Received: 21 February 2022 | Revised: 22 January 2024 | Accepted: 4 March 2024

DOI: 10.1002/smr.2668

RESEARCH ARTICLE - EMPIRICAL

Journal of Software: Evolution and Process | WILEY

Context factors perceived important when looking for similar experiences in decision-making for software components: An interview study

Efi Papatheocharous[1,2] | Claes Wohlin[2] | Deepika Badampudi[2] | Jan Carlson[3] | Krzysztof Wnuk[2]

[1]RISE Research Institutes of Sweden, Kista, Sweden

[2]Blekinge Institute of Technology, Karlskrona, Sweden

[3]Mälardalen University, Västerås, Sweden

Correspondence
Efi Papatheocharous, RISE Research Institutes of Sweden, Kista, Sweden.
Email: efi.papatheocharous@ri.se

Funding information
Knowledge Foundation, Grant/Award Number: 20140218

Abstract

During software evolution, decisions related to components' origin or source significantly impact the quality properties of the product and development metrics such as cost, time to market, ease of maintenance, and further evolution. Thus, such decisions should ideally be supported by evidence, i.e., using previous experiences and information from different sources, even own previous experiences. A hindering factor to such reuse of previous experiences is that these decisions are highly context-dependent and it is difficult to identify when previous experiences come from sufficiently similar contexts to be useful in a current setting. Conversely, when documenting a decision (as a decision experience), it is difficult to know which context factors will be most beneficial when reusing the experience in the future. An interview study is performed to identify a list of context factors that are perceived to be most important by practitioners when using experiences to support decision-making for component sourcing, using a specific scenario with alternative sources of experiences. We observed that the further away (from a company or an interviewee) the experience evidence is, as is the case for online experiences, the more context factors are perceived as important by practitioners to make use of the experience. Furthermore, we discuss and identify further research to make this type of decision-making more evidence-based.

KEYWORDS
components off-the-shelf, context factors, decision experience, decision-making, experience source, in-house, open-source software

1 | INTRODUCTION

Decisions related to choosing components (e.g., in-house development, components off-the-shelf (COTS), or open source) when developing software-intensive systems are highly context-dependent. What worked in one case in a software development organisation with a particular product, or business model, or the customer may not work as well in another organisation with different characteristics because the context is different. Therefore, decision-makers need to understand the context in which a decision related to choosing software components was made to

determine if the lessons learned (aka. decision experience) can be reused. Context is defined by the Oxford Dictionary as "the circumstances that form the setting for an event, statement, or idea, and in terms of which it can be fully understood".[1] According to Petersen et al., the context provides the ramification of the studied phenomenon.[2] For example, when conducting an industrial study, the context of the company comprises, among others, factors such as the company size, development practices, and staff experience. We refer to these factors as being *context factors*.

Concerning architectural decisions related to choosing components, some sourcing decisions could benefit only particular products in a certain domain or company, for example, because of security, technology/vendor lock-in, or scalability requirements. Therefore, considering the evidence around a particular decision, understanding a decision's context factors and constraints that played an important role when making the decision becomes critical. It is essential when one tries to benefit from another decision already made, firstly, to understand to what extent the experiences about that particular decision are transferable to the current decision case, which includes understanding how similar the two contexts are and if any critical context factors are different. The focus of this paper is not on identifying factors considered in choosing components, but rather on identifying *context factors* to determine the similarity between decisions. In other words, this paper is about eliciting important context factors that would make a decision relevant (due to similarity) to a decision-makers' context when choosing between software components.

Experiences form valuable evidence that can be, in some cases, (re)used for improved decision-making in software engineering. In 2004, evidence-based software engineering (EBSE) was coined as a concept by Kitchenham et al.[3] and further directed towards practice by Dybå et al.[4] The context in which the evidence is obtained remains a critical aspect, which is challenging to document.[2] In particular, three aspects of context factors may be distinguished as essential: documentation of context factors, context factors perceived as important, and genuinely significant context factors. A brief description of these aspects follows, although we focus on the second aspect concerning contextual factors perceived as important by practitioners in this article. The reasoning for focusing on the second aspect is that limited literature exists to the best of our knowledge, and in particular, with a connection to decision-making for component-based software engineering. The limited related research we found (discussed in Section 2) refers to the first aspect, while the second aspect needs to precede any study related to the third aspect.

The first aspect is to *document contextual information* to encourage the transformation of software engineering into an evidence-based discipline. Several efforts have been made in this direction. Petersen and Wohlin[5] present a checklist for context factors to consider when documenting empirical studies. Petersen et al.[2] use a proposed context checklist to classify practitioners' context and document the context of case studies. Clarke and O'Connor[6] propose a framework for situational factors for software development processes. Dybå et al.[7,8] highlight four perspectives important to consider when capturing and trying to understand the importance of context: "what works for whom, where, when, and why." A key challenge is that any such listing of context factors quickly grows very large, and hence, there is a need to understand which factors stand out in different situations. This takes us to the second aspect.

The second aspect is to understand what *context factors are perceived as important* in a given situation. With limited effort available for documenting context, a selection must be made on what context factors to include and at what level of detail, without knowing when and for what purpose the information will be used in the future. Documenting every context factor that potentially could be relevant in some future scenario is prohibitively costly, especially because the person documenting a decision and its context is typically not the one who will benefit from it later. Any insight into what context factors are generally considered important helps with the prioritisation of what to document. Knowledge or assumptions about what type of scenarios the context will be used in provide additional support because it is likely that different context factors are perceived as important depending on the scenario.

The third aspect concerns *genuinely significant context factors*, for example, those that contribute to cause–effect relationships. This aspect is essential, in the long run, for practitioners if they are willing to use previous experiences. It is crucial to understand which of the context factors perceived as important are vital and which are not. The key challenge is that it is difficult to agree upon and identify generalisable comprehensive usage scenarios of previous experiences for practitioners, which can be integrated into their practical work. Moreover, experiences that can form significant evidence are difficult to access and use.

The primary aim of this article is to make a valuable contribution through an initial investigation into which context factors practitioners perceive to be essential when considering decision experiences of software component reuse. This refers to the second aspect, as described above. That is, for which context cases, is a decision experience relevant and where should a practitioner look to find such an experience? We use the term "experience sources" to distinguish the various alternative sources or channels developers use to access prior experiences of software components reuse. These sources for example might be sources of experiences of similar cases within the same company or from cases found on the web, in published articles (peer-reviewed or nonscientific literature). These sources should primarily involve experiences and decisions undertaken by individuals or organisations in prior instances where they were in search of software components for reuse. The sources of these experiences should exhibit a level of generality to be relevant in other scenarios where insights from such experiences may be potentially beneficial.

To the best of our knowledge, this is the first such study. In our view, this study opens the path to research for making architectural decisions concerning sourcing options of components for software development more evidence-based. The work is a continuation of the research conducted by Petersen et al.,[2] where the focus was on the documentation of contextual information (the first aspect above). This article explores and provides a list of context factors perceived to be important when reusing experiences from prior decision-making cases related to software component selection, as these were suggested and discussed with practitioners having extensive experience in carrying out such decisions. The

primary value to practitioners is an identified set of context factors that are perceived as most important for reusing experiences from cases of the past, whether these are internal cases (i.e., within the same context) or external cases (i.e., outside a particular context). This can be used as a checklist of the most important context factors guiding a decision and when documenting the context information of a particular decision. The decision is thus supported by empirical data and relevant context factors. The decision thus becomes more transparent and replicable. Such decisions are valuable to others, as they provide a clear relation to the context factors, a clearer rationale and better-informed decision-making by offering insights into similar previous experiences and a basis for future decisions. For researchers, the primary value is information about what context factors to describe, for example, in case studies, to make it easier for practitioners to judge how applicable the research results can be in their particular context.

The remainder of the article is structured as follows Section 2 presents related work. The design of the interview study is presented in Section 3. The results of the interview study are presented in Section 4. Section 5 presents an analysis of the results, and then Section 6 provides a discussion of the findings. Before concluding the article, the validity threats are discussed in Section 7. Finally, the conclusions are presented in Section 8.

2 | RELATED WORK

To make informed decisions concerning the selection of software components, including decisions about their origin or source, experiences and evidence from different sources using the components can be quite helpful. Thus, EBSE[3,4] is an essential concept to support informed decision-making. Furthermore, the evidence needs to be credible, as discussed by Wohlin and Rainer.[9] They discuss challenges and provide recommendations to move towards more EBSE. Moreover, Wohlin et al.[10] provide a roadmap for research related to decision-making to select assets in composite software. One vital aspect highlighted in the roadmap to enable evidence-based decision-making is the context. The importance of context is also argued by Briand et al.[11] when they put forward the need for context-driven software engineering research. Furthermore, Basili et al.[12] continue the discussion along the same lines by stressing the importance of context. Tang and Lau[13] point out the need for software architects to share "everything" related to their design decisions and reasoning, so that architecture reviewers could track and trace those design decisions easily, and then, reviewers would have the knowledge to find design issues.

Reusing experiences is at the core of the concept of the experience factory proposed by Basili et al.[14] Basili et al.[14] propose the support of software engineering by using direct feedback from analysing and synthesising all kinds of experiences gathered from projects, independent of the development organisation. However, it comes with challenges given that software engineers have beliefs that may not agree with evidence.[15] Passos et al.[16] have investigated the origin and impact on software development team practices. As it turns out, the perceptions of software engineers may not be aligned with existing evidence. However, we hypothesise that it may be a stepping stone towards getting practitioners to reuse experiences more in the future, for example, to improve decision-making. It is the case because they would not reuse experiences unless the context factors of their beliefs or perceptions match those being available in relation to the experiences. In the long run, it is vital to identify the context factors that connect evidence to specific cases in industry practice.

Understanding what context factors are perceived as important in a given situation is essential for industry–academia collaboration.[17] Understanding the most significant context factors for a company is an integral part of the problem formulation phase of co-production. Moreover, context factors influence the process of creating a candidate solution and greatly determine the suitability of the proposed solution.[18,19]

When selecting components, decision-makers have different development options to choose from, such as in-house development, COTS, and open source. Badampudi et al.[20] present a systematic literature review to identify the decision factors that influence the selection between different development options. One of the results indicates that one option is often more suitable than another, depending on the context. For example, the availability of technical support provided by COTS vendors may depend on the size of the organisation using the component—because larger organisations may have more influence on licence negotiations than smaller ones.[20]

After selecting the development option(s), practitioners may need to choose specific components from the development options, such as COTS. Li et al.[21] describe how COTS components are selected (and evaluated) via direct assessment processes (based on requirements satisfaction) or based on domain models (context as defined in our study, in some way). The latter involves mapping component modules to applicable domains. Their focus is on customising the COTS development process for specific contexts, and they identify that it is difficult to determine the precise type of a project context. Continuing their previous work, Li et al.[22] discuss component selection and highlight the need for improved knowledge management around decision-making, mostly internally in the companies, and raise the need for external channels to share and communicate knowledge between organisations. Client recommendations or other sources of evidence, literature or discussion boards, prototypes, and specific evaluations should also be used in the selection process. They highlight the need to avoid depending on only individual persons' viewpoints, which can be subjective, and build more formal selection processes. More formal processes exist, as highlighted by the authors, only when products relate to specific industrial standards, for example, safety standards. They conclude with the need to be more precise on the assumptions and contexts of the components selected. Li et al.[23] carried out a survey among developers and elicited factors and metrics to assess open-source software (OSS). They identify a set of eight main factors and 74 sub-factors, together with 170 related metrics that companies can use to select

OSS to be integrated into their software projects. In their work, the main factors identified are used to assess OSS; however, no explicit definitions of the main factors are given. They instead recommend the use of the list of factors selected as a checklist during the selection of OSS.

Several authors have addressed the challenges of capturing context. Petersen and Wohlin[5] describe the context in an onion model with different layers. They divide the inner layer of context factors into product, processes, people, and practices. The next layer is the organisation layer, and finally, the most outer layer is the market. Context elements are perceived not only as integral to the solution but also as impacting one another. The presentation in the article is a preliminary step to raise awareness of examples of elements that may be essential to document in context descriptions. The context model is intended as a checklist, and it should not be viewed as an exhaustive listing where everything needs to be documented. Clarke and O'Connor[6] take it one step further and propose a framework of 170 situational factors for software development processes divided into business, application, management, requirements, technical, personal, operational, and organization.

Dybå et al.[7,8] highlight four perspectives that are important to consider when capturing and trying to understand the importance of context: "what works for whom, where, when, and why." In contrast, Kirk and MacDonell[24] investigate the theoretical basis for software context with the objective of tailoring software practice. They classify context factors from the literature to try to propose a framework for operationalisation. However, a pilot study exposed many factors that are impossible to classify. They illustrate the aspects relating to tactical factors in their mapping for understanding software practice.

Bi et al.[25] in their study of the use of architecture patterns, the consideration of quality attributes and design contexts in software design, state that design context is not specified explicitly as requirements. Stakeholders do neither make contextual knowledge explicit nor document them in a systematic way, even though design contexts are conditions that can directly influence design decisions. Bedjeti et al.[26] identify four context categories of the viewpoint (i.e., platform context, user context, application context, and organisational context); then, they evaluated and refined the viewpoint in two case studies. Harper and Zheng[27] propose a systematic process for identifying and documenting design context to support architectural decision-making. They also propose a framework in terms of who, what, when, where, why, and how to explore the design contexts of a system and its environment.

Cartaxo et al.[28] conduct a systematic literature review on the mechanisms to characterise context in empirical software engineering studies. They also emphasise that the context could be everything, and more research is needed to identify context factors that are perceived as important for practitioners. They identify and synthesise findings supporting the documentation and characterization of context for empirical studies in software engineering. Furthermore, they suggest describing the context as "The set of information that is not the main interest of an empirical research" (in other words, they are not the study's interventions), but they have an influence on the study's results.

In our previous work,[29] we have investigated the representation of context information. A hierarchical context model was proposed to allow for trade-offs between the overhead of specifying detailed context information and the value of this information in reuse situations later on and permitting different aspects of the context to be elaborated in other cases.[29] The context model[2] was evaluated concerning understandability from the perspective of practitioners using it to capture their context, which complements the current study where we address the perceived usefulness of different context aspects for experience reuse.

Engineers use information that they trust.[30] The consequence of this can be that some engineers seek information from within the organisation (close by co-workers) whereas others trust more online sources. However, the cost of obtaining information is also considered important by engineers, and because many of them work in remote teams, the cost of getting information online is significantly lower than for meeting face-to-face. The advent of Internet discussion forums has also changed the behaviour of information seekers as they could reach more diverse opinions. Hertzum[30] concludes that engineers are looking for information that is (1) accessible in a way that enables the engineer to form a perception of its quality and (2) perceived to be of sufficiently high quality. The preference for a channel and the principle of least effort are among the two theoretical constructs in information searching.[31] Freund[32] derived contextual factors and conditioning variables that impact information source selection among software engineers. Borg et al.[33] studied information-seeking behaviour in change impact analysis, while our study looks into the component reuse scenario. Motivated by the discussion concerning information seeking, the study presented here investigates which context factors are most essential when reusing experiences from others.

3 | METHOD

We designed an interview study to investigate which context factors are perceived as important by practitioners. The scenario, discussed in the interviews, focuses on reusing software components from various sources and is detailed in the subsection that follows.

3.1 | Scenario

All software components have embedded characteristics, also called properties. Most quality aspects (e.g., usability, performance, maintainability, and security) are embedded properties of software components. Next, each component is applied in a context that may differ more or less

significantly. Therefore, when looking at previous decisions, it becomes essential to evaluate which cases are sufficiently similar to the current setting based on the context factors. Some context factors are documented, and some remain implicit and not documented. Both documented and non-documented context factors may include context factors perceived as important by the practitioners. The scenario that we investigate in this article focuses on understanding what context factors practitioners perceive as important when identifying previous similar cases to draw experiences from. Finally, some of the context factors are probably genuinely significant, that is, contributing to cause–effect relationships. Due to the exploratory nature of this study, we are unable to study the true significance of the context factors perceived as important by our respondents.

We specified the scenario used in the study, in a well-defined manner, making sure it covers relevant aspects of sourcing decision options when choosing software components. We carried out a preliminary review with a small reference industrial group, with whom we carefully reviewed the scenario description, ensuring it is concise and relevant, and that it is not missing any relevant important aspects. We also covered whether the scenario is relevant when making such a decision and relevant to the intended participants (i.e., it is a scenario they can relate to). We ensured that the sources/scenarios used are realistic and cover representative conditions and situations, thus increasing the closeness to real-world options for experience sources. The choice of the experience sources/scenarios created a clear and focused context to base the study on. It consists of a generic list of experience sources/scenarios that cover all possible real-world options, even though in reality not all the scenarios would be applicable in all cases. However, the choice ensures a generic-enough basis, covering most of the real-world situations. This enhances the generalisability of the research findings, making them applicable to a wider context. However, we agree that it may be challenging to replicate real-world situations precisely, that is why the closest to real-world conditions was to discuss with industry experts the generic hypothetical scenarios. The hypothetical scenarios offer a practical and feasible way to carry out the research without the compromise of real-life constraints. For example, using the scenarios allowed researchers to avoid touching on sensitive or confidential company information.

Figure 1 presents an illustration of the scenario investigated in this article. The scenario was described to the interviewees as follows: You (interviewee) are about to evolve your existing product and have identified some different options for doing so, in the form of software components. You realise that you have different options, particularly regarding the components supporting your intended added value with the new version of the product, rather than being a core value by themselves. It is assumed that the core value comes from software developed by the organisation owning the product. You are prepared to consider different options for these "supportive" components: in-house development, in-house reuse, open source, COTS, and outsourcing to someone else to develop it for you. To ensure that you make an informed decision, you want to build on experience from "similar" previous cases. When looking for "similar" cases, you realise that they can be found in different experience sources (S):

S1. Your own experiences from developing software systems (usually implicitly used)
S2. Company internal
    (a) Close to you either geographically (easy to talk to others) or close concerning a similar type of product
    (b) Other experiences within the company
S3. Other companies where you have access to their experiences (e.g., close collaborators or good personal contacts)

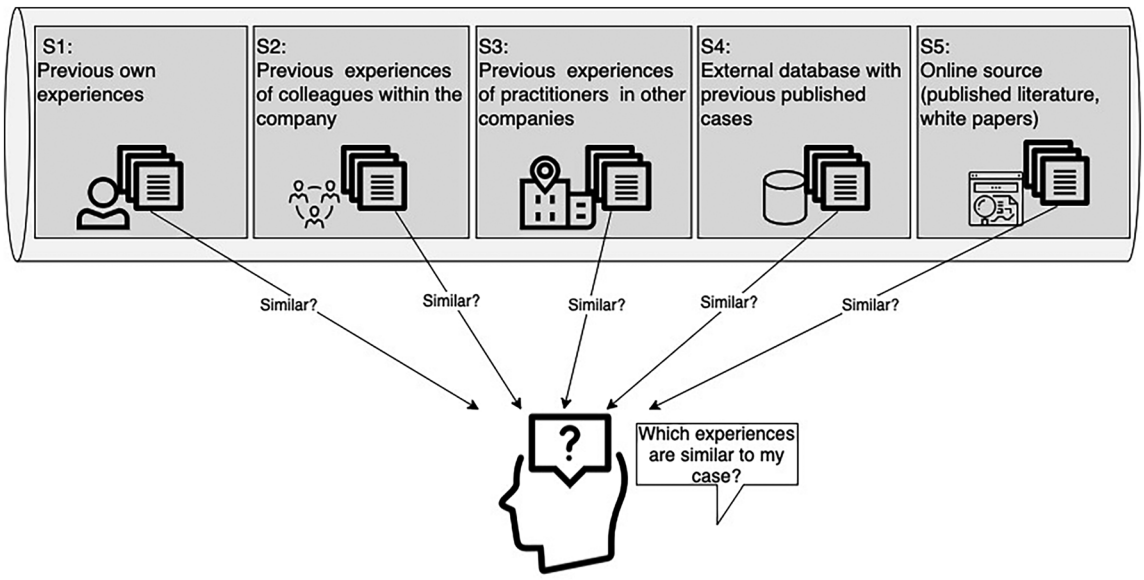

FIGURE 1 The illustration of the scenario investigated in this work.

S4. Open external databases with published cases
S5. Online sources
   (a) Various online sources such as developer forums (e.g., Stack Overflow), blogs, newsletters, and social networks (e.g., LinkedIn)
   (b) Published literature (both research and more practitioner-oriented publications), including books, journal and magazine articles, and conference papers.

All the above-listed experience sources (S1–S5—as illustrated in Figure 1) are covered sequentially in the interviews and are associated with context factors. It is assumed that different context factors may be perceived as important with respect to the various sources of experiences investigated. For example, it is hypothesised (and hence the order of the experience sources) that more context factors have to be considered as the experience sources are perceived to be further away from the actual case under consideration. Whether this hypothesis is correct or not is one of the outcomes of the research study.

3.2 | Research questions

Our primary research question was concerned with which context factors are essential when selecting components in the context of software-intensive system development. Based on the general objective of identifying context factors that are perceived as important and the scenario presented in Section 3.1. The following specific research questions are addressed in the study:

- *RQ1: Which context factors are perceived as important by the practitioners when selecting components?*
- *RQ2: Are there any differences in the perceived importance of context factors between the experience sources covered in the scenario?*

3.3 | Interviewees

The interviewees were selected based on their experience in decision-making using software components for product development. The interviewees carry out or are involved daily or monthly (at least) in decision-making using software components in their respective companies and could relate to the examined scenario. The interviews were conducted with persons who could discuss context factors and relate to the scenario of reusing experiences from previous cases from different potential sources, for example, from other companies or open-source projects, to enable generalisability. The selection of the interviewees was made based on convenience sampling from the researchers' industrial network.[34] The objective was to cover different roles involved in the decision-making in these situations at different companies. The roles included managers, architects, and technical leads. For each participant, we kept track of their experience concerning decision-making for software components, the name of the company and the interviewee's role. The interviewee's role in the company is, in the majority of the cases, related to technical lead role, particularly coordinating and contributing to the company's intake process of component software. For example, software architects, solutions architects, and lead developer roles were interviewed. In addition, the interviewees were selected based on their long experience in such roles. Interviewees were promised anonymity, but we kept track of the names and contact details to be able to follow up with the participants and provide the results to them. A form was created for this purpose (see Appendix A) to collect their feedback about our interpretation of the most important context factors they mentioned in the interviews. Furthermore, when it comes to the companies, we describe the company in general terms to provide some context for our study and the interviews in Section 4. We collected the following context information: the size of the unit being considered (which may be the whole company, or a specific business unit or something else), application domain, type of system (including critical non-functional properties, e.g., performance, safety, and security), and software development process/methods. The most critical non-functional requirements (NFRs) collected provide an environment description of the particular company's or product's context and were collected as background information, at the end of the interviews.

3.4 | Interviews

Four researchers conducted the interviews. All interviews were recorded. The researchers carried out three interviews each and transcribed the collected data from the recorded interviews within a few days after. Next, follow-up discussions with interviewees were conducted, when needed, which was the case in a majority of the twelve interviews.

The interviews started by asking the Interviewee (I) to provide some personal information regarding their role and personal work experience. The interview continued by asking the interviewee about factors related to their context (i.e., own software-intensive system and organisation)

and the components' origin (where typically the different components they consider come from, e.g., open-source communities). The interviewee was given the scenario description and was asked to consider different perspectives deemed essential in their decision-making process.

All five experience sources listed above in the scenario (S1–S5) were covered in the interviews in sequential order. The sub-items for S2 and S5 were used not only to examine if there were differences for different subcategories but also to explain the specifics of each experience source. For example, in S5, if there was a difference between more practice-oriented online sources versus more research-oriented online sources, including research publications.

The interviewees were asked to provide essential context factors they would consider when reusing information from a particular experience source to make a product evolution decision in their context. When needed, to make the discussion more tangible, a recent relevant decision of component sourcing was described by the interviewee from their practical experience. Examples were used, for example, when reading up on experiences found in online sources, which context factors make the experiences relevant to your context? Thus, these experiences may be considered as relevant input to your decision-making. When listing the factors, we asked the interviewees to prioritise and highlight the top three context factors that they would consider for each of the five experience sources above (S1–S5). Once done, the interviewee was asked if she/he wanted to make any changes to the context factors (things may pop up in the interviewee's mind as the interview proceeds). At the end of the interview, the interviewer went through the checklist of the top-level perspectives found in the context model presented in Carlson et al.[29] That is, reminding the interviewee to consider organization, product, processes/methods, market/business, and stakeholders related to the decision.

Towards the end, interviewees were asked to provide some more detailed information about their own context and to mention any additional factor that the interviewee considered important. This is captured at the end of the interview to cover the context of the interviewee/company while ensuring that the questions about the context of the interviewee do not bias the interviewee's answers concerning the most important context factors in the analysed scenario.

3.5 | Analysis method

The analysis method used in our study is best described as content analysis.[35] Table 1 depicts an example of analysis and extraction for Interview 2 and Experience Source 2 (S2).

During the interviews, we discussed context factors for each of the experience sources (S1–S5) in the scenario. Each researcher who conducted an interview extracted statements from the interviews' transcripts for all experience sources (S1–S5). The resulting list of statements was inserted into a Google sheet, including additional notes and answers to follow-up questions (Steps 1 and 2 in Table 1). From the statements, the same researcher coded a list of context factors, tagged with the interview ID and the experience source ID, to allow tracing back to the interview and mapping to the experience sources intended to analyse within the scenario. In the next step, a second researcher reviewed the interview recording and output (Steps 1 and 2) collected by the researcher who carried out the interview. The second researcher reviewed the responses and the coded context factors and left comments when a discussion was needed as shown in Step 4 in Table 1. The reviews were done in a round-robin way to increase consistency between reviewers and hence interviews. If questions arose or differences concerning identifying the context factors, these were discussed among the two researchers, until an agreement was reached. Based on the review, the recording and output were revisited and context factors were added when they were missed by the first researcher (see Step 5 in Table 1). As already mentioned, we

TABLE 1 Example of data extraction and review related to Experience Source 2 from Interview 2.

Step	Experience Source 2	What context factors would you consider when identifying similar cases from YOUR company?
1	Extracting responses (verbatim)	If someone at the other end of the company has used a tool then that is fine but that does not mean I am going to push for that technology because *I have not used it*. We need to find out if they are using it for the *same purpose*. The other team members might be working with the technology for a long time. It is good to know if it takes a long time to *learn the technology*. We are only developing in Java. It is not the best programming language, but this is what we use. We need to check if they use the same *programming language*. I need to investigate if it fits well in my context. I also need to look if the component that was used has enough *technical support available*.
2	Adding interviewer notes	The interviewee mentioned that if the context is different, then the decision is not reusable. A follow-up question was asked to clarify what is meant by a different context.
3	Context factor coding	Programming language, team competence, functional use, and availability of support
4	Internal review comment	Why not elicit also "*Stakeholder experience*" as it is mentioned, "[…] because I have not used it"?
5	Revised context factor coding	Programming language, stakeholder experience, team competence, functional use, and availability of support
6	Validation by interviewee	Stakeholder experience with respect to the product and team competence—if we have the knowledge already, then that might be worth more than a better-suited option that we know nothing about.

used an interview validation form (see Appendix A) to validate the collected data. The revised context factors along with the interviewee responses were added to the validation form. The form was discussed with the interviewees to clarify that the context factors collected were indeed the context factors that were intended to be brought forward by the interviewees. In addition, the interviewees had the possibility to add further comments. As seen in Table 1 Step 6, the interviewee provided further explanation for the "stakeholder experience" context factor.

3.5.1 | Organizing similar context factors

After performing all 12 interviews and reviewing all extracted quotes, we performed additional analysis to find common names for similar factors described by several interviewees. The revised context factors were analysed by two authors in a round-robin manner (cf. Section 5) until an agreement was reached concerning the naming. We used the responses from the interview transcripts and the validation forms to group and rename the context factors. In the example shown in Table 1, a new context factor—*stakeholder experience* was discussed to be added based on the review comment in Step 4 in Table 1. The new factor was renamed to team experience, whereas descriptions from other interviews were merged to form its description (i.e., "*[…] finding the right person who can help you solve the problem within the problem space […] finding a person who had previously faced a similar problem with the one I am trying to solve*" [Interview 4, S2]).

The final output is a list of context factors for each of the five experience sources (S1–S5) per interview. These, together with the rest of the data collected from the interviewee/company, were reviewed by all researchers and are presented in the results section. The top five most common context factors (mentioned in most experience sources and interviews) are also described with the associated descriptions extracted from the interviews.

4 | RESULTS

4.1 | Overview of interviews and context information

To put the results into context, Tables 2 and 3 provide an overview of the characterisation of the environment of the 12 interviews. The interviews are characterised by the company's application domain, the size of the company, the size of the development unit the interviewee refers to, the role of the interviewee within the company, the product type, the development process/method used, and the most critical NFRs. The NFRs relate to the particular domain (or environment) of the company or products and do not relate to the context factors analysed in the scenario. Table 4 indicates the total years of experience of the interviewees in their current role involved in decision-making for component intake and their

TABLE 2 Overview of the interviews.

Interview	Domain	Company size	Development unit size	Role	Product type
Interview 1	Telecom	100.000	Large	3rd-Party Provider (3PP) Coordinator	Embedded systems
Interview 2	Telecom	100.000	Large	Software architect	Embedded systems
Interview 3	Telecom	100.000	Large	Solutions architect	Embedded systems
Interview 4	Finance	1.500	55	Domain architect	Banking systems
Interview 5	Automotive	38.000	120	Architect at design stage	Engine systems
Interview 6	Telecom	100.000	300	Software architect	Radio base stations
Interview 7	Finance	1.000	2	Performance analyst	Testing frameworks
Interview 8	Security	50	7	Senior developer	Security analysis
Interview 9	Automotive	200		Product manager	Embedded systems
Interview 10	Automotive	38.000	60	Senior system architect and lead developer	Embedded systems
Interview 11	Agriculture	20	6	CTO and co-founder	SaaS—support selling and buying grain and other farming products
Interview 12	Automotive aftermarket	30	5	CTO, founder, and lead developer	Supporting car repair process after accident, paint and bodywork fixes

TABLE 3 Overview of the interviews (cont.).

Interview	Development methodology	Most critical NFRs (using interviewees' wording)
Interview 1	Agile, Continuous Delivery	Performance
Interview 2	Agile, Continuous Delivery	Availability, Serviceability, Scalability, and Performance
Interview 3	Agile, Continuous Delivery	Scalability and Performance
Interview 4	Agile, up to the team	Security, Meeting regulations, and Time-to-market
Interview 5	Agile, modified, long development cycles	Performance, Functionality–suitability–feasibility based on the manufacturing restrictions, and Quality (i.e., life-time (longevity), safety, light-weight for sustainability purposes)
Interview 6	Agile	Performance (throughput and latency), Scalability for different scenarios, i.e., small and large user bases, and Modular architecture
Interview 7	Agile, ad-hoc	Performance
Interview 8	Agile	Performance, Scalability, and Price
Interview 9		Safety and Performance (timing)
Interview 10	Undergoing Agile transformation	Safety, User experience, and Security
Interview 11	Agile process (close to SCRUM) but modified	Usability—minimal workflows, Reliability, and Ubiquity
Interview 12	Agile with a product owner, no SCRUM master role	Flexibility (tailoring the software to customer needs), Usability, and Portability

TABLE 4 Interviewees years of experience in current role and overall.

Interview	Experience (in current role)	Overall experience
Interview 1	10	25
Interview 2	3	20
Interview 3	11	15
Interview 4	4	10
Interview 5	3	4
Interview 6	4	14
Interview 7	2.5	2.5
Interview 8	10	18
Interview 9		19
Interview 10	3	15
Interview 11	5	35
Interview 12	15	20

overall years of experience (in other roles or other companies). The interviewees in total have 70.5 years of experience in their current role and 197.5 years overall experience (on average 6.4 and 16.5 years, respectively).

Interviews 1–3 and 6 were collected from individuals in different units of a multinational networking and telecommunications company, offering services, software, and infrastructure. Interviews 5 and 10 involve interviewees from another multinational company producing cars and focusing on aspects such as security, services and software. The company is undergoing a transformation into a software-intensive company since active safety, and electric cars start to dominate the product-line strategy. Interview 4 represents a bank that provides online financial services such as payment solutions for online storefronts, direct payments, post-purchase payments, and more. Interview 7 is a multinational company developing software for financial institutions, but the specific software system considered is an in-house developed framework for performance testing of financial applications. Interview 8 comes from a multinational company providing security analysis software and services. Interview 9 involves a multinational company developing display computers and communication modules, mainly for the automotive sector. Interview 11 relates to a company offering a software platform for matching supply and demand in the grain industry. The platform helps grain growers plan their production and the buyers to know what supplies will be available in what region and in which quantity. Interview 12 comes from a company that provides a software service that allows car repair shops to streamline their work. The software offers access to part inventory and helps to plan work for car mechanics and painters. It also connects with insurance companies for faster handling of cost coverage.

The interviews cover several different domains and product types. Company sizes ranging from very large international companies to SMEs. Moreover, many different roles have been included in the interviews, primarily architects but also, for example, developers and managers. The majority of the companies use some form of Agile development methodology. Performance is one of the most prominent quality properties identified in the environmental setting of many of the interviews. In the automotive domain (I5, I9, I10, and I12), safety and security are highlighted, and scalability is mentioned in several cases. We should clarify that the NFRs discussed here are different from the context factors collected from the scenario with respect to experience sources for the particular component sourcing decision-making scenario.

4.2 | Perceived important context factors (RQ1)

Table 5 summarises the context factors mentioned in Interviews 1 to 6 in the different experience sources (S) of the scenario, and Table 6 presents Interviews 7 to 12. The context factors appear in the order they were mentioned in relation to the order of the experience source covered (and within the same experience source as prioritized by the interviewees). The context factors' frequency order is shown in Table 7. Furthermore, the Interviews are described one at a time as follows.

In Interview 1, the interviewee considers using open-source components as frequently as possible. Initially, the licensing constraints were mentioned as an important context factor of the decision, giving the example of when a prior decision is taken outside the region of their own, this might mean that some constraints (e.g., legal regional restrictions) are not transparent, thus such a decision cannot be reused. In addition, complexity in terms of the system size and performance is an essential factor in all experience sources (S1–S5). The performance requirement on smaller systems might differ from larger systems and thus not considered relevant to the current decision. Moreover, when considering experiences in other companies (S3), the release cycle (frequencies of releases) of the company was perceived to be important. For example, a software provider company may release every 3 weeks, whereas an automotive company may have a longer release cycle. The differences in the release cycles may challenge the reusing of experiences. The interviewee mentioned that experiences from all experience sources (S1–S5) referring to cases where the system complexity, availability of support, licence and support costs, and licence models used (e.g., subscription-based, "buy-out", or perpetual-based models) would be relevant to them and can be reused.

In Interview 2, when considering the interviewee's own (S1) or colleagues' (S2) experiences, the most important context factors were the team's experience and competence. The team's knowledge and experience, for example, concerning technology, can affect the time to market. The decision can be reused if the team's competence level is similar. Furthermore, colleagues' experiences (S2) were considered relevant in decisions where the context factors such as functional use, programming language, and availability of support were the same. Similar to Interview 1, it is important to know the context of the previous decision by investigating context factors related to the component. For example, what kind of support was available for the component in the previous decision? Will the new component under investigation have the same support? If yes, then the previous decision context will be similar and thus the decision may be reused. Own experiences (S1) are mostly reused when the decision is made for the same project and the same team and when there are similar project "configurations" (i.e., we have a similar product type and team setup). When talking about external sources (S3–S5), the context factors such as application domain and NFRs such as scalability were perceived to be most important. For example, if a component was previously selected because of its good scalability and if scalability is not required in the current decision, then it may not be useful to reuse the decision. Moreover, product type was considered essential when considering their own internal and other companies' experiences (S1 and S3).

In Interview 3, the interviewee talked about a recent decision to choose an open-source tool and build a framework on top of it to automate testing. In this interview, functional use, team competence, product type, the technology used, and development process (such as the testing process) were considered most important in two experience sources, including own and external experience sources (i.e., combinations of S1–S3 and S5 were mentioned). For example, the competence of the team where the decision was taken should match the competence of the team where the decision is to be reused. Functional use was the most mentioned factor in S1, S2, and S5 experience sources. Certain context factors were only mentioned when discussing a particular experience source, such as time to market in S1, the company's geographical location, which influences the organizational culture in S2, and the application domain in S3. S4 was not perceived as a relevant source for relevant experiences for the interviewee.

In Interview 4, the interviewee described that the company typically works with in-house development (in-house reuse from an ecosystem of own-built services), uses and contributes with open source or buys COTS, and rarely outsources. The primary context factor mentioned for S1–S3 was functional use. For example, the prior decision's functional use should match the current decision's functional use to reuse the decision experience. Functional use was described as solving a similar functional requirement, or addressing a problem in the same problem space or looking for a similar architectural pattern. It was mentioned: "*Software engineers tend to go too far in building their own software [and end up] rebuilding something that already exists*." This means it is important to be able to find similar solutions and experiences from using them, instead of rebuilding them from scratch. Another important factor in S2, S3, and S5 was the existence of a person (or a community) with the corresponding expertise to solve a problem in a particular problem space. In the same interview (Interview 4), the application domain was perceived as important by the practitioner in S4, and finally, in S5, the maturity of the technology was most important. Other context factors were also mentioned to

TABLE 5 Context factors mentioned in Interviews 1 to 6.

Interview	Context Factors	Sources
Interview 1	Licensing constraints	S1–S5
	Complexity	S1–S5
	Availability of support	S1–S5
	Licence and support costs	S1–S5
	Licence model	S1–S5
	Release cycle	S3
Interview 2	Product type	S1, S3
	Team experience	S1, S2
	Team competence	S1, S2
	Functional use	S2
	Programming language	S2
	Availability of support	S2
	Application domain	S3–S5
	Scalability	S3, S4
Interview 3	Functional use	S1, S2, S5
	Product type	S1, S3
	Time to market	S1
	Team competence	S1, S2
	Geographical location (of the company)	S2
	Technology used	S2, S3
	Application domain	S3
	Development process	S3, S5
Interview 4	Functional use	S1–S3
	Scalability	S1, S2
	Time to market	S1, S2
	Complexity	S1, S2
	Team turnover	S1, S2
	Team experience	S2, S3
	Geographical location (of the person)	S2, S3
	Application domain	S4
	Market geography	S4
	Development process	S4
	Technological maturity	S5
	Product architecture	S5
	Availability of support	S5
	Licence model	S5
Interview 5	Product type	S1–S5
	Operational requirements	S1–S3
	Manufacturing restrictions	S1, S2
	Team experience	S2, S3
	Robustness	S3, S5
	Product architecture	S4, S5
	Conditions used in simulations	S5
	Software used in simulations	S5
Interview 6	Size of customer base	S1, S4
	Code quality	S1, S2

(Continues)

TABLE 5 (Continued)

Interview	Context Factors	Sources
	Scalability	S1, S2, S4
	Functional use	S1, S5
	Correctness	S1
	Availability of support	S2, S4
	Business relations (political)	S3
	Licence model	S3, S4
	Business agreements	S3
	Product type	S5
	Development process	S5

make the decision reusable, that is, scalability, time to market, complexity—related to product architecture, team turnover and experience, geographical location for S1–S3; market geography (where the product is sold) and development process for S4; and technological maturity, product architecture, and licence model for S5.

In Interview 5, the practitioner is involved in a lot of hardware design decisions along with some software system design. Many different departments are involved in the decision. Initially, the main factor perceived as important in all the experience sources for a decision experience to be relevant was the type of product developed. The prior experience's product type should match the current decision's product type to make the decision to be relevant. Electrical products would differ substantially from the "traditional" product-line products. Operational requirements were particularly essential in S1–S3, that is, how the product behaves during operation (e.g., how it withstands its workload) and the manufacturing constraints, that is, if the options considered at the assembly would work. In S3 and S5, results from "tests performed", that is, robustness was perceived as important, and "failures posted from usage or designs to avoid" were highlighted. In addition, context factors like the team experience for S2 and S3, product architecture for S4 and S5, and the conditions and the software used in the simulations (for S5) were identified.

In Interview 6, the product context was a radio base station. Code quality was one of the most important context factors emerging in S1 and S2, and the availability of support in S2 and S4. Thus, for a decision to be relevant, it would have to involve similar quality software and support than the one looking to have in the current context, to be able to reuse the decision. Support was expressed as "*having similar time zones*", "*willingness to provide support*", and the number of people or a community being available. In S1 and S5, the product type and functional use related to the product for which the decision was made were perceived as important. Additional context factors were mentioned, such as Customer base size (S1 and S4), Scalability (S1, S2, and S4), Correctness (tried and tested software with the least bugs and faults)—related to code quality in S1, Business relations and political issues (S3), Licence model (S3 and S4), Business agreements (S3), Product type and Development process (S5).

In Interview 7, the interviewee's focus was a test framework for performance analytics. In this interview, information about functional use and the technology used (i.e., whether it is the same technology) were mentioned as the most important context factors in a decision. Therefore, for a decision to be relevant for reuse in a current decision case the prior decisions' functional use and technology should match the current decision. Moreover, it was described that the key context factors do not change much between the experience sources, but the need to explicitly document them does. When reusing decision experiences from the same company or close collaborators (S2), much of the context is already implicitly known. However, for reuse from other companies (S3) or general repositories or research (S4), the same context factors need to be documented explicitly. For all experience sources (S1–S5) and in addition to the context factors mentioned above, the development process, sourcing alternatives, organisational size and resources, the market landscape, decision date (when the decision is taken), development cost, NFRs, product domain, and component costs were context factors mentioned as important. Additionally, in S2–S5, the decision goal, team size, and legislation, and finally, in S3–S5, the application domain and complexity were mentioned.

In Interview 8, the emphasised context factors in all experience sources (S1–S5) were related to key non-functional properties such as complexity and latency as well as the product domain. For a decision to be relevant in a current decision case, these factors need to match. Also, functional use, the degree of effect the decision has on the entire system and the effect over time were mentioned. In S2–S5, the team competence, product type, decision type, and other NFRs were important. The key context factors from the first experience source (S1) were found important also in the other experience sources (S2 onward). However, a couple of additional context factors were added in S3–S5, particularly when moving from the known context of the same company to the reuse of information from external sources. In S3–S5, the technology used (if the same technology is used) and the company's reputation (the company from which the decision is reused) in terms of competence/success are perceived as important. For S4, the overall prioritization of the criteria used in the two decisions (the documented one and the current one) and to what extent they are similar, and in S5 the decision date and the level of experience of the experience source were important. The latter related to ranking of the quality of the experience source, which is an entirely new aspect to investigate.

TABLE 6 Context factors mentioned in Interviews 7 to 12.

Interview	Context factors	Sources
Interview 7	Functional use	S1–S5
	Technology used	S1–S5
	Development process	S1–S5
	Sourcing alternatives	S1–S5
	Organisational size	S1–S5
	Organisation resources	S1–S5
	Market landscape	S1–S5
	Decision date	S1–S5
	Development cost	S1–S5
	NFRs	S1–S5
	Product domain	S1–S5
	Component cost	S1–S5
	Decision goal	S2–S5
	Team size	S2–S5
	Legislation	S2–S5
	Application domain	S3–S5
	Complexity	S3–S5
Interview 8	Complexity	S1-S5
	Latency	S1–S5
	Product domain	S1–S5
	Functional use	S1–S5
	Degree of effect on system	S1–S5
	Effect over time (of the decision)	S1–S5
	Team competence	S2–S5
	Product type	S2–S5
	Decision type	S2–S5
	NFRs	S2–S5
	Technology used	S3–S5
	Company competence	S3–S5
	Company success	S3–S5
	Decision criteria prioritisation	S4
	Decision date	S5
	Experience (of the experience source)	S5
Interview 9	Functional use	S1–S5
	Decision satisfaction	S1–S5
	Decision criteria	S1–S5
	Tests performed	S1–S5
	Stakeholders involved	S1–S5
	Decision date	S1–S5
	Hardware dependency	S1–S5
	Availability of support	S2–S5
	Correctness	S2–S5
	Safety level	S2–S5
	Application domain	S3–S5
	Credibility (of the experience source)	S3–S5
	Volume of products	S3–S5

(Continues)

TABLE 6 (Continued)

Interview	Context factors	Sources
	Certification	S3–S5
	Technology used	S5
Interview 10	Company success	S1
	Application domain	S1,S3
	Product type	S1,S2
	Credibility (of the experience source)	S2,S4,S5
	Quality (of the case)	S2
	Technological maturity	S5
Interview 11	Direct/indirect experience	S1,S2,S5
	Experience (of the experience source	S2
	Application domain	S3,S5
	Type of customer base	S3
	Size of customer base	S3
	Functional use	S3
	Experience duration	S4,S5
	Product size	S5
Interview 12	Direct/indirect experience	S1,S2,S4,S5
	Credibility (of the experience source)	S2,S4,S5
	Experience (of the experience source)	S2
	Application domain	S3–S5
	Community interest	S4,S5
	Experience duration	S5
	Product type	S5
	Product size	S5

In Interview 9, the same functional use was identified as a key factor between the prior decision and the current decision. That is, the functional use should match between the experience decision being considered and the current decision. Also, the existence of post–decision reflection about the level of satisfaction from the previous decision was identified as important. This is related to how good the decision was, judging it after some time has passed. It was also highlighted, both for internal (S2) and external (S3) reuse, that it is essential to know who was involved in the decision-making, in terms of roles and skills, and explicitly documenting who participated. Other important context factors mentioned for all experience sources (S1–S5) were the decision criteria used, tests performed, decision date, and whether there is some hardware dependency—related to product type. In S2–S5, the availability of support, correctness (code quality), and safety; in S3–S5, the application domain, credibility of the experience source, volume of products, and certification; and in S5, the technology used were mentioned.

In Interview 10, the interviewee talked about a transformation trend from component sources from COTS to open source and mentioned in relation to the reuse of decisions that the most important context factors of a decision are the application domain (S1 and S3), product type (S1 and S2), and credibility of the experience source (S2, S4, and S5) which dominated the discussed experience sources. These factors would need to match with the current decision context to make the discussed prior decision to be relevant. Our interviewee would consider reusing experiences from a similar domain and product type if they originate from a trusted source. For example, if a component was used in the automotive domain (same as the domain company the interviewee is working in), that experience is considered more valuable than a decision from a different domain (e.g., Telecom), even if the Telecom experience is more detailed than in the automotive experience. Moreover, the quality of the experience was perceived as important (in S2) and the technological maturity (in S5).

In Interview 11, outsourcing is rarely used because of intellectual property ownership issues. The ease of use for the component is the most important context factor, and when it comes to reusing experiences, the dominant aspect mentioned was the experience duration (in S4 and S5) and the nature of experience (meaning if it is a direct experience of the person sharing it or if it is indirectly obtained, e.g., via discussions and opinions) in S1, S2, and S5. For example, a person sharing the experience may have been involved in selecting a component and later integrating it into the product, maintaining changes, updating to new versions, and checking compatibility with the product's general architecture guidelines. The experience of the provider of the information was important in S2, whereas the application domain (both in S3 and S5). In external sources (S3), the type and size of the customer base and functional use were mentioned. The main reason here is that the product has a rather small

TABLE 7 Context factors for different experience sources.

Context factors	S1	S2	S3	S4	S5	Total
Functional use	6	6	5	3	6	**26**
Product type	4	4	4	2	5	**19**
Application domain	1		7	5	5	**18**
Complexity	3	3	3	3	3	**15**
Availability of support	1	4	2	3	3	**13**
Decision date	2	2	2	2	3	**11**
Technology used	1	2	3	2	3	**11**
Product domain	2	2	2	2	2	**10**
Development process	1	1	2	2	3	**9**
Non-functional requirements	1	2	2	2	2	**9**
Credibility (of the experience source)		2	1	3	3	**9**
Team competence	2	3	1	1	1	**8**
Licence model	1	1	2	2	2	**8**
Scalability	2	2	1	2		**7**
Team experience	1	3	2		1	**7**
Direct/indirect experience	2	2		1	2	**7**
Total of frequently mentioned context factors (top 16 above)	**30**	**39**	**39**	**35**	**44**	**187**
Other context factors	*25*	*36*	*35*	*33*	*40*	*169*
Total of all context factors	**55**	**75**	**74**	**68**	**84**	**356**
Count of all context factors	**39**	**48**	**50**	**48**	**51**	**236**

number of customers (hundreds), and these are mostly large farms so the experiences from applying a component for millions of users are not so relevant and valuable here, according to the interviewee. The product size was mentioned in S5.

In Interview 12, the nature of experience (direct or indirect) also dominates the experience sources discussed and the relevance of a prior decision to a current decision. One explanation here is that the CEO is very technical and worked as a developer, and lead architect for several years in several companies, including the current one. Therefore, she values first-hand experience more than second-hand opinions. Moreover, the reputation of the provider of the experience and application domain that the experience was gathered from were identified as important context factors across the experience sources. The credibility of the provider of the experience and the degree of their experience are the most important context factors. This relates to the meritocracy principle followed by the CEO. Moreover, the application domain (in S3–S5), the community interest (S4 and S5), the product type, and product size (S5) were among the highlighted context factors. The product analysed in interview 12 has a very specialised (and therefore) limited audience and is integrated with the insurance policies of the country it is operating in. Therefore, experiences from using a component globally or for millions of B2B customers are not as valuable and useful here.

4.3 | Differences in context factors between the experience sources in the scenario (RQ2)

From Tables 5 and 6, we can conclude that the context factors are more similar within Interviews 1, 7, 8, and 9 between the different experience sources (they agree in many interviews in all experience sources, i.e., S1–S5) compared with the remaining interviews. For the rest of the interviews, we do not see a clear pattern of similarity between the context factors and experience sources within the scenario. In Interview 4, half of the factors are mentioned in S1–S3, and completely new sets of context factors appear in S4 and S5, respectively. An agreement of the context factors mentioned for S1 and S2 is found in Interview 2, and different context factors are mentioned for S3–S5 (with product type being the exception, appearing both in S1 and S3). The situation is much more mixed for Interviews 3, 5, 6, 10, 11, and 12.

Table 7 shows the most frequently mentioned context factors in each of the experience sources (S1–S5). "Most frequently" refers to context factors mentioned in more than one interview and in more than two experience sources. The remaining context factors that are mentioned at most by two interviewees for not more than one experience source are listed as "Other context factors". These are shown in italics in Table 7. The second to sixth columns present the number of times a context factor is mentioned in the respective experience source. At the end of Table 7 (and in bold) the total count of the frequently mentioned context factors (top 16 context factors), the total sum of all context factors and the count of all context factors (how many different factors) are shown. The reader is referred to see Appendix B for the complete list of context

factors mentioned for the different experience sources of the scenario. We provide the number of times a context factor is mentioned for each experience source, including the number of interviewees mentioning a context factor. The last row includes the count of the context factors mentioned for each experience source without considering the number of interviewees. For example, if 10 context factors are mentioned when discussing an experience source, and each of those context factors is mentioned by two interviewees, the total would be 20, whereas the count of context factors would be 10.

The count of context factors indicates that 12 additional context factors are considered in S5 compared with S1. Therefore, more context factors are considered for online sources, which is further away from their own experiences. However, there is not much difference in the number of context factors mentioned in S2–S4.

5 | ANALYSIS

Concerning RQ1, in relation to which context factors are perceived as important when selecting components in software-intensive systems development, several context factors were common among the interviews and the different experience sources in the scenario investigated. We discuss the top five most frequently mentioned context factors in the interviews, including highlighting some quotes from the interviews.

1. Functional use: The purpose of the component, the "*functionality of the component in the previous decision, because we prefer to reuse a solution even if it might not be optimal, to reduce the number of, for example, external libraries we depend on*" (Interview 8). "*We need to find out if they [colleagues of the interviewee] are using it [the component] for the same purpose*" (Interview 2). "*It refers to whether the component offers the same functionality*."
2. Product type: Sometimes mentioned as system type. "*A major factor is the kind of the product for which I am seeking solutions. If the products are similar, then the decision can be reused*" (Interview 3). "*[…] relevant product information includes: functionality, complexity and technologies*" (Interview 7). "*As stated, the product type relates to functional use (mentioned above), however, products of the same type maybe also be having different functional use. Such an example is the same product type 'consumer-oriented software' of an email client and a text editor, which share some functionality but not all functional use is the same*."
3. Application domain: Refers to the main line of business of a company. "*The closer to our domain other companies are, the more useful their experiences*" (Interview 2). "*These decisions, their problem space, are very company–specific. The application domain (for example banking) is critical*" (Interview 4). It was mentioned in another interview that decisions are interesting even from a different domain "*[…] for example if the automotive industry is investigating directions that we think will be relevant to us in the near future, a decision made in that domain would still be interesting to us*" (Interview 9).
4. Complexity: In Interview 8, the "*amount and complexity of data processed*" is mentioned, whereas Interviews 4 and 7 refer to product complexity (concerning technical debt in particular in Interview 4).
5. Availability of support: Community support "*does it get bug fixes, does it have active community members, does it get support?*" (Interview 4). Other interviews referred to this as "*activeness of the community*" (Interview 1) and "*How many are supporting it […] willingness to provide the support and quality of the support*" (Interview 6).

With respect to RQ2, concerning differences in context factors between the experience sources in the scenario, we observe the following:

- In some interviews (Interviews 1, 7, 8, and 9), the context factors mentioned across all experience sources in the scenario do not change significantly. That means that across most of the experience sources we analysed, the same factors are considered important context factors. Thus, we observe fewer differences than expected between the experience sources of the scenario we investigated. Because the context factors do not change, practitioners can consult the sources that are the most convenient for them regardless of the context, thus reducing the time or effort needed to look into less available sources of information.
- In Interviews 2, 4, 6, and 11, a set of the same context factors is mentioned for experience sources S1 and S2 in the scenario, within the interviews (and not across interviews). This indicates a similarity of the context factors considered essential within each company/interview and between these two experience sources (S1 and S2). However, the context factors cannot be generalised across the interviews.
- In Interviews 2, 6–9, 11, and 12, the context factors change when experience sources S3–S5 are discussed. The experience sources of S3 and onward are considered distant from the companies, as they involve considering experiences outside the company. Thus, perhaps this justifies the change in the context factors mentioned, which we observed.
- A similar observation may also be made for Interviews 3, 5 and 10, but to a lesser degree.
- For experience source S3 and in Interviews 1, 3, 6, and 11, new context factors appeared.
- Concerning experience source S4, it was observed (in Interviews 4 and 8) that new context factors appeared.
- For experience source S5, it was observed (in Interviews 4–6 and 8–12) that new context factors appeared.

6 | DISCUSSION

As a result of the study, four areas for further discussion emerged.

Number of context factors: The exploratory nature of this study resulted in a large number of context factors. The context factors were also mentioned differently by the interviewees due to the context diversity, and where appropriate, examples were used to clarify the meaning of the context factors. We, therefore, see the need for more research on how to formulate and document context factors perceived as important when reusing knowledge to make decisions. Developing a unified description of the context factors is the first step towards this clarification. Focusing on context factors perceived as important when reusing knowledge helps document the most critical context factors and limit irrelevant information.

Implicit versus explicit context: Some participants discussed aspects that could be considered implicit information and thus excluded from the description, for example, the understanding of everyday decisions and the general organisational structure of a company. The difference between implicit and explicit context can also change in different knowledge reuse scenarios and depends on who the recipient of the information is. Knowing what context information to capture explicitly and at what level of detail can be very difficult, particularly when the recipient is unknown. For experienced practitioners, most context factors can be considered implicit. This may not be true for less experienced practitioners who need more verbose context descriptions. In a future study, we need to consider this difficulty when designing the study and investigate different scenarios depending on the experience of the interviewees. This study is a first step towards collecting context information without knowing the recipient (as it was left unknown during the interviews) and contributes to investigating what context factors practitioners, in general, find essential and make these context factors more explicit. We, therefore, highlight the need to explicitly document and describe the context information of decision cases. Explicit documentation of cases helps taking full advantage of the various context aspects, supporting making decisions more evidence based. This documentation also supports learning and knowledge exchange between experienced and inexperienced practitioners.

Mutual understanding of context: Understanding and communicating what context is and how it differentiates from, for example, product properties, business constraints, and organisational considerations, such as culture and development methods, is needed to clarify where and how context can be used. A mutual understanding of context will help identify the opportunities of using context to filter previous decision cases in a better way and identify those more likely to be relevant to a new or similar decision. While humans may comprehend the context, it is considered harder to describe and ensure a correct interpretation. The need to document contextual information is therefore highlighted here to support the mutual understanding of context. This brings us to the point of the practicality of the documented context.

Practicality of context and context factors: Identifying which context factors are perceived as most important when reusing or recalling knowledge from previous cases is essential. However, it is not clear how they can be used to solve a particular problem unless their prioritisation is clear, and hence to provide an opportunity to apply them in practice. In particular, the fact that an appropriate level of accepted ambiguity for the representation of context needs to be devised. However, this introduces another challenge, as it is hard to be concrete and unambiguous in the representation of context. For example, when describing the context of a decision-maker belonging to an incubator unit, and testing an innovation within a large multinational company, should the context describe the large company or is the incubator aspect of the context more relevant? In these two cases, we consider the context to be different. These differences are meant to be captured in cases where they would affect the decision. Therefore, it is vital to raise awareness of implicit and explicit context factors and the need for more codification of important implicit context factors.

6.1 | Implications to practitioners and researchers

The descriptions of the identified contextual factors that are perceived as important when reusing prior knowledge about software component reuse could be of interest to both researchers and practitioners. We would like to highlight implications for researchers and practitioners based on the results presented in Table 7. The first implication that is relevant for both practitioners and researchers emerges from the top four context factors elicited in this study (functional use, product type, application domain, and complexity). We believe that these factors could be automatically extracted and documented. This means that whenever a component is reused, it could carry a "backlog information" about previous usage scenarios, domains, product types, and complexity of previous contexts where it was deployed. This could formulate a form of "components curriculum vitae" to be verified before the next use. Researchers working in the area of recommendation systems could use our results as input for designing and evaluating context-aware proactive recommender systems supporting the code comprehension process, e.g., extending the work of Aghajani.[36] Practitioners could then use the support from the recommender system to obtain accurate context information for their upcoming decision. Automation also removes the risk of not capturing the most significant and beneficial information for decision-making in components reuse.

The results of our study imply that researchers should dedicate more effort to supporting practitioners in devising clear guidelines for quantifying and documenting subjective decision factors. As seen in Table 7, subjectivity and credibility of experience are listed as factors for different experience sources. Practitioners need to be aware that in-depth personal experience could and should not be compared with opinions and

subjective judgment from persons who have not worked with the development or deployment of software components. Thus, our work can support the efforts to understand how opinions are formulated by software engineers and what sources of opinions are considered, such as the work of Devanbu et al.[37]

Practitioners and researchers could also be supported by the context factors identified as important from the results of the study to classify solutions or decisions based on context factors. They may refer to those classes when similar context factors occur and reuse the similar case experiences. They may use case experiences to support decision-making for component sourcing and cater for particular solutions (for example to look for company-internal solutions or in open external databases to find similar decision alternatives to a sourcing problem). Past decisions may be (re)used for improved decision-making in software engineering and enable practitioners to reuse evidence from prior decisions in future decision-making. This implies creating efficient knowledge exchange methods between the cases and robust filtering methods of previous cases that can highlight the most relevant past experiences for the current decision to be made. Knowledge management in software engineering should also be applied to reusing experiences with selected components, a point not clearly visible in a review conducted in this area.[38]

The study increases the overall understanding of context which is in its nature a challenging and ambiguous topic. The study also increases our understanding of contextual factors which are important when choosing components, but also in other cases. Therefore, we believe that by making these context factors clearer or highlighted, researchers may build a taxonomy of factors to use when describing the context in case studies. This topic is both under-researched and essential not only to set the scene for research work but also to make comparable studies between them.

7 | VALIDITY THREATS

The identified validity threats of this work are presented in this section as follows:[39]

- *Construct validity* refers to asking the right questions and selecting the right interviewees. The form of the questionnaire used (being structured) and having a semi-structured set of questions ensured that the process led to a discussion. This eventually meant being clear about what the interviewers were asking, and hence collecting the correct information, that is, the vital context factors when selecting components. If needed, the validation form was used to clarify the correctness of the information collected. Furthermore, the interviewees with prior experience in selecting software components were selected, which helped in getting relevant answers. The interviewees were identified using the authors' industrial contacts. Despite that, a threat to construct validity remains because, in the study, we elicited perceptions on a given scenario and context factors associated with it. We had no mechanism for evaluating if extracted experiences were real and thus had to rely on the opinions expressed as the single source of evidence. This lack of triangulation remains a threat to the construct validity of our study. We know that practitioners could have had some difficulties identifying the most essential context factors to reuse given this scenario. Therefore, we dedicated additional effort to explain the scenario and providing relevant examples during the interviews.
- *Descriptive validity* refers to the objectivity, correctness and completeness of the information gathered in the interviews. The validation process of the information from the interviews included review iterations of the data collected involving at least two and mostly three researchers, who inspected and discussed the final data used in the results section of this article. The original interview data were also recorded and reviewed several times by the researchers to ensure correctness and completeness. Objectivity is one issue that remains open in this study as the selection of the participants introduces some bias to the results towards individual perceptions and companies. One way to address it was to use a generalised scenario for the data collection and ask about the interviewees' context at the end of the interview, to not bias the interviewee about the context factors. We also assume that our respondents are objective when discussing relevant context factors for each scenario.
- *Interpretative validity* refers to researchers' bias in drawing conclusions. To mitigate this threat, we recorded all interviews and extracted the responses as close to the statements of the interviewee as possible. In an attempt to dissolve ambiguities or uncertainties, some of the interviewees were contacted a second time and, in some cases, a third time. These steps, along with the validation forms used, have increased the validity of our contribution.
- *Generalisability* refers to the ability to generalise the findings. The limited number of participants included in the interviews represents a possible threat to validity. The interviewees representing different roles working in eight different companies and six domains contribute to producing generic results. However, given that it is the first study of this kind, we did not primarily aim for generalisability. It was seen as more essential to contribute with an understanding concerning context factors and their importance as a starting point for further research. The diversity in the responses from the interviews highlights the need for further studies. We do not claim statistical validity of our findings. Therefore, we have focused on analytic generalisation rather than statistical generalisation[40] by comparing the characteristics of the cases and presenting case descriptions.

8 | CONCLUSION

This study contributes with the context factors that are perceived as important when reusing previous knowledge, within the specific scenario of deciding which software component types to use. The context factors may also be of interest in other scenarios, which is an area for further research. Furthermore, we contribute with listings of context factors perceived to be important when reusing experiences from other decision-making cases of selecting among software components options. Thus, the objective is to influence the research community to include the perceived most important contextual factors in publications.

The research contributes to an increased understanding and provides support for both practitioners and researchers. The context factors perceived to be important by practitioners were studied through an interview study with practitioners in different roles and from different companies. The interview study was conducted given a specific scenario to make the discussion more concrete with the practitioners. The drawback is that it affects generalisability. Furthermore, the practitioners interviewed may not be fully representative of the general population of software engineers working in the industry. However, the study provides some initial and novel findings concerning which context factors that are perceived as important by practitioners, given that we have not found many existing studies on this specific topic within software engineering.

ACKNOWLEDGMENTS

The work is supported by a research grant for the ORION project (reference number 20140218) from the Knowledge Foundation in Sweden.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available in the supplementary material of this article.

REFERENCES

1. Oxford University Press. Oxford Dictionary. Accessed: 2022-02-15; 2020.
2. Petersen K, Carlson J, Papatheocharous E, Wnuk K. Context checklist for industrial software engineering research and practice. *Comput Stand Interf*. 2021;78:103541.
3. Kitchenham BA, Dybå T, Jørgensen M. Evidence-based software engineering. In: Proceedings 26th International Conference on Software Engineering. IEEE; 2004:273-281.
4. Dybå T, Kitchenham BA, Jørgensen M. Evidence-based software engineering for practitioners. *IEEE Softw*. 2005;22(1):58-65.
5. Petersen K, Wohlin C. Context in industrial software engineering research. In: Proceedings 3rd International Symposium on Empirical Software Engineering and Measurement. IEEE; 2009:401-404.
6. Clarke P, O'Connor RV. The situational factors that affect the software development process: towards a comprehensive reference framework. *Inform Softw Technol*. 2012;54(5):433-447.
7. Dybå T, Sjøberg DIK, Cruzes DS. What works for whom, where, when, and why? On the role of context in empirical software engineering. In: Proceedings 6th International Symposium on Empirical Software Engineering and Measurement. ACM; 2012:19-28.
8. Dybå T. Contextualizing empirical evidence. *IEEE Softw*. 2013;30(1):81-83.
9. Wohlin C, Rainer A. Challenges and recommendations to publishing and using credible evidence in software engineering. *Inform Softw Technol*. 2021;134:106555.
10. Wohlin C, Papatheocharous E, Carlson J, et al. Towards evidence-based decision-making for identification and usage of assets in composite software: a research roadmap. *J Softw: Evol Process*. 2021;134:106555.
11. Briand L, Bianculli D, Nejati S, Pastore F, Sabetzadeh M. The case for context-driven software engineering research: generalizability is overrated. *IEEE Softw*. 2017;34(5):72-75.
12. Basili V, Briand L, Bianculli D, Nejati S, Pastore F, Sabetzadeh M. Software engineering research and industry: a symbiotic relationship to foster impact. *IEEE Softw*. 2018;35(5):44-49.
13. Tang A, Lau MF. Software architecture review by association. *J Syst Software*. 2014;88:87-101.
14. Basili VR, Caldiera G, Rombach HD. Experience factory. *Encyclopedia of Software Engineering*: American Cancer Society; 2002. https://onlinelibrary.wiley.com/doi/abs/10.1002/0471028959.sof110
15. Shrikanth NC, Menzies T. Assessing practitioner beliefs about software defect prediction. In: 2020 IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP). ACM; 2020:182-190.
16. Passos C, Braun AP, Cruzes DS, Mendonca M. Analyzing the impact of beliefs in software project practices. In: Proceedings International Symposium on Empirical Software Engineering and Measurement. IEEE; 2011:444-452.
17. Wohlin C, Aurum A, Angelis L, et al. The success factors powering industry-academia collaboration. *IEEE Softw*. 2012;29(2):67-73.
18. Gorschek T, Garre P, Larsson S, Wohlin C. A model for technology transfer in practice. *IEEE Softw*. 2006;23(6):88-95.
19. Gorschek T, Wnuk K. Third generation industrial co-production in software engineering. In: Felderer M, Travassos GH, eds. *Contemporary Empirical Methods in Software Engineering*: Springer International Publishing; 2020:503-525.
20. Badampudi D, Wohlin C, Petersen K. Software component decision-making: in-house, OSS, COTS or outsourcing—a systematic literature review. *J Syst Softw*. 2016;121:105-124.
21. Li J, Bjørnson FO, Conradi R, Kampenes VB. An empirical study of variations in COTS-based software development processes in the Norwegian IT industry. *Empir Softw Eng*. 2006;11(3):433-461.
22. Li J, Conradi R, Bunse C, Torchiano M, Slyngstad OPN, Morisio M. Development with off-the-shelf components: 10 facts. *IEEE Softw*. 2009;26(2):80-87.

23. Li X, Moreschini S, Zhang Z, Taibi D. Exploring factors and metrics to select open source software components for integration: an empirical study. *J Syst Softw*. 2022;188:111255.
24. Kirk D, MacDonell S. Categorising software contexts. In: Proceedings 20th Americas Conference on Information Systems (AMCIS). Association for Information Systems; 2014.
25. Bi T, Liang P, Tang A. Architecture patterns, quality attributes, and design contexts: How developers design with them. In: 2018 25th Asia-Pacific Software Engineering Conference (APSEC). IEEE; 2018:49-58.
26. Bedjeti A, Lago P, Lewis GA, De Boer RD, Hilliard R. Modeling context with an architecture viewpoint. In: 2017 IEEE International Conference on Software Architecture (ICSA). IEEE; 2017:117-120.
27. Harper KE, Zheng J. Exploring software architecture context. In: 2015 12th Working IEEE/IFIP Conference on Software Architecture. IEEE; 2015: 123-126.
28. Cartaxo B, Saraiva J, Almeida A, Barreiros E, Neto WPF, Soares S. Mechanisms to characterize context of empirical studies in software engineering. In: Proceedings 18th Ibero-Americal Conference on Software Engineering. Curran Associates, Inc.; 2015:281.
29. Carlson J, Papatheocharous E, Petersen K. A context model for architectural decision support. In: Proceedings 1st International Workshop on Decision Making in Software ARCHitecture (MARCH). IEEE; 2016:9-15.
30. Hertzum M. The importance of trust in software engineers' assessment and choice of information sources. *Inform Org*. 2002;12(1):1-18.
31. Jansen BJ, Rieh SY. The seventeen theoretical constructs of information searching and information retrieval. *J Am Soc Inform Sci Technol*. 2010;61(8): 1517-1534.
32. Freund L. Contextualizing the information-seeking behavior of software engineers. *J Assoc Inform Sci Technol*. 2015;66(8):1594-1605.
33. Borg M, Alégroth E, Runeson P. Software engineers' information seeking behavior in change impact analysis—an interview study. In: 2017 IEEE/ACM 25th International Conference on Program Comprehension (ICPC). IEEE; 2017:12-22.
34. Baltes S, Ralph P. Sampling in software engineering research: a critical review and guidelines. arXiv preprint arXiv:200207764; 2020.
35. Elo S, Kyngäs H. The qualitative content analysis process. *J Adv Nurs*. 2008;62(1):107-115.
36. Aghajani E. Context-aware software documentation. In: 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME). IEEE; 2018:727-731.
37. Devanbu P, Zimmermann T, Bird C. Belief and evidence: how software engineers form their opinions. *IEEE Softw*. 2018;35(6):72-76. https://doi.org/10.1109/MS.2018.4321246
38. Bjørnson FO, Dingsøyr T. Knowledge management in software engineering: a systematic review of studied concepts, findings and research methods used. *Inform Softw Technol*. 2008;50(11):1055-1068. https://doi.org/10.1016/j.infsof.2008.03.006
39. Petersen K, Gencel C. Worldviews, research methods, and their relationship to validity in empirical software engineering research. In: 2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement. IEEE; 2013:81-89.
40. Flyvbjerg B. Five misunderstandings about case-study research. *Qual Inq*. 2006;12(2):219-245.

How to cite this article: Papatheocharous E, Wohlin C, Badampudi D, Carlson J, Wnuk K. Context factors perceived important when looking for similar experiences in decision-making for software components: An interview study. *J Softw Evol Proc*. 2024;e2668. https://doi.org/10.1002/smr.2668

APPENDIX A: INTERVIEW VALIDATION FORM

See Figures A1 and A2.

Summary of interview

We would like to verify if we have interpreted your answers in the correct way. We have transcribed the key points of the interview in Table 1. Kindly verify your response and add information if you perceive it to be missing. The background of the interview is presented in the text box below.

Background - General context
You (interviewee) are about to **evolve your existing product** and have identified some different options for doing so, in the form of software components. You are prepared to consider different options for these components: in-house development, in-house reuse, open source, COTS and outsourcing to someone else to develop it for you.
To make a decision consider that you **want to build on experience from "similar" previous cases**.
When looking for "similar" cases within different situations (see below) you mentioned the following **context factors** that may be important.

Table 1: Key responses from the interview

Situation 1	What context factors would you consider when identifying similar cases **from your own experiences?**	
Correct? (Yes/No)	**Your response**	**Ranking of factors (including the missed ones)**
•	•	•
•	•	•
•	•	•
		•
Missed factors?	•	
Comments (If any)		

Situation 2 (a)	What context factors would you consider when identifying similar cases from **YOUR company, from a colleague who is close to you** (either regarding geographically close (easy to talk to) or close concerning a similar type of product)?	
Correct? (Yes/No)	**Your response**	**Ranking of factors (including the missed ones)**
•	•	•
•	•	•
•	•	•
		•
Missed factors?	•	
Comments (If any)		

Situation 2 (b)	What context factors would you consider when identifying similar cases from **YOUR company, from a colleague who is *not* so close to you** (either regarding geographically close (easy to talk to) or close concerning a similar type of product)?	
Correct? (Yes/No)	**Your response**	**Ranking of factors (including the missed ones)**
•	•	•
•	•	•
•	•	•
		•
Missed factors?	•	
Comments (If any)		

FIGURE A1 Interview form template used for validation (Part 1/2).

Situation 3	What context factors would you consider when identifying similar cases from ***other* companies** where you have acce... to their experiences (for example, close collaborators or good personal contacts)?	
Correct? (Yes/No)	**Your response**	**Ranking of factors (including the missed ones)**
• • •	• • •	• • • •
Missed factors?	•	
Comments (If any)		

Situation 4	What context factors would you consider when identifying similar cases from **open external databases with published cases**? (hypothetical case, if such a database would exist)	
Correct? (Yes/No)	**Your response**	**Ranking of factors (including the missed ones)**
• • •	• • •	• • • •
Missed factors?	•	
Comments (If any)		

Situation 5 (a)	What context factors would you consider when identifying similar cases from **online sources** such as developers' forum (such as Stack Overflow), blogs, newsletters and social networks, such as Linkedin?	
Correct? (Yes/No)	**Your response**	**Ranking of factors (including the missed ones)**
• • •	• • •	• • • •
Missed factors?	•	
Comments (If any)		

Situation 5 (b)	What context factors would you consider when identifying similar cases from **published literature** (both research and more practitioner-oriented publications), including books, journal articles, conference papers?	
Correct? (Yes/No)	**Your response**	**Ranking of factors (including the missed ones)**
• • •	• • •	• • • •
Missed factors?	•	
Comments (If any)		

FIGURE A2 Interview form template used for validation (Part 2/2).

APPENDIX B: FULL LIST OF CONTEXT FACTORS

See the supplementary material for the full table of context factors collected in the 12 interviews—https://bthse-my.sharepoint.com/:x:/g/personal/deb_bth_se/EcUrW5HX6MhJgmOmGBxy4qgB90MRDAnS0xIlw7wBzc7IVQ?e=Yfd3Fq.