 4. Results

## 4.1. Demographic data of respondents

The 56 participants came from different organizations and/or different projects (as mentioned above, participants referred to one particular project when answering the questionnaire). Thus, our results not only reflect how QAs are treated by individuals, but also how organizations and projects treat QAs. In this section we provide information about the location, distribution, practical experience, educational background and the role of participants, as well as the size of participants' organizations.

**Location of participants:** Participants were from countries all over the world. The countries with the most participation were Brazil (9, 16.1%), Spain (7, 12.5%) and Argentina and Germany (both 5, 8.9%), but we had participants from 6 different continents: Europe (22, 39.3%), South America (15, 26.8%), North America (8, 14.3%), Asia (6, 10.7%), Australia (4, 7.1%), and Africa (1, 1.8%).

**Distribution of participants:**

More than half of the participants (58.9%) were both researchers and practitioners (e.g., industrial researchers or research consultants). See

Table 2 for details. Note that researchers who are not practitioners at the same time (17.9%) still have practical design experience.

Table 2. Distribution of practitioners and researchers with practical design experience

| | | **Practitioner** | |
|---|---|---|---|
| | | No | Yes |
| **Researcher** | No | 0 (0%) | 13 (23.2%) |
| | Yes | 10 (17.9%) | 33 (58.9%) |

**Practical experience of participants:**

- Practitioners had, on average, 11.17 years of experience in SBS design (standard deviation 8.14).
- Researchers with practical experience had, on average, 4.14 years of practical SBS design experience (standard deviation 6.52).

**Educational background of participants:** The academic background of participants was mostly a degree in Computer Science with at least a Bachelor degree (see Figure 2). The majority of our participants (64.3%) had received training in service-oriented computing.

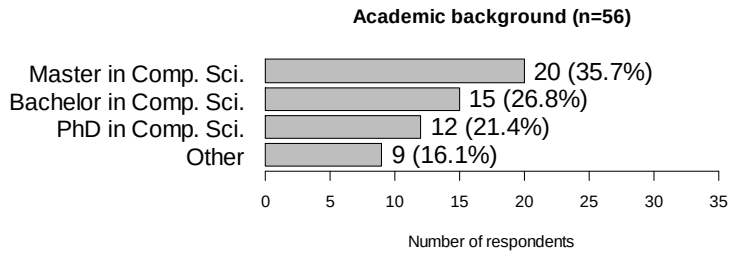


Figure 2. Academic background and practitioner role

**Role of participants on their organization.** As can be seen in Figure 3, the majority of the 46 practitioners (including practitioners that were also researchers) were architects or designers, but there were also developers, project managers and a few who had other responsibilities (e.g., software analyst). We note that even if some practitioners were not architects or designers as

their main role, all of them indicated (by answering a dedicated question) that they had architecting responsibilities for the project on which they based their answers.

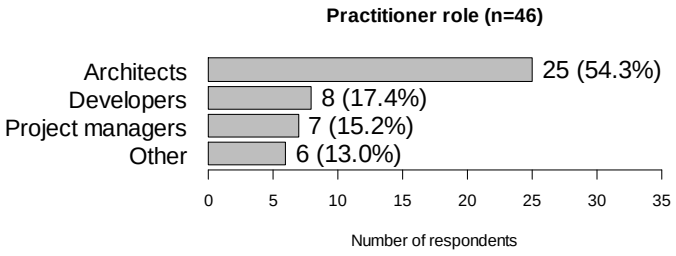


Figure 3. Practitioner role

**Size of participants' organization:** Figure 4 shows that the majority of responses came from participants from large companies. Note that Figure 4 only includes responses from practitioners (including practitioners that were also researchers), but not from researchers with practical design experience that were not practitioners at the same time.

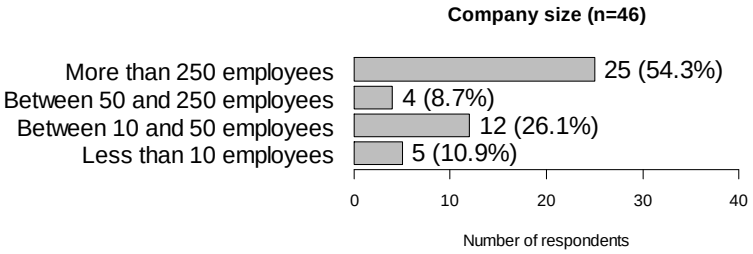


Figure 4. Company size

**Types of projects mentioned by the participants:** The participants provided information for a high variety of projects types. The range goes from administration support applications for government and academia (e.g., "*information system for the [country name omitted] employment department"* or "*student admission system, that use web service to register students*") to health and bioinformatics related systems (e.g., "*B2B between insurance institutions and healthcare organization*" or "*a service-based architecture for federating multiple bioinformatics databases*"), also including smaller projects that were centered in service-based tools (e.g., "*a tool for web service monitoring and SLA checking*"). Since the quality of the answers in many cases did not allow a proper interpretation we skipped the content analysis of this question.

## 4.2. RQ1: How important are quality attributes in the context of service-based systems design?

### *4.2.1. RQ1.1: How important are quality attributes in comparison to functionality?*

The results are shown in Figure 5 (5 respondents did not provide an answer). The majority of respondents indicated that functionality and QAs were considered equally important across most projects that participants related to when answering the questionnaire. Furthermore, in most projects QAs were made explicit, but still a significant amount of respondents stated that QAs were treated implicitly.

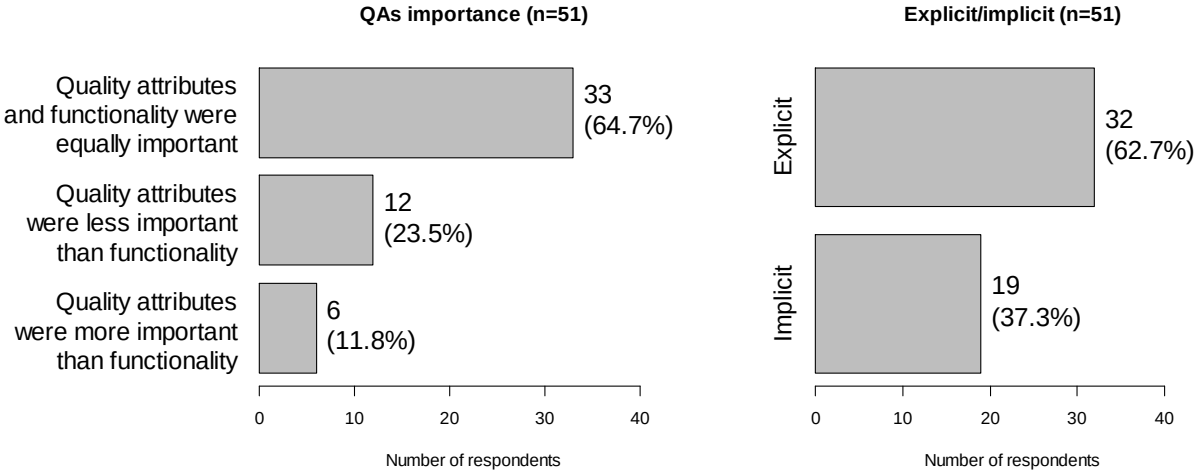


Figure 5. Role of QAs

To study the dependency between the importance of QAs and the implicit or explicit nature of QAs, we created a cross-tabulation (see Table 3). Fisher's exact test led to $p < 0.001$ which means that there is a statistically significant relationship between the importance of QAs and their implicit or explicit nature. It means that there is a high probability that projects which treat functionality and quality equally important also treat QAs explicitly. On the other hand, there is a very low probability that QAs would be treated equally important to functionality when QAs were considered implicitly

We did not find any statistically significant correlation (based on cross-tabulation and Fisher's exact test) between the background of participants as stated in Section 4.1 (years of experience, educational background and role of participants, size of organization / project) and whether QAs

are treated more or less important than functionality. Similarly, we did not find any statistically significant correlation between the background of participants and whether or not QAs are considered implicitly or explicitly.

Table 3. Cross-tabulation of the importance of QAs and their implicit or explicit nature

| | **QA explicit** | **QA implicit** | **Total** |
|---|---|---|---|
| Quality attributes and functionality were equally important | 24 (47.1%) | 9 (17.6%) | 33 (64.7%) |
| Quality attributes were less important than functionality | 2 (3.9%) | 10 (19.6%) | 12 (23.5%) |
| Quality attributes were more important than functionality | 6 (11.8%) | 0 (0.0%) | 6 (11.8%) |
| **Total** | 32 (62.7%) | 19 (37.3%) | 51 (100%) |

*4.2.2. RQ1.2: Are certain quality attributes more important in particular application domains?*

We mapped all QAs stated by participants to QAs for SBSs as defined by the S-Cube quality model [21]. This was done through content analysis of quality attribute scenarios where three researchers categorized each QA (Cohen's kappa: 0.688, $p < 0.001$). Figure 6 shows the frequency distribution of QAs. Four participants did not specify a valid QA. In Figure 6 we group data-related QAs based on the S-Cube quality model (data reliability, completeness, accuracy, integrity, validity). As can be seen, dependability and performance are the most frequently addressed QAs.

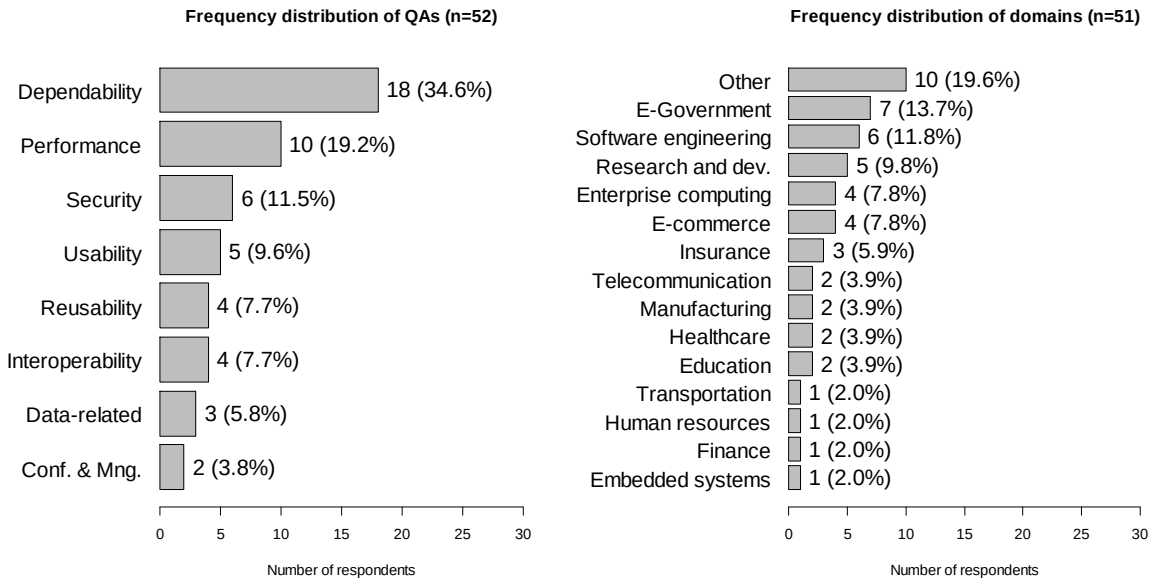


Figure 6. Frequency distribution of QAs and domains

Figure 6 also shows the frequency distribution of project domains (as described in Section 3.3, participants answered according to their experiences in one specific project). The category "Other" includes domains such as aerospace, real state, social networking, bioinformatics, etc.

Fisher's exact test did not reveal any relationship between QAs and domains ($p = 0.635$). Also, cross-tabbing QAs and domains did not show a QA that would be addressed more than twice by a domain (with the only exception of domain Government and the QA Performance with three occurrences). This means, we could not identify any QA that would be particularly relevant for a certain domain.

## 4.3. RQ2: How are quality attributes addressed in the context of service-based systems design?

### *4.3.1. RQ2.1: What kinds of decisions are used to address quality attributes?*

We asked the participants about the most important AD they made related to the QA they selected as most important. We classified these decisions into decision types based on Kruchten's taxonomy [41] (see Section 3.1, Cohen's kappa: 0.484, $p < 0.001$). Furthermore, we classified decisions into the following three categories of decisions that emerged during data analysis (Cohen's kappa: 0.427, $p < 0.001$):

- *Ad-hoc*: Solution that is specific to a concrete problem of the project (e.g., the architect decides to create a separate service to store sensitive information about the users to improve the security of the system).
- *Pattern*: Reusable and widely-known architectural solution (e.g., the decision to use of the Model-View-Controller (MVC) pattern [13] for structuring the user interaction).
- *Technology*: A piece of implemented software that fulfills some required functionality (e.g., the decision to use PostgreSQL instead of other DBMS because the project only uses OSS licenses).

As Figure 7 shows that property decisions are used (and considered) most (Figure 7 not only shows the actual decision made but also alternative decisions considered for that decision). Furthermore, most of the decisions were classified as ad-hoc or pattern, and only few decisions are technology-related decisions. Note that one decision was not classified because the participant did not provide a description for it. The fact that ad-hoc decisions dominate could mean that SBS is still not a mature area and architects need to come up with new solutions frequently (rather than reusing existing and established architecture / design patterns and technologies).

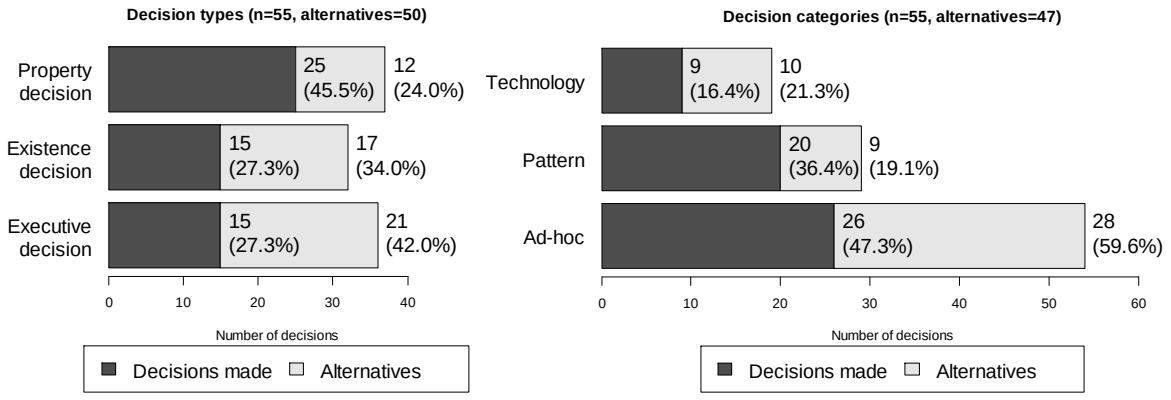


Figure 7. Classification of decisions

Half of the participants indicated that the mentioned decision had at least one alternative (28, 50.0%). Furthermore, a clear majority declared that decisions related to QAs were documented explicitly (39, 86.7%).

*Other observations:*

a) There is a correlation between having the QA explicitly as requirements of the project and documenting decisions (Fisher's exact test: $p$=0.009). All participants that treated QA explicitly documented the decisions and also, all participants that did not document decisions treated QA implicitly.
b) There is a correlation between the number of alternatives considered to accommodate QAs and the fact that a participant received some SOA training in the past (Fisher's exact test: $p$=0.164). 75.0% of participants that provided at least 1 alternative had SOA training. 65.0% of participants without SOA training did not provide any alternative.
c) We tried to find correlations between the decision classifications and the QAs mentioned by the participants, but in this case the results are not significant enough. We obtained $p$=0.320 and $p$=0.440 for decision types and categories respectively.

#### *4.3.2. RQ2.2: What is the impact of these decisions on other quality attributes?*

We asked participants which attributes of the S-Cube quality model [21] had been affected by the mentioned decision. In particular, we first asked which of the nine top-level S-Cube QAs (performance, security, data-related, cost, dependability, configuration and management, usability, quality in use, and standard compliance related) were affected by the decision. The impact could be from very negative to very positive (on a Likert scale). In Figure 8 we show the number of decisions that impacted each QA. Few correlations were found between decisions types and their impact in quality attributes. Remarkably, 73.3% of existence decisions impacted in data quality attribute ($p$=0.035), 66.7% of executive decisions impacted in configuration and management ($p < 0.001$), and 80.0% of executive decisions impacted in performance ($p$=0.045). Note that these correlations do not differentiate between positive and negative impact. Decision categories did not show any significant correlation.

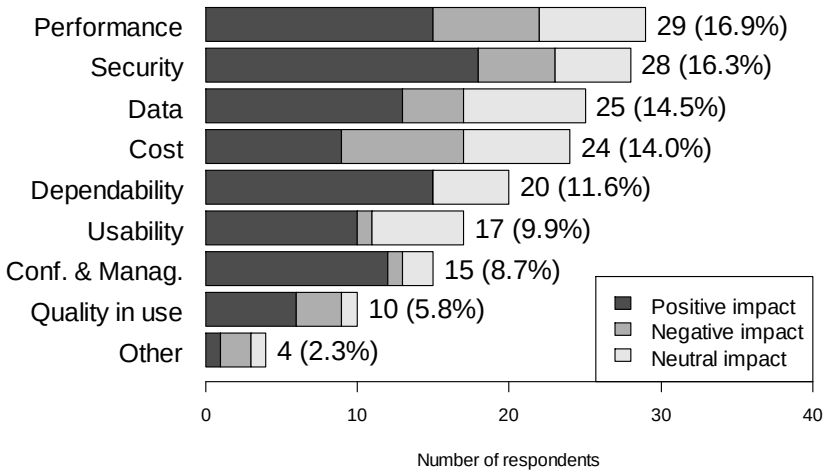


Figure 8. Impact of decisions in the software quality

On average, each decision that was made in order to accommodate one QA impacted three other QA (mean = 3.05, and standard deviation = 1.656).

### *4.3.3. RQ2.3: What rationale is behind the architectural decisions?*

We asked participants why they chose a decision over its alternatives. We obtained 35 responses to this question. The results are shown in Figure 9 were obtained using content analysis (Cohen's kappa: 0.839, $p < 0.001$). From these responses, 16 (45.7%) stated the positive impact on certain QAs as the primary reason, e.g., "[The decision showed] better performance measurement results". Another group of 10 participants (28.6%) used their experience as the principal reason, e.g., "[The decision was] based on previous experience and consultancy". "Business" includes reasons related to the business goals of the project, e.g., "[The decisions was chosen because] we had invested too much in the first project and we wanted some return on that investment".

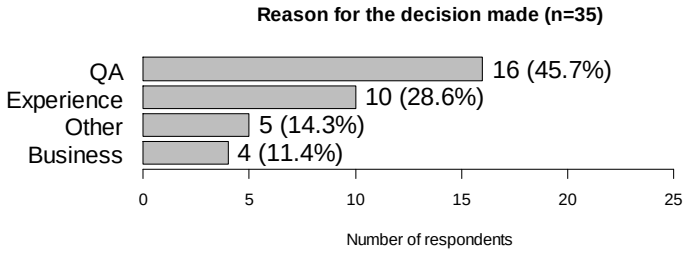


Figure 9. Reasons given for the decisions made

Interestingly, 60.0% of decisions made based on experience were from architects without SOC training, while 83.3% of decisions made based on QA were from architects that had some kind of SOC training (Fisher's exact test: $p$=0.104).

*4.3.4. RQ2.4: After the decision was made, were quality attributes satisfied?*

We asked participants if they validated the mentioned QA, and how (we obtained 42 responses to this question). We classified the responses in three evaluation methods. These evaluation methods emerged during data analysis (Cohen's kappa: 0.711, $p < 0.001$):

- *Testing*: Executing the system using different test cases (depending on the QA), for example, stress testing to validate reliability. Here, we refer explicitly to tests that are planned and executed during system development.
- *Measuring*: Collection of measures that result from applying several metrics (depending on the QA), for example, measuring the response time to validate the performance. We refer explicitly to monitoring activities done once the system is deployed.
- *Observation*: Gathering feedback obtained from the users of the software product. The feedback could come from a beta testing phase or from the final customer. The difference with monitoring is that this method is not done automatically and is normally triggered by the unsatisfied users (e.g., a report of the issues detected in the system to validate quality in use).

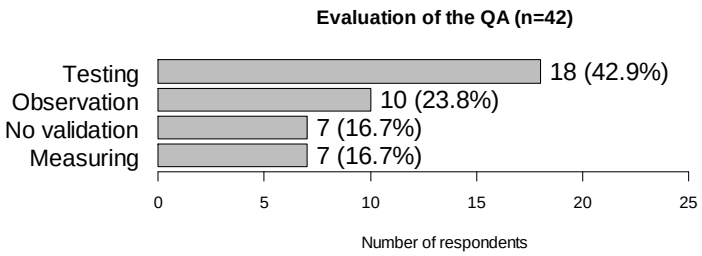


Figure 10. Evaluation of the QA

Figure 10 shows how Testing was the favorite option. In most cases, one of the three evaluation methods was used (83.3%), just 16.7% of the responses indicated that no evaluation was performed. Also, we found that 50% of participants that did not document the decisions were participants that did not perform evaluation. Interestingly, 84.7% of participants that documented the decisions also performed some kind of evaluation (Fisher's exact test: $p$=0.083). This is an indicator that documentation implies that more systematic and structured development approaches are in place, which also include more thorough evaluation.

# 5. Discussion of results

## 5.1. Relation to existing literature

### *5.1.1. Important quality attributes*

Quality requirements in industry were studied by Svensson et al. [53] who found that usability and performance are currently the most important quality requirements in industry. On the other hand, reusability and maintainability seem to be the least important QAs. Even though these quality requirements apply to general software development rather than SBS, our study confirmed that performance and usability are also important in SBS design.

A study in the embedded systems industry [52] studied how quality requirements are handled in practice. The study involved interviews with five product managers and five project leaders from five companies. Even though the research questions and the domain of this study were different than our research questions, the study found that usability and performance are the most important quality aspects. In contrast, our study found dependability and performance as the most important QA. The difference in the importance of usability could be due to the nature of embedded systems versus SBS: in embedded systems user interfaces receive great attention and can determine the acceptance of a system by end users; in SBS the composition of a system by third-party services imposes considerable challenges on dependability and performance.

In [10] the authors conducted a survey to evaluate a catalogue of non-functional properties for SOA, from the perspective of service consumers. The design and goal of this survey differed to ours: The survey required participants to evaluate a catalogue with 19 non-functional properties. These QA were prescribed, rather than elicited from participants as in our study. The study found that security was prioritized as being absolutely essential in a quality model for SOA. However, our study showed that security was only a concern in six projects. Also, from the eight QA (Figure 6) found in our study, only three (performance, security, usability) are also included in the list of non-functional properties for SOA proposed as a result of the survey presented in [10]. Interoperability, a QA found in our study was considered as relevant for service providers, not for service consumers in [10]. Interestingly, reusability or dependability, two main features of SBS were not found to be relevant non-functional characteristic in SOA in [10].

### *5.1.2. Relevance of quality attributes*

Literature argues that QAs are important and are a major challenge when designing SBS [23]. This is not directly confirmed in our study but only hinted as 65.1% of the participants indicated that QAs were made explicit. The fact that QAs are made explicit could be an indicator that special attention is paid to QA because they pose a major challenge.

General literature about software architecting and design claim that quality requirements drive the architecture [9]. Our study cannot confirm or reject this theory for the domain of SBS. We found that QAs were rarely indicated to be more important than functionality. However, stating that QAs drive the architecture is different from stating that QAs are more important than functionality. In fact, both (QAs and functionality) are important, but design methods (e.g., [3]) usually start with analyzing QAs as key drivers which then help select architecture solutions such as patterns and tactics. This could be an indicator that QAs were treated as architectural drivers for high-level architectural decisions. It also indicates that using a service-based solution is not only a technology-driven decision but has sound rationale based on QAs.

A study on the reasoning process of professional software architects revealed that most architects consider functionality as important or very important, but quality requirements as clearly more important than functional requirements [56]. This study acknowledges that one of the most important things in architectural decision making is to treat both functional and non-functional requirements as first-class concerns [56]. Our study confirms this for SBS since the majority of participants treated QAs and functionality as equally important. The authors of this study also mention that almost all of their participants think about alternatives for their decisions [56]. On the other hand, our results indicate that only half of the participants thought about at least one alternative decision.

In another study, van Heesch and Avgeriou studied the reasoning process of junior architects [57]. In their study, more than 80% of participants indicated that quality requirements play a prominent role during design. A similar result can be found in our study with practitioners in the context of SBSs as only 12% of our participants indicated that QAs were less important than functionality. On the other hand, our results show that only 23% of participants treated QAs as more important than functionality (i.e., QAs did not play the most prominent role for 23% of the participants).

Non-functional requirements as seen by architects in general were studied by Poort et al. [49]. The study found that as long as architects are aware of non-functional requirements, they do not adversely affect project success. This is in line with our results that most participants consider quality attributes explicitly and at least equally important as functionality. Furthermore, Poort et al. found that modifiability requires special attention if it is business-critical. In contrast, we did not find any indicator that modifiability could threaten project success. This may be due to the fact that SBS are considered highly flexible and reconfigurable systems by definition, so the focus is on run-time adaptation.

### 5.2. Implications for researchers

Even though QAs for SBSs have been proposed in previous research (e.g., in [45]) there has not yet been any empirical evidence for the importance of these QAs and how they are addressed in industry. Consequently, our study is one of the few, to the best of our knowledge, empirical studies on QAs for SBSs. Thus, future research might aim at identifying in detail how QAs are handled in industrial practice (e.g., through case studies rather than broad surveys) to confirm or refute our findings.

A promising direction for future research on QAs in SBS is to focus on dependability and performance, as these are the two most mentioned QAs. On the other hand, as we could not identify any QA that is specific for a particular domain (see Section 4.2.2 for identified domains), we can conclude that this could mean that each domain has several QAs of interest and the distribution between them is random. The same applies when comparing the architectural decisions with QA and software domains. However, these results should be validated in further studies to find out if decisions, QA, and domains are really unrelated, because in all cases, significance was insufficient to make any claim about dependencies.

As previously studied, many architecture-related problems are rooted in requirements [19]. These problems are often about quality requirements and quality drivers (e.g., identification of quality drivers, understanding and modeling of quality requirements). Our study complements these findings for SBSs and highlights the need for further research to identify more specific quality-related problems. For example, certain types of QAs (e.g., performance, security) could be targeted to identify if and how problems related to these QAs during design are rooted in requirements for SBSs, or if we need better design methodologies for SBS, or we need better alignment between requirements engineering and architecting. Also, new methods for handling

QAs in industry could be developed. As there is no QA specific to one industry, we might benefit from generic solutions that could fit SBSs design in many industries. For example, research could identify tactics for SBS design or SOA-specific analysis methods based on current frameworks for reasoning about QAs [4].

While most participants (86.7%) agreed that they document architectural decisions, no comment was made to what extent decisions were documented. In literature there are diverse templates to document architectural decisions, such as [55] that could help improve documentation. However, probably none of them is used by practitioners [54]. Thus, it could be interesting to extend our study to get more insights into what is documented as architectural decision in practice. Also, it seems that documentation of decisions is linked to explicit QAs, because we found that when a project had explicit QAs it also had decisions explicitly in the documentation. Furthermore, when the decisions were documented, the QAs were in most cases also validated in some way. Thus, there are two trends: participants that document both decisions and QAs and then validate the software, and participants that do not document and do not validate. There do not seem to be many cases in the middle, i.e. partially documenting and partially validating.

### 5.3. Implications for practitioners

The majority of participants with SOC training treated QAs and functionality equally important. Furthermore, even the majority of untrained participants treated QAs and functionality as equally important. Consequently, with regard to training, our results are inconclusive: receiving training in SOC does not affect whether practitioners treat QAs as more important than functionality. The only difference we observed is that trained participants consider quality explicitly more than untrained participants.

We have found that larger companies tend to treat QAs explicitly and equally important to functionality. As larger companies usually have more mature software development processes in place, we can suggest that explicit QAs and equal importance of QAs and functionality is a best practice for the design of SBS.

When it comes to making architectural decisions, having specific training in SOC makes a difference: participants with training provided more alternatives to the decisions than participants without training. This means that they consider a wider spectrum of possibilities to design the architecture. We also found that participants with training in SOC make decisions using some criteria such as QA or business goals, while participants without training normally use their

experience or gut feeling. All these facts together seem to point out that participants with some kind of SOA training make decisions of better quality. This is because decisions are made in a systematic manner, with a clear reasoning behind decisions. However, for further validation we would need to conduct further studies and relate our results to the quality of the actual outcome of a design process.

One of the most impacted QA once the architectural decision is made is cost. Cost is the QA which is most negatively affected. As only a small number of decisions had no impact on cost, it seems that most decisions increase or decrease of the cost of producing a system. It is important to note however that cost was never mentioned as the motivating QA for making a certain decision.

Furthermore, there is no decision that had a negative impact on dependability or usability, which could mean that causing a negative impact on any of these two QA is considered a very bad practice in SBS. We can also notice that performance, security and dependability are the three QA which are mostly positively affected.

## 6. Validity

There might have been confounding variables and other sources that could bias our results [38]. When designing surveys, variables are difficult to control [14], in particular when using online questionnaires. To control variables, exclusion or randomization can be applied [56]. Exclusion means that participants who are not sufficiently experienced were excluded from the study. We ensured this by having a check question that only allowed participants with design responsibility in a project to proceed with the questionnaire. Randomization means that we used a sampling technique which leads to random participants. Furthermore, validity is subject to ambiguous and poorly phrased questions. To mitigate this risk, we piloted the data collection instrument in multiple iterations until potential respondents understood our questions and intentions. Another limitation is that participants might not have answered truthfully to the questions [56]. To address this problem, we made participation voluntary and anonymous. Furthermore, participants spent personal time on answering the questionnaire. We can therefore assume that those who volunteered to spend time have no reason to be dishonest [56]. Also, participants might not have

had the same understanding as we required them (e.g., what is a decision, what is a QA); we tried to mitigate this through sanity checks.

External validity is concerned with the problem of generalizing the results to the target population. We assume that our results are applicable to a population that meets the sampling criteria of our survey (i.e., practitioners with design responsibility in SBSs). However, answers are not just influenced by the understanding of participants, but also the characteristics of companies and software projects in which participants worked. We provided a brief discussion of how company and project size affected the results. This helped us understand the influence of the domain on the results [56]. Furthermore, we only had a limited number of participants. However, this is due to the fact that our survey targeted a very specific population and required participants with knowledge about QAs, experience with designing SBSs, and involvement in a real project. The participation of this study (56 participants) compared to other empirical studies in software architecture is slightly above (e.g., 11 software companies [53], 53 industrial software architects [56], 22 students [57]). It is worth to remark that the mentioned examples were not limited to SBS.

## 7. Conclusions

This paper presented an empirical study to investigate the role of QAs in SBS design. We collected data in a survey with participants from industry, and from different organizations and project domains. We interpreted results in terms of implications for both researchers and practitioners. Furthermore, we compared our results with previous research on QAs and architectural decision making.

With regard to RQ1 we found that QAs are mostly considered as important as functionality, and that QAs are often made explicit. The most important QAs in SBS are dependability and performance. We did not find QAs that would be important specifically in a particular domain. As suggested in [53], quality requirements are poorly addressed in industrial practice; our position is that future studies should focus on the most important QAs (i.e., dependability and performance) rather than finding a blanket solution for QAs in general.

With regard to RQ2 we found that most decisions made to accommodate QAs are property decisions and ad-hoc solutions. Furthermore, we found trade-offs between QAs, when decisions

made to accommodate one QA had a negative impact on other QAs. This means, when considering architectural decisions, one should always reflect on their impact on QAs. Also, the positive impact of a decision was the primary driver for taking a certain decision. Regarding evaluation of QAs, testing during the system development was the most used approach, and few cases mentioned the use of monitoring techniques once a system is produced and deployed.

Besides the future work discussed in Section 5, future work should focus on validating and customizing existing quality models specifically for SBS. The S-Cube quality model used in our research provides a foundation but can be further evaluated in empirical studies. As it has been found with other quality models, theoretical models without empirical validation might not be applicable as they are ambiguous and incomplete [1].

Some of our findings may not be surprising as previous research has already reported important quality attributes in SBS. However, our research differs in that we a) provide empirical evidence about the importance of certain quality attributes in practice, and b) provide empirical evidence of how quality attributes *are* treated in practice, rather than proposing a new methodology for how quality attributes *should* be treated.

Based on Bennett et al. and Kitchenham et al., we compiled a checklist to evaluate our survey [11, 32]. In the following we show what items of this checklist are covered by what section of our paper. We also explain what items on the checklist are not met by our study, and why. See Table 4 and the corresponding footnotes.

Table 4: Checklist to evaluate our survey

| | | |
|---|---|---|
| Background | Justification of research method | Section 3 |
| | Background literature review | Section 2 |
| | Explicit research questions | Section 3.1 |
| | Clear study objectives | Section 1.2 |
| Research method | Description of data analysis methods | Section 3.4 |
| | Discussion on questionnaire administration | Section 3.2.3 |
| | Description of data collection | Section 3.2.2 |
| | Description of dates of data collection | Section 3.2.3 |
| | Number and types of contact[1] | Section 3.2.2 |
| | Sufficient description of method (for replication)[2] | Appendix |
| | Evidence for reliability and validity | Section 6 |
| | Discussion of methods for verifying data entry | Sections 3.4 and 3.5 |
| Sample selection | Calculation of sample size[3] | Section 3.2.2 and 4.1 |
| | Description of population and sample | Section 3.2.2 and 6 |
| Research tool | n/a (except for online survey system) | Section 3.3 |

| | | |
|---|---|---|
| Results | Presentation of results of the research | Section 4 |
| | Did the results address study objectives[4] | Section 3.4 |
| | Description that results are based on partial sample[5] | Section 4 |
| | Description about the generalizability of results | Section 6 |
| Interpretation and discussion | Interpretation and discussion of findings | Section 5 and 6 |
| | Conclusions and recommendations | Section 5 |
| | Study limitations discussed | Section 6 |
| Ethics and disclosure | n/a[6] | n/a |

[1] We had no personal contact with all participants but only contacted around 50 potential participants in person. Most participants were contacted through online postings and advertising at conferences. Thus, we have no information about the number of contacts we made in total.

[2] We believe that this criterion is met as we provided details on our sample as well as the questionnaire for data collection in the appendix.

[3] We did not calculate the sample size because we do not know the size of the population.

[4] We mapped survey questions to research questions. Research questions were obtained from the study objective.

[5] We highlighted results that are only applicable to a subset of participants (e.g., participants with training in service-based systems)

[6] Due to the nature of our study and the context in which it was conducted, it was not necessary to obtain consent from participants. As participation was voluntarily, filling in the questionnaire expressed implicit consent. Also, it was not necessary to obtain ethics approval for the study and to ensure fair treatment of human subjects. Finally, any sponsorship of the study did not have any impact on the results (sponsorship of the study is made explicit in the acknowledgements).

## 8. Acknowledgments

We would like to thank the reviewers of SQJ journal for their valuable comments. This research has been partially sponsored by the Spanish project TIN2010-19130-c02-01 and NWO SaS-LeG, contract no. 638.000.000.07N07.

# Appendix

Table 5. Questions about the profile of participants

| ID | Question | Scale |
|---|---|---|
| P1 | What country do you reside in? | List of countries |
| P2 | What is your educational background (highest degree obtained so far)? | BSc / MSc / PhD / Other |
| P3 | Have you ever received any training related to service-oriented computing? | Yes / No |
| P4 | Do you have experience in academic research? | Yes / No |
| P5 | How many years have you spent on research related to service-based systems? | Integer > 0 |
| P6 | Do you have experience in IT industry? | Yes / No |
| P7 | How many years of experience do you have in IT industry? | Integer > 0 |
| P8 | What is your main role in your company? | Project manager / Architect, designer / Developer / Other |
| P9 | What is the size of your company (number of employees)? | < 10 / 10 – 50 / 50 – 250 / > 250 |
| P10 | What domain is your company in? | List of domains / Other |
| P11 | How many years have you spent on doing work related to service-oriented computing? | Integer > 0 |

Table 6. Project-specific questions

| ID | Question | Scale |
|---|---|---|
| CHK | Did you have design responsibility in the project? | Yes / No |
| PS1 | Please provide the following metrics related to the size of the project | Person months / SLOC |
| PS2 | What is the domain of the project you are thinking about? | List of domains / Other |
| PS3 | Please provide a brief description of the project. | Free text |
| PS4 | What type of software was developed in the project? | Single services(s) / Service-based system / Hybrid system / Other |
| PS5 | Why was service-orientation chosen for the given project? | Strategic decision of company / Certain quality attributes suggested the use of a service-based solution / We wanted to experiment with services / Because of other concerns / I don't know |
| PS6 | Select the sentence that describes the use of external services in your project best. | Project did not use external services but only services developed in-house / Project used external services from trusted sources / Search for external services was done, not considering a specific source / Software used self-adapting mechanism to discover new services when necessary |
| PS7 | Compared to functionality, how important were quality attributes when designing the system of the project you are thinking about? | Quality attributes were not important / Quality attributes were less important than functionality / Functionality and quality attributes were equally important / Quality attributes were more important than functionality / I don't know |
| PS8 | Were quality attributes considered implicitly or explicitly? | Implicitly (quality attributes existed but were not considered as particular requirements) / Explicitly (quality attributes were made explicit in requirements) |

Table 7: Questions to elicit the most important project-specific quality attribute as a scenario

| ID | Question | Scale |
|---|---|---|
| QA1 | What was the most important quality attribute in your project? | Free text |
| QA2 | What part of the system was affected most by this quality attribute? | Free text |
| QA3 | What situations or events had to happen to make this quality attribute evident or visible to the end users or other stakeholders? | Free text |
| QA4 | What restrictions or goals were imposed on this quality attribute? | Free text |
| QA5 | How did you measure or test the satisfaction of this quality attribute (include quantitative information if applicable)? | Free text |

Table 8. Questions to describe architectural decisions related to a quality attribute

| ID | Question | Scale |
|---|---|---|
| AD1 | What was the most important design decision that you made in the project that is related to this quality attribute? | Free text |
| AD2 | What other alternatives did you consider for this decision? | Free text |
| AD3 | What is the reason why you selected this decision? Also, why did you reject the other alternatives? | Free text |
| AD4 | Was this decision related or forced by previous decisions? | Free text |
| AD5 | What other quality attributes were affected (negatively or positively) by this decision, and how? | List of quality attributes from S-Cube and scale from very negative, negative, positive to very positive |

## References


[1] Al-Kilidar, H., Cox, K., Kitchenham, B.: The Use and Usefulness of the ISO / IEC 9126 Quality Standard. International Symposium on Empirical Software Engineering, pp. 126-132. IEEE Computer Society, Noosa Heads, Australia (2005)

[2] Ameller, D., Galster, M., Avgeriou, P., Franch, X.: The Role of Quality Attributes in Service-based Systems Design. 7th European Conference on Software Architecture (ECSA), pp. 200-207. Springer Verlag, Montpellier, France (2013)

[3] Bachmann, F., Bass, L.: Introduction to the Attribute Driven Design Method. In: 23rd International Conference on Software Engineering, pp. 745-746. IEEE Computer Society, (Year)

[4] Bachmann, F., Bass, L., Klein, M., Shelton, C.: Designing Software Architectures to achieve Quality Attribute Requirements. IEE Proceedings Software 152, 153-165 (2005)

[5] Balasubramaniam, S., Lewis, G.A., Morris, E., Simanta, S., Smith, D.B.: Challenges for Assuring Quality of Service in a Service-oriented Environment. 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems, pp. 103-106. IEEE Computer Society, Vancouver, Canada (2009)

[6] Barbacci, M.R., Ellison, R.J., Lattanze, A.J., Stafford, J.A., Weinstock, C.B., Wood, W.G.: Quality Attribute Workshops (QAWs), Third Edition. Technical Report, SEI CMU (2003)

[7] Barbacci, M.R., Kleiin, M.H., Weinstock, C.B.: Principles for Evaluating the Quality Attributes of a Software Architecture. Technical Report, SEI CMU (1997)

[8] Basili, V., Caldiera, G., Rombach, D.: The Goal Question Metric Approach. In: Marciniak, J.J. (ed.) Encyclopedia of Software Engineering, vol. 1, pp. 528-532. John Wiley & Sons, New York, NY (1994)

[9] Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice. Addison-Wesley, Boston, MA (2003)

[10] Becha, H., Amyot, D.: Non-functional Properties in Service Oriented Architecture - A Consumer's Perspective. Journal of Software 7, 575-587 (2012)

[11] Bennett, C., Khangura, S., Brehaut, J.C., Graham, I.D., Moher, D., Potter, B.K., Grimshaw, J.M.: Reporting Guidelines for Survey Research: An Analysis of Published Guidance and Reporting Practices. PLoS Med 8, 1-12 (2011)

[12] Bosch, J.: Software Architecture: The Next Step. In: First European Workshop on Software Architecture, pp. 194-199. Springer-Verlag, (Year)

[13] Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.: Pattern-Oriented Software Architecture Volume 1: A System of Patterns. Wiley, West Sussex, UK (1996)

[14] Ciolkowski, M., Laitenberger, O., Vegas, S., Biffl, S.: Practical Experiences in the Design and Conduct of Surveys in Empirical Software Engineering. In: Conradi, R., Wang, A.I. (eds.) Empirical Methods and Studies in Software Engineering, pp. 104-128. Springer Verlag, Berlin / Heidelberg (2003)

[15] Cohen, S., Krut, R.: Managing Variation in Services in a Software Product Line Context. Technical Note, CMU SEI (2010)

[16] Creswell, J.W.: Research Design: Qualitative, Quantitative, and Mixed Methods Approaches. pp. 246. Sage Publications, (Year)

[17] de Oliveira Cavalcanti, R., de Almeida, E.S., Meira, S.: Extending the RiPLE-DE Process with Quality Attribute Variability Realization. 7th International ACM Sigsoft

Conference on the Quality of Software Architectures (QoSA), pp. 159-163. ACM, Boulder, CO (2011)
[18] Erl, T.: Service-Oriented Architecture (SOA): Concepts, Technology, and Design. Prentice Hall, Upper Saddle River, NJ (2005)
[19] Ferrari, R.N., Madhavji, N.H.: Architecting-Problems Rooted in Requirements. Information and Software Technology 50, 53-66 (2008)
[20] Franch, X., Carvallo, J.P.: Using Quality Models in Software Package Selection. IEEE Software 20, 34-41 (2003)
[21] Gehlert, A., Metzger, A.: Quality Reference Model for SBA. S-Cube (2009)
[22] Gray, D.E.: Doing Research in the Real World. Sage Publications, London, UK (2009)
[23] Gu, Q., Lago, P.: Exploring Service-oriented System Engineering Challenges: A systematic Literature Review. Service Oriented Computing and Applications 3, 171-188 (2009)
[24] Harrison, N., Avgeriou, P., Zdun, U.: Using Patterns to Capture Architectural Decisions. IEEE Software 24, 38-45 (2007)
[25] IEEE Computer Society Software Engineering Standards Committee: IEEE Standard Glossary of Software Engineering Terminology. vol. IEEE Std 610.12-1990, (1990)
[26] ISO/IEC: Software engineering - Product quality - Part 1: Quality model. vol. ISO/IEC 9126-1, Geneva, Switzerland (2001)
[27] ISO/IEC: Software engineering - Product quality - Part 2: External metrics. pp. 86, Geneva, Switzerland (2003)
[28] ISO/IEC: Software engineering - Product quality - Part 3: Internal metrics. pp. 62, Geneva, Switzerland (2003)
[29] ISO/IEC: Software engineering - Product quality - Part 4: Quality in use metrics. pp. 59, Geneva, Switzerland (2004)
[30] Jansen, A., Avgeriou, P., Ven, J.S.v.d.: Enriching Software Architecture Documentation. Journal of Systems and Software 82, 1232-1248 (2009)
[31] Kim, S., Kim, D.-K., Lu, L., Park, S.: Quality-driven Architecture Development Using Architectural Tactics. Journal of Systems and Software 82, 1211-1231 (2009)
[32] Kitchenham, B., Al-Khilidar, H., Babar, M.A., Berry, M., Cox, K., Keung, J., Kurniawati, F., Staples, M., Zhang, H., Zhu, L.: Evaluating Guidelines for Reporting Empirical Software Engineering Studies. Empirical Software Engineering 13, 37-121 (2008)
[33] Kitchenham, B., Pfleeger, S.L.: Principles of Survey Research - Part 2: Designing a Survey. ACM SIGSOFT Software Engineering Notes 27, 18-20 (2002)
[34] Kitchenham, B., Pfleeger, S.L.: Principles of Survey Research - Part 3: Constructing a Survey Instrument. ACM SIGSOFT Software Engineering Notes 27, 20-24 (2002)
[35] Kitchenham, B., Pfleeger, S.L.: Principles of Survey Research - Part 4: Questionnaire Evaluation. ACM SIGSOFT Software Engineering Notes 27, 20-23 (2002)
[36] Kitchenham, B., Pfleeger, S.L.: Principles of Survey Research - Part 5: Populations and Samples. ACM SIGSOFT Software Engineering Notes 27, 17-20 (2002)
[37] Kitchenham, B., Pfleeger, S.L.: Principles of Survey Research - Part 6: Data Analysis. ACM SIGSOFT Software Engineering Notes 28, 24-27 (2003)
[38] Kitchenham, B.A., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., Emam, K.E., Rosenberg, J.: Preliminary Guidelines for Empirical Research in Software Engineering. IEEE Transactions on Software Engineering 28, 721-734 (2002)

[39] Kontogogos, A., Avgeriou, P.: An Overview of Software Engineering Approaches to Service Oriented Architectures in Various Fields. 18th International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), pp. 254-259. IEEE Computer Society, Groningen, The Netherlands (2009)
[40] Krippendorff, K.: Content Analysis: An Introduction to its Methodology. Sage Publications, Thousand Oaks, CA (2003)
[41] Kruchten, P.: An Ontology of Architectural Design Decisions in Software-intensive Systems. In: 2nd Groningen Workshop on Software Variability, pp. 54-61. (Year)
[42] Mack, N., Woodsong, C., MacQueen, M.M., Guest, G., Namey, E.: Qualitative Research Methods: A Data Collector's Field Guide. Family Health International, Research Triangle Park, NC (2005)
[43] Miles, M.B., Huberman, A.M.: Qualitative Data Analysis. Sage Publications, Thousand Oaks, CA (1994)
[44] O'Brien, L., Bass, L., Merson, P.: Quality Attributes and Service-Oriented Architectures. Technical Note, CMU SEI (2005)
[45] O'Brien, L., Merson, P., Bass, L.: Quality Attributes for Service-oriented Architectures. International Workshop on Systems Development in SOA Environments, pp. 1-7. IEEE Computer Society, Minneapolis, MN (2007)
[46] OASIS: Reference Model for Service Oriented Architecture 1.0. (2006)
[47] Papazoglou, M.: Web Services: Principles and Technology. Prentice Hall, Upper Saddle River, NJ (2007)
[48] Pfleeger, S.L., Kitchenham, B.A.: Principles of Survey Research - Part 1: Turning Lemons into Lemonade. ACM SIGSOFT Software Engineering Notes 26, 16-18 (2001)
[49] Poort, E., Martens, N., van de Weerd, I., van Vliet, H.: How Architects see Non-functional Requirements: Beware of Modifiability. 18th International Working Conference on Requirements Engineering: Foundations for Software Quality (REFSQ), pp. 37-51. Springer Verlag, Essen, Germany (2012)
[50] Shim, B., Choue, S., Kim, S., Park, S.: A Design Quality Model for Service-Oriented Architectures. In: 15th Asia-Pacific Software Engineering Conference, pp. 403-410. IEEE Computer Society, (Year)
[51] Sindhgatta, R., Sengupta, B., Ponnalagu, K.: Measuring the Quality of Service-oriented Design. 7th International Joint Conference on Service-oriented Computing (ISOC-ServiceWave), pp. 485-499. Springer Verlag, Stockholm, Sweden (2009)
[52] Svensson, R.B., Gorschek, T., Regnell, B.: Quality Requirements in Practice: An Interview Study in Requirements Engineering for Embedded Systems. 5th International Working Conference on Requirements Engineering: Foundation for Software Quality, pp. 218-232. Springer Verlag, Amsterdam, The Netherlands (2009)
[53] Svensson, R.B., Gorschek, T., Regnell, B., Torkar, R., Shahrokni, A., Feldt, R.: Quality Requirements in Industrial Practice: An Extended Interview Study at Eleven Companies. IEEE Transactions on Software Engineering 38, 923-935 (2011)
[54] Tofan, D., Galster, M., Avgeriou, P., Schuitema, W.: Past and future of software architectural decisions - A systematic mapping study. Information and Software Technology 56, 850-872 (2014)
[55] Tyree, J., Akerman, A.: Architecture Decisions: Demystifying Architecture. IEEE Software 22, 19-27 (2005)

[56] van Heesch, U., Avgeriou, P.: Mature Architecting - A Survey about the Reasoning Process of Professional Architects. 9th Working IEEE/IFIP Conference on Software Architecture, pp. 260-269. IEEE Computer Society, Boulder, CO (2011)
[57] van Heesch, U., Avgeriou, P.: Naive Architecting - Understanding the Reasoning Process of Students - A Descriptive Survey. 4th European Conference on Software Architecture, pp. 24-37. Springer Verlag, Copenhagen, Denmark (2010)
[58] Voelz, D., Goeb, A.: What is Different in Quality Management for SOA? 14th IEEE International Enterprise Distributed Object Computing Conference (EDOC), pp. 47-56. IEEE Computer Society, Vitoria, Brazil (2010)
[59] Vogt, P.: Dictionary of Statistics and Methodology - A Non-technical Guide for the Social Sciences. Sage Publications, Thousand Oaks, CA (2005)
[60] Wohlin, C., Hoest, M., Henningsson, K.: Empricial Research Methods in Software Engineering. In: Conradi, R., Wang, A.I. (eds.) Empirical Methods and Studies in Software Engineering, pp. 7-23. Springer Verlag, Berlin / Heidelberg (2003)
[61] Wojcik, R., Bachmann, F., Bass, L., Clements, P., Merson, P., Nord, R., Wood, B.: Attribute-Driven Design (ADD), Version 2.0. Technical Report, SEI CMU (2006)