An Industry-Driven Template for the Documentation of Non-Functional Requirements

Sabine Molenaar[1] and Fabiano Dalpiaz[1]

Dept. of Information and Computing Sciences, Utrecht University,
Utrecht, The Netherlands
{s.molenaar,f.dalpiaz}@uu.nl

Abstract. [Context] In software development, the quality attributes that systems should exhibit are often expressed as non-functional requirements (NFRs). However, unlike functional requirements, there is no widely adopted writing format for NFRs. **[Problem]** This deficiency, and the more complex nature of NFRs due to their difficult quantification and architectural level of abstraction, poses obstacles for the specification of NFRs in practice. **[Results]** Following the design cycle methodology, we put forward a process for developing an NFR template based on meta-requirements from literature and gathered from practitioners in a focus group, and two existing templates. We validated the template through interviews with target users from three organizations, focusing on whether it could help practitioners overcome specific challenges in NFR writing. **[Contribution]** The interviewees appreciated the explicit documentation of measurable fit criteria the most, as it allows users to make the NFR quantifiable, and found the template easy to use. All expressed an intention to use the template again in the future.

Keywords: Non-functional requirement · Quality requirement · Requirements engineering · Design science.

1 Introduction

Non-Functional Requirements (NFRs) are by nature more difficult to validate than functional requirements, as they are generally formulated in a way that is not measurable and they are usually subjectively evaluated [7]. NFRs are often not considered [13], and software at times fulfills NFRs in ad-hoc ways, as opposed to functional requirements, which are taken into account when software is designed [3]. In 2004, Paech and Kurkow stated that the understanding of NFRs is still limited and that they are barely discussed in prominent RE textbooks [26].

Glinz confirmed that quality requirements are described in a qualitative way such as "*we want an easy to use system*", but this makes them subjective and difficult to measure [11]. In 2016, Eckhardt *et al.* still found some NFRs too vague to include in their study [8].

In 2018, Kopczyńska *et al.* [16] found that NFRs are still described as "*too often neglected, especially those that are difficult to write and ostensibly obvious*".

They also found that using a template results in more complete NFRs, which increases their verifiability, when compared to ad-hoc approaches to NFRs. This study used a catalog of 400 templates, and found that about 20% of the templates were used for almost 80% of the NFRs across the 40 investigated projects. Also, each project used less than 10% of the templates on average [16].

For functional requirements, many popular templates exist, such as the User Story (US) template by Mike Cohn [5], EARS [20], and FRETish [9], to name a few. In addition, guidelines and instructions exist for functional requirements, such as the ISO/IEC/IEEE 29148:2018 standard [1], the Quality User Story framework [19] and INVEST [33]. NFRs are not afforded the same luxury: while quality characteristics for NFRs are available [16], they are not commonly used.

In a previous study, we found that practitioners are unsure of how to document NFRs and, in an attempt to document them in a consistent and familiar way, try to 'force' them into a US template [23]. We intend to tackle this difficulty and uncertainty in expressing NFRs with a design cycle [34]. Our process demonstrates how to develop an artifact that supports practitioners in documenting NFRs in their particular real-life context. We take a novel approach, by actively engaging practitioners through the elicitation of meta-requirements, based on challenges they encounter in practice and by evaluating the suitability of existing templates. This ensures a tailored artifact, rather than a one-size-fits-all solution, which is unlikely to suit the context optimally.

This paper makes the following contributions to the literature:

1. We present a process for developing a situational NFR template, tailored to a specific context;
2. Through meta-requirements from literature and practitioners, following our process, we construct an industry-aligned NFR template that we name *iNFR*;
3. We validate *iNFR* through interviews with seven practitioners who could use the template. They valued the fit criteria most, as they allow users to make NFRs more specific and quantifiable. They found the template easy to use and expressed intention to use in the future.

Paper organization. We discuss relevant terminology in Sec. 2. We present our research method in Sec. 3 and we discuss the problem and available NFR definition methods in Sec. 4. We describe the design of the treatment in Sec. 5, followed by the treatment validation in Sec. 6. Finally, we present a discussion and conclusion in Sec. 7 and Sec. 8, respectively.

2 Background

Lawrence *et al.* identified ignoring NFRs as one of the top ten risks of requirements engineering in 2001 [18]. NFRs help ensure constraints on the system architecture are considered during design, instead of requiring later refactoring [4]. Paech and Kerkow explain that the need for NFRs only grows as the IT market becomes more competitive, both because society as a whole becomes more dependent on IT, so the system and system components on which we rely

should be of high quality, and because customers and suppliers need to negotiate on the quality they expect [26].

We also discuss the basic terminology for this paper. In a previous study [22], we found that the terms NFR and Quality Requirement (QR) are used interchangeably by some, while others assign them different meanings. We follow the latter and use Glinz's definitions [12]:

1. NFR: "*a quality requirement or a constraint*".
2. QR: "*a requirement that pertains to a quality concern that is not covered by functional requirements*".
3. Constraint: "*a requirement that limits the solution space beyond what is necessary for meeting the given functional requirements and quality requirements*".
4. Functional requirement: "*a requirement concerning a result of behavior that shall be provided by a function of a system (or of a component or service)*".

For the sake of brevity and legibility, we will use the term NFR, which covers both QRs and constraints. In addition, we will not discuss which terms other, related works used and if this was the correct term and how it was defined, we will simply use the term mentioned in the original work.

There is some discourse regarding what an NFR is and if, because most NFRs describe system behavior, they cannot be managed and assessed like functional requirements instead. According to Eckhardt *et al.*: "*Our results suggest that most "non-functional" requirements are not non-functional as they describe behavior of a system. Consequently, we argue that many so-called NFRs can be handled similarly to functional requirements.*" [8]. In our previous study, practitioners shared similar sentiments; they find it difficult to judge when NFRs become functional in a previous study [23].

However, NFRs should be considered at architecture level, rather than related to specific features or functionalities of a system or system component. Rozanski & Woods refer to quality goals as 'cross-cutting concerns' that are "*likely to impact all of the different structures that make up your architecture*" [31]. Cleland-Huang *et al.* confirm this, stating: "*Early detection of NFRs is useful because it enables system level constraints to be considered and incorporated into early architectural designs as opposed to being refactored in at a later time.*" [4].

3 Research method

We address a design problem, namely: *how to design an industry-specific template for defining NFRs?* We employ a design cycle, which consists of three phases: (i) problem investigation, (ii) treatment design, and (iii) treatment validation [34]. This process is an example of how a situational NFR template can be developed for specific contexts, such as organizations or even departments. Supporting materials are made available through an online appendix [21]. For phases one and two we used both literature and input from practitioners. Phase three solely involves feedback from practitioners. The research design is shown in Figure 1.

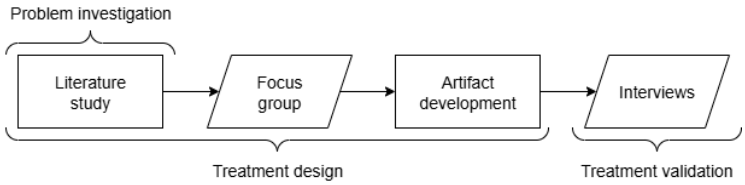

Fig. 1. Research design showing the three design science phases (parallelograms indicate activities involving practitioners).

Problem investigation. We analyzed the literature and used findings from our previous work [22], for which data was gathered at the end of 2022. We focused on gathering information about the importance of NFRs, as well as difficulties practitioners may face in defining them, and available treatments in preparation for the treatment design phase. The relevant studies are discussed in Section 4.

Treatment design. Our main focus is on the elicitation and specification of meta-requirements, the available treatments and, potentially, the development of a new treatment. We refer to requirements for the treatment as 'meta-requirements', to avoid confusion. We first consider the findings form our review study [22], we investigated the characteristics of definitions of the terms NFR and QR. We broaden this by gathering information from practitioners. From them, we wish to get answers to two questions: (i) how do they currently define NFRs and (ii) what are the advantages of and obstacles with available treatments? We used a focus group to obtain answers to these evaluative questions [32].

The focus group session included three participants, who discussed their process of defining and documenting NFRs. Afterwards, the participants were shown two NFR templates and asked to document an NFR using these templates. Benefits and limitations were discussed in the group, and the findings led to a set of meta-requirements. Both the meta-requirements gathered from the literature and the focus group are then used to either modify an existing treatment or create a new one; we refer to this activity as 'artifact development' in Figure 1.

Treatment validation. The modified or newly developed artifact from the previous phase is validated through expert opinion, as recommended by Wieringa [34]. Experts are asked to use the artifact and answer questions regarding the use through a one-on-one semi-structured interview. These questions are based on the input provided by the focus group participants, namely whether the treatment could help them overcome the challenges they described. In addition, participating experts are asked to answer questions regarding usefulness, ease of use, and intention to use.

Participants. Demographics of the participants are summarized in Table 1. All participants were selected through convenience sampling; willingness and availability to participate, although experience with definition and management of requirements was a prerequisite.

We present anonymized descriptions and data due to confidentiality agreements and to prevent information from being traced to a specific individual. 'Organization A' is a large organization based in the Netherlands, that develops and maintains a large number of applications. Development teams are organized in agile release trains and over 200 agile teams are active within the organization. In addition to the focus group participants, we also interviewed two IT specialists from the organization, who were uninvolved in the study until that point. To get an outside perspective, we invited two IT consultants from different consultancies to participate in the validation. Both IT consultancies are based in the Netherlands and have fewer than fifty employees.

Table 1. Demographics of participants in focus group and interviews.

	Focus group	**Interviews**	
	Org. A	**Org. A**	**Consultants**
Participants	3	5	2
Type of role(s)	Product owner, scrum master	Product owner, scrum master, IT specialist	IT consultant
Experience with reqs.	4 – 5 years	4 – 15 years	8 years

4 Problem investigation

In a previous study, we found that practitioners experience difficulty in expressing NFRs, because they are unsure of how to document them. Practitioners oftentimes forced them into a US template [23]. Through a literature study, we investigate the challenges in defining NFRs further in Section 4.1. In addition, we describe existing methods for defining NFRs in Section 4.2.

4.1 Challenges in defining NFRs

Even though NFRs are deemed important for project success, they are notoriously difficult to define and validate. There are three main factors that make them more complex than functional requirements: (i) they have a strong impact on the architecture, (ii) they are hard to quantify, as they are of a qualitative nature rather than a quantitative one, and (iii) there is no one-size-fits-all approach. The challenges are described in the following paragraphs.

Architecture level. As discussed in Section 2, NFRs should be considered at architecture level [4, 31]. Ebert states that "*improving an architecture once it is not maintainable any more is impossible*" [7], but that delaying the implementation of technical requirements will not endanger the system in question. However, he also explains that NFRs are often not measurable during the design phase [7]. So, while NFRs need to be considered during the design of the system and express requirements for the system as a whole, they can only be tested once the system has been built. The IREB guide follows this view and defines quality

requirements as requirements that "*influence the system architecture more than functional requirements do*" and constraints as types of requirements that "*can constrain the system itself*" and impose limitations during development [28]. Also, this makes NFRs harder to trace than functional requirements [3, 7].

Quantification. As stated previously, NFRs often cannot be measured during the design phase [7], but matters become more complicated if measurement of the NFRs becomes complex, or even impossible, in itself. Eckhardt *et al.* state that NFRs are often documented vaguely and without quantitative measures, making them difficult to test [8].

Glinz identifies three types of problems as a result of ambiguous and unverifiable quality requirements: (i) the quality of the system does not meet stakeholder expectations, (ii) the system outperforms the stakeholder expectations, potentially making the system more expensive than it needs to be, and (iii) developers and stakeholders cannot agree on whether quality requirements are met, as they cannot be measured objectively [11].

In their empirical study on NFR templates, Kopczyńska *et al.* [16] provide users with some guidance, by presenting quality characteristics for a single NFR. Two of those characteristics, individual completeness and verifiability, require the user to ensure that it is possible to verify the NFR.

Lack of a single approach. Previously we described that NFRs lack a common method [3, 26] and that many definition methods exist, while for functional requirements the approach may seem more obvious given the context.

4.2 Available NFR definition methods

While NFRs may not have a best practice definition method like functional requirements in agile in the form of the Connextra template for USs [5], various approaches have been proposed over the years. Note that we consider NFR templates as a specific NFR definition methods. We therefore use the the terms 'NFR definition method' and 'NFR template' interchangeably. We discuss a non-exhaustive selection of NFR definition methods in the following paragraphs.

Bowen *et al.* presented consumer-oriented software quality attributes, which link quality attributes to user concerns in the form of questions. For instance, when considering the performance of the software: "*How well does it utilize a resource?*" [2]. Mylopoulos *et al.* used these quality attributes in their goal graph modeling approach, which presents NFRs as related goals in a graph and shows whether and to what degree these goals are supported by the design of the system [24]. This evolved to a modeling formalism for representing how functional requirements affect soft goals (e.g., sharing of information), either positively or negatively, and these soft goals are then related to NFRs. This approach allows for modeling and reasoning about the impact of system attributes on NFRs; an NFR is satisfied when all its related soft goals are satisfied and it is not negatively impacted by any attributes [25].

Ebert specified NFRs using tables that contain quality goals, priority, measurable targets, related quality and release criteria, and applicable metrics. Using

this approach, NFRs can be measured quantitatively [7]. The Quality-Aware Predesign Model (QAPM) also contains a table to represent quality requirements, which includes, among others: quality characteristic, threshold, applicability, and description [14]. Similarly, Cysneiros *et al.* presented the NFR description language that explains how an NFR should be described. The language includes a name, description, related design decisions, priority, and attributes [6].

The first four steps of the QUality PERformance (QUPER) approach also aim to define quality indicators for a system or product. A template was made to cover these first four steps and require information such as quality type, market expectations related to this quality (e.g. a five second response-time), quality indicators of competitor products and targets. This template focuses mainly on quantitative specifications [29].

Robertson & Robertson propose the use of "The system shall..." template, followed by some NFR the system should meet. However, this Volere Snow Card (VSC) template also includes a rationale, originator, fit criterion, dependencies and support materials among others. This template can also be used for functional requirements. For NFRs, they prescribe properties the NFRs should cover; for usability, they mention ease of remembering as example property [30].

Kopczyńska & Nawrocki presented a more specific template in 2014, called the Non-Functional Requirement Template (NoRT). This template differs for different types of NFRs, because it includes standard text and parameters the user needs to fill in. For instance, the performance requirement template is: "*[Average/maximum in <number>% cases] time between <request> and <response> should be longer than <time amount>* [15].

5 Treatment design

We base the treatment design on requirements gathered from literature and the industry; we elicited the latter from practitioners through a focus group.

5.1 Meta-requirements from literature

In our previous study on the definitions of NFRs and QRs, we identified four characteristics these definitions typically include [22]: (i) distinguished from functional requirements, (ii) describe a quality attribute the system or system component should exhibit, (iii) often include examples of NFR/QR types (e.g., availability), and (iv) may constrain functional requirements.

From these characteristics, we define three meta-requirements for the artifact (the first and fourth characteristics were combined into L-RQ1):

L-RQ1. The artifact must link an NFR to one or more functional requirement(s);
L-RQ2. The artifact must specify for which system or system component an NFR is relevant;
L-RQ3. The artifact must include examples of (quality) attributes.

5.2 Meta-requirements from industry

We invited three practitioners from Organization A to participate in a focus group to explain how they currently define NFRs and what the (dis)advantages are in available treatments, in autumn 2025. We selected two templates for the focus group: the VSC template [30] and NoRT [15].

We selected these two because they take different approaches regarding degree of freedom and granularity; the former is high-level and allows for a lot of freedom, while the second is more specific and restrictive. For the VSC template, we used the "*The system shall*" phrase and provided the participants with the additional guidance for NFRs presented in [30]. For NoRT, a few examples were made available (e.g., the performance format) and participants only needed to fill in the given parameters.

Current way of working. All three participants stated that they define NFRs based on functional requirements. In their opinion, the need of expressing an NFR depends on the functional requirement at hand. There is one exception, however, as Organization A has defined a set of *security* and *privacy* NFRs that are applicable to all projects and systems.

They do not use a specific format or template. One of the participants often uses a US template, another includes them as checklist items and the third often records NFRs as part of the acceptance criteria for a US. All participants reuse previously defined NFRs if possible.

Challenges in defining NFRs. The main challenge the participants face in defining NFRs is making them quantifiable; if they are not, they are difficult to test (as stated in [8]). Adherence to some NFRs can only be assessed once the system is in place, according to the participants. One of them mentioned *availability* as an example, which needs the system to be up and running and it may evolve over time (see also [7]).

One participant explained that they would like to apply the same NFRs to all users or user groups, but this is sometimes not possible. This leads to having various levels of quality within an NFR, as the requirements are more strict for some user groups than others. This occurs when one type of NFR is considered more important (e.g., *accessibility* over *usability*). Moreover, this trade-off between NFRs should become apparent from their prioritization, but this is rarely the case as NFRs are defined in relation to functional requirements.

More generally, the participants stated that they lack guidelines or best practices (as described in [26]) for defining NFRs. It is also difficult to determine a level of quality across requirements, making it hard to determine a minimum threshold to meet and enforcing it (similar to [4, 31]).

Evaluation of available treatments The participants defined NFRs using the VSC template and NoRT and were asked about the main benefits and limitations of the two treatments after using them. The most relevant feedback from the participants is presented in Table 2.

Table 2. Benefits and limitations of VSC and NoRT according to the focus group participants.

	VSC	**NoRT**
Benefits	The user has more freedom (it allows you to include motivations for example) The properties to consider are helpful It invites you to create a list	It helps you along, as part of the requirement is already included It is more specific
Limitations	Difficult to make it more specific/testable	If you include a template for every type of NFR, team members are less likely to read them Risk of filling it in for the sake of filling it in Not a one-size-fits-all-solution; users may need to deviate

The participants said that how much guidance you need, for instance via a template, depends on (i) your experience with writing NFRs, (ii) your experience with the system at hand, and (iii) the type of NFR you are working on.

The type of NFR matters, because 'user-oriented NFRs' (e.g., *usability*) are generally less clear and specific than 'technology-oriented NFRs' (e.g., *availability*). They explain that the second type is often easier to measure as they are objective, while the first type depends on the user and is therefore subjective. In addition, they consider the technology-oriented NFRs to be more stable, while the user-oriented NFRs can evolve over time. They think this distinction should be included in a template. A similar distinction is made in [7]; between user-oriented and development-oriented. Ebert, however, sees user-oriented as being of most interest to the customer, while the practitioners consider user-oriented as interaction with the user.

Ideally, they would prefer a combination of the two templates. The first should function as a starting point, which can then be refined to include more details. The participants appreciated the properties to consider in the first treatment and the template of the second treatment, because that makes the attributes you might want to describe explicit. They explained that this forces the users to consider everything, serves as a reminder, and prevents you from making assumptions (i.e., 'attribute x should be obvious for everyone').

We have summarized the feedback from the focus group participants into industry meta-requirements for the artifact:

I-RQ1. The artifact must include different (quality) attributes to consider;
I-RQ2. The artifact must allow for further refinement of an NFR;
I-RQ3. The artifact must distinguish between user-oriented and technology-oriented NFRs;
I-RQ4. The artifact must include a level of priority;
I-RQ5. The artifact must distinguish between NFRs that should be tested during run-time and can be tested offline;
I-RQ6. The artifact must distinguish between NFRs that should be tested periodically (or continuously) or once (prior to implementation);
I-RQ7. The artifact must specify whether the NFR is an exception and/or specification of a different NFR.

5.3 Artifact development

First, we created a conceptual model to illustrate the relationships between the various concepts in the study; identified through literature and empirical research. This model can be seen as a high-level specification for the artifact. The conceptual model is shown in Figure 2.

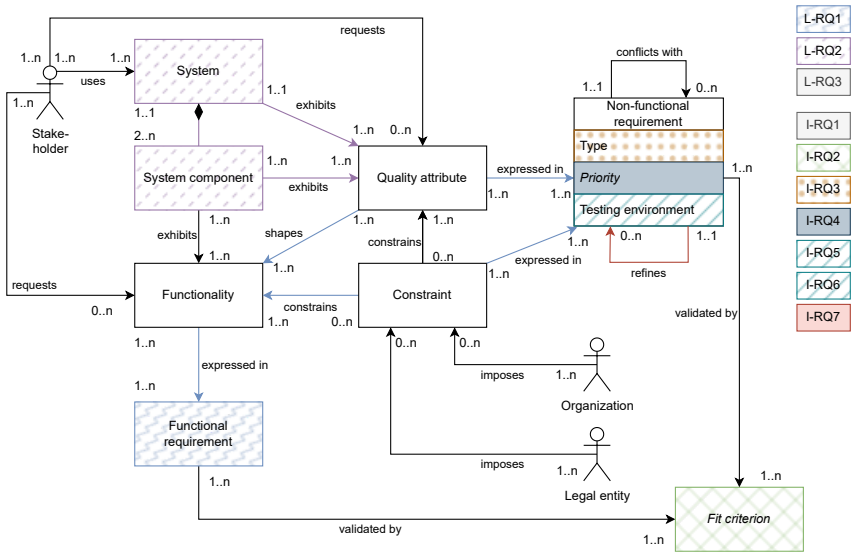

Fig. 2. Conceptual model of NFR and related concepts, highlighting elements that support the requirements for the artifact (included VSC elements are in italics).

In line with the literature [10], NFRs can express quality attributes or constraints. NFR have a type, hinting to the adoption of NFR taxonomies. This is partially covered in the quality characteristics for NFRs, which state that NFRs should be unambiguous [16]. NFRs constrain or shape functionality, so these functionality also need a common understanding. We assume, for example, that these functionalities are expressed in functional requirements (e.g., in USs).

We decided to relate quality attribute both to system components and to the system as a whole. While certain quality attributes can only be understood by looking at the system as a whole (e.g., performance), others need to be specified on individual components as part of a refinement process [27].

5.4 The iNFR template

The developed artifact, the industry-NFR (iNFR) template, is shown in Figure 3.

ID: NFR-0001			
Priority level: 1			
System (component): System Y			
Source *(choose one)*	X Stakeholder: End-user	O Own organization: …	O Legal entity: …
Type *(choose one per column)*	O Desire, user-oriented	O Usability O Other: …	
	X Desire, technology-oriented	O Efficiency O Maintainability O Portability X Reliability O Security O Other: …	
	O Constraint	O Cultural O Environmental O Implementation & design O Interface O Legal O Other: …	
Description *The system shall …* be available 99.8% of the time during office hours.			
Fit criteria 1. Down time must not exceed 4.8 minutes per work week. 2. Maintenance must be scheduled and performed on weekends. 3. Users must be informed of scheduled maintenance 4 weeks in advance.			
Validation *(choose one)*	X Run-time online	O Run-time offline	O Compile-time
Testing *(choose one)*	X Continuous	O Periodical	O Single test
Conflicting NFR(s) *(mandatory in case of constraint)* n/a			
Impacted functional requirement(s) *(mandatory in case of constraint)* FR-0005			

Fig. 3. The developed iNFR template, showing a fictitious example.

The focus group stated that they would prefer a combination of the VSC and NoRT templates; start with a high-level description, which can be further refined. We met this request by using the "*The system shall*" phrase from the VSC template for the general description and including fit criteria, a term used by Robertson & Robertson to denote "*a quantification or measurement of the requirement such that you are able to determine whether the delivered product satisfies the requirement*" [30]. We also explicitly included NFRs types, as per the needs of the focus group (I-RQ3) and categorized them into three types: technology-oriented, user-oriented and constraints. In this categorization, we follow the practitioners' inputs rather than existing proposals like Ebert's [7].

For the NFR examples (L-RQ3/I-RQ1), we used the NFR classes listed by Eckhardt and colleagues: *efficiency*, *maintainability*, *portability*, *reliability*, *security*, and *usability*, because they were extracted from industry requirements specifications [8]. We assume that these are understood by other practitioners as well. In addition, for the constraints, we used Glinz's taxonomy for example types [10]. All types are included in alphabetical order in the template. For use in other contexts, we recommend using a taxonomy commonly used in that context. For instance, a company could select the taxonomy, so that all employees have a common understanding. We do not prescribe a specific taxonomy, because they can be domain-specific, but examples are presented in [2, 7, 10, 30, 8].

We also included fields that allow users to specify how and when the NFR should be validated and tested. The focus group also discussed NFRs that might

conflict with other NFRs. We fulfilled this need by including "conflicting NFRs". Finally, we included a field for functional requirements that are impacted by the NFR in question. In a real-world setting, the template should be integrated into an issue tracking system.

6 Treatment validation

As described in Section 3, we validated the template through expert opinion. All experts were given the NFR template, four fictitious example NFRs created using the template and a consent form. All interviews with the experts were conducted by the same researcher in autumn 2025. The experts were first asked to define a (fictitious) NFR for a self-chosen (fictitious) system using the empty template; the examples served as support material. The experts provided positive and negative feedback regarding fields in the template, which is summarized in Table 3.

Table 3. Feedback on template fields, where "responses" is the number of experts that provided the same feedback (out of the total of seven experts).

Template field	Feedback	Responses
ID	n/a	n/a
Priority	Default options (in line with organization) would be nice	1
System (component)	n/a	n/a
Source	Often apparent from description; can be considered redundant	3
	"Initiator" might be a better term	2
	Multiple sources should be possible	1
Type	Somewhat confusing; how to read and what is expected	3
	Easier to select a type than having to write one from scratch	4
Description	n/a	n/a
Fit criteria	Allow you to make the NFR more specific and quantifiable	6
Validation	Not immediately clear what the use of this field is	3
Testing	Nice that this is made explicit, as it forces you to consider this	3
	More fields to provide more details would be better	4
Conflicting NFRs	Nice that this is included, but still needs to be assessed manually	3
Impacted FRs	Dependencies are essential in the development process, important that this is included	2

The experts then evaluated whether the template could support practitioners in overcoming the challenges from the focus group. We identified four challenges, one of which, concerning quantifiability (*C3*), is confirmed by the literature, as discussed in Section 4. Finally, they were asked to rate the usefulness, ease of use and intention to use of the template. Both the challenges and the template were evaluated on a five-point Likert scale (from 1 – strongly disagree to 5 – strongly agree), the scores are shown in Table 4.

The consolidated scores per statement are visualized in Figure 4. Note that 'strongly disagree' is not included, because this score was not observed.

The experts were generally positive, with some reservations, particularly for C4. They see potential, but would have to use the template from start to finish (refinement to implementation) in their development process to be able to assess

Table 4. Validation results showing scores on the Likert scales for each expert (E), E6 and E7 are the consultants outside Organization A.

ID	Challenge statements	E1	E2	E3	E4	E5	E6	E7
C1	NFRs may conflict with others, which makes it more difficult to make the right decision in selecting which NFR(s) to implement.	4	4	3	4	4	4	4
C2	Some NFRs may be forgotten or overlooked.	4	4	4	4	4	4	5
C3	It is often difficult to make NFRs quantifiable and therefore testable.	4	5	3	5	2	5	3
C4	Some NFRs require specific circumstances for testing, but they are often not made explicit.	2	4	4	5	2	2	5
ID	**Template statements**	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**
T1	The template was useful for defining an NFR.	5	4	3	4	4	4	5
T2	The template was easy to use.	4	4	4	5	3	5	4
T3	I would use this template again to define an NFR.	4	5	4	4	4	4	4

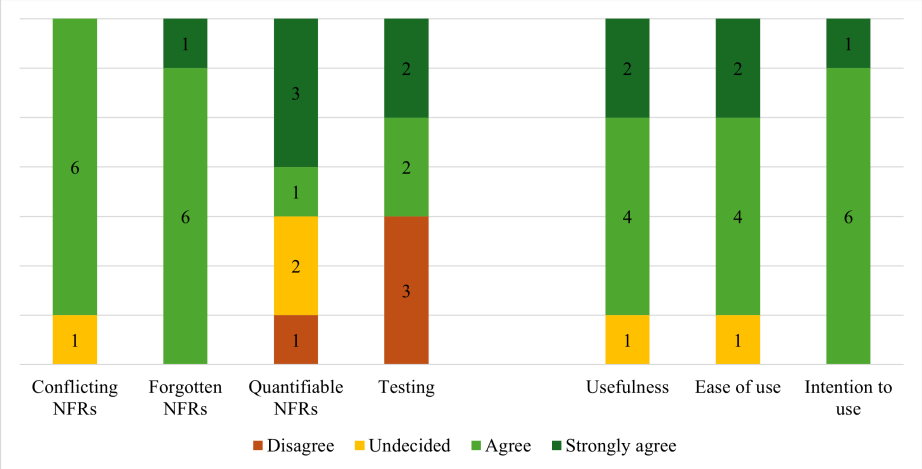

Fig. 4. Consolidated validation scores per statement.

the value of the template in overcoming this challenge (same for C1). Currently, there is no proof that these NFRs are testable (also related to C3) in this way and the testing field does not provide enough details for proper testing to be performed. All experts agreed that the template was useful and easy to use and that they intend to use the template again in the future to define an NFR.

7 Discussion

We use the design science research validity framework to discuss the threats to validity, categorized in criterion, causal, and context validity [17].

Criterion validity. The meta-requirements for the artifact were determined by gathering input from a focus group containing participants from the target user group, namely practitioners. We triangulated these meta-requirements

using literature; which often stated similar conclusions. We also gathered meta-requirements from literature, in addition to the industry meta-requirements.

Causal validity. While validation of the artifact is limited and can only attest to the perceived mitigation of challenges, interviewees generally found the template useful for defining an NFR and would use the template again in the future. They did express doubts regarding some fields in the template, for instance, they are unsure how useful the template is in prioritizing conflicting NFRs. Similarly, they would need to use the template from refinement to implementation in the development process in order to estimate the usefulness of the validation and testing fields.

Context validity. We elicited meta-requirements for the artifact from target end-users and performed validation through expert opinion. The focus group participants were involved in validation, as they are familiar with the described challenges, as well as two uninvolved experts to get a broader and more objective perspective. To gather more insight into the contextual validity, we also asked experts from different organizations to evaluate the template.

Future Work. While the interviewees stated an intention to use the artifact, it remains uncertain whether they will use it and, if so, to what extent. If the use of the NFR template results in higher quality NFRs was not considered in this study, but could provide valuable insight. Furthermore, the experts stated they would need to use the template in a full development process to be sure of the benefits regarding conflicts and testing. LLMs could potentially be used to help practitioners fill in and adhere to the template or even propose draft NFRs based on existing system documentation. Finally, we identify an opportunity to formalize the process described into a method.

8 Conclusion

We used a design cycle to address the design problem: *how to design an industry-specific template for defining NFRs?* We identified three meta-requirements from related literature, as well as seven meta-requirements from industry through a focus group session. We used these meta-requirements, as well as additional insights provided by the focus group participants, to create a conceptual model. The developed artifact, the iNFR template, is based on this model and industry needs. We then invited four practitioners to provide their expert opinion and validate the template. They expect that practitioners could be supported in overcoming the challenges described by the focus group, but would like to use the template in a full development process to test the full benefits. They valued the 'fit criteria' field most, as this helps make NFRs more specific and quantifiable. The experts generally found the template useful as well as easy to use and all experts expressed an intention to use the template to define an NFR again in the future. Through our iNFR-template, we propose a process for developing a situational NFR template that can be tailored to specific contexts and organizations, which can be used by others as well.

Data Availability Statement. Due to confidentiality agreements, no raw data could be shared. Instead, the focus group and interview protocols, as well as the iNFR template (including fictitious examples) are made available [21].

References

1. ISO/IEC/IEEE 29148:2018 systems and software engineering — Life cycle processes — Requirements engineering (2018), second edition, 2018-12
2. Bowen, T.P., Wigle, G.B., Tsai, J.T.: Specification of software quality attributes. Rome Air Development Center, Air Force Systems Command (1985)
3. Chung, L.: Representation and utilization of non-functional requirements for information system design. In: Proceedings of the International Conference on Advanced Information Systems Engineering. pp. 5–30. Springer (1991)
4. Cleland-Huang, J., Settimi, R., Zou, X., Solc, P.: The detection and classification of non-functional requirements with application to early aspects. In: Proceedings of the International Requirements Engineering Conference. pp. 39–48. IEEE (2006)
5. Cohn, M.: User stories applied: For agile software development. Addison-Wesley Professional (2004)
6. Cysneiros, L.M., do Prado Leite, J.C.S., de Melo Sabat Neto, J.: A framework for integrating non-functional requirements into conceptual models. Requirements Engineering **6**(2), 97–115 (2001)
7. Ebert, C.: Dealing with nonfunctional requirements in large software systems. Annals of Software Engineering **3**(1), 367–395 (1997)
8. Eckhardt, J., Vogelsang, A., Fernández, D.M.: Are "non-functional" requirements really non-functional? an investigation of non-functional requirements in practice. In: Proceedings of the International Conference on Software Engineering. pp. 832–842 (2016)
9. Giannakopoulou, D., Mavridou, A., Rhein, J., Pressburger, T., Schumann, J., Shi, N.: Formal requirements elicitation with FRET. In: Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality (2020)
10. Glinz, M.: On non-functional requirements. In: Proceedings of the IEEE International Requirements Engineering Conference. pp. 21–26. IEEE (2007)
11. Glinz, M.: A risk-based, value-oriented approach to quality requirements. IEEE Software **25**(2), 34–41 (2008)
12. Glinz, M.: A glossary of requirements engineering terminology: Standard glossary of the Certified Professional for Requirements Engineering (CPRE) studies and exam, version 1.0. Tech. rep., IREB, Karlsruhe, Germany (2011)
13. Grimshaw, D.J., Draper, G.W.: Non-functional requirements analysis: deficiencies in structured methods. Information and Software Technology **43**(11), 629–634 (2001)
14. Kop, C., Mayr, H.C.: Conceptual predesign bridging the gap between requirements and conceptual design. In: Proceedings of the International Symposium on Requirements Engineering. pp. 90–98. IEEE (1998)
15. Kopczyńska, S., Nawrocki, J.: Using non-functional requirements templates for elicitation: A case study. In: Proceedings of the International Workshop on Requirements Patterns. pp. 47–54. IEEE (2014)
16. Kopczyńska, S., Nawrocki, J., Ochodek, M.: An empirical study on catalog of non-functional requirement templates: Usefulness and maintenance issues. Information and Software Technology **103**, 75–91 (2018)

17. Larsen, K.R., Lukyanenko, R., Mueller, R.M., Storey, V.C., VanderMeer, D., Parsons, J., Hovorka, D.S.: Validity in design science research. In: Proceedings of the International Conference on Design Science Research in Information Systems and Technology. pp. 272–282. Springer (2020)
18. Lawrence, B., Wiegers, K., Ebert, C.: The top risk of requirements engineering. IEEE Software **18**(6), 62–63 (2001)
19. Lucassen, G., Dalpiaz, F., van der Werf, J.M.E., Brinkkemper, S.: Improving agile requirements: the quality user story framework and tool. Requirements engineering **21**(3), 383–403 (2016)
20. Mavin, A., Wilkinson, P., Harwood, A., Novak, M.: Easy approach to requirements syntax (EARS). In: Proceedings of the IEEE International Requirements Engineering Conference. pp. 317–322. IEEE (2009)
21. Molenaar, S.: iNFR template online appendix (2026), https://doi.org/10.5281/zenodo.18267403
22. Molenaar, S., van den Berg, N., Dalpiaz, F., Brinkkemper, S.: Concept definition review: A method for studying terminology in software engineering. Information and Software Technology **180**, article nr. 107648 (2025)
23. Molenaar, S., Dalpiaz, F.: Improving the writing quality of user stories: A canonical action research study. In: Proceedings of the International Product-Focused Software Process Improvement Conference. pp. 102–118. Cham: Springer Nature Switzerland (2025)
24. Mylopoulos, J., Chung, L., Nixon, B., et al.: Representing and using nonfunctional requirements: A process-oriented approach. IEEE Transactions on Software Engineering **18**(6), 483–497 (1992)
25. Mylopoulos, J., Chung, L., Yu, E.: From object-oriented to goal-oriented requirements analysis. Communications of the ACM **42**(1), 31–37 (1999)
26. Paech, B., Kerkow, D.: Non-functional requirements engineering-quality is essential. In: Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality Workshops. pp. 237–250 (2004)
27. Pohl, K.: Requirements Engineering: Fundamentals, Principles, and Techniques. Springer Verlag, Heidelberg, Germany (2010)
28. Pohl, K.: Requirements engineering fundamentals: A study guide for the certified professional for requirements engineering exam-foundation level-IREB compliant. Rocky Nook, Santa Barbara, CA, USA (2016)
29. Regnell, B., Svensson, R.B., Olsson, T.: Supporting roadmapping of quality requirements. IEEE Software **25**(2), 42–47 (2008)
30. Robertson, S., Robertson, J.: Mastering the requirements process: getting requirements right, 3rd ed. Addison Wesley, Boston, MA, USA (2013)
31. Rozanski, N., Woods, E.: Software systems architecture: Working with stakeholders using viewpoints and perspectives, 2nd ed. Addison Wesley, Boston, MA, USA (2012)
32. Verhoeven, N.: Doing research: The Hows and Whys of Applied Research, 5th ed. Boom uitgevers, Amsterdam, The Netherlands (2019)
33. Wake, B.: INVEST in Good Stories, and SMART Tasks. http://xp123.com/articles/invest-in-good-stories-and-smart-tasks/, accessed 12-03-2025 (2003)
34. Wieringa, R.: Design science methodology for information systems and software engineering. Springer Verlag, Heidelberg, Germany (2014)