Software Quality Journal (2024) 32:1075–1135
https://doi.org/10.1007/s11219-024-09678-0

RESEARCH

Towards engineering higher quality intelligent environments: a multi case study approach

Adityarajsingh Santokhee[1] · Juan Carlos Augusto[2] · Lindsey Brodie[2]

Accepted: 28 April 2024 / Published online: 14 June 2024

Abstract
This study addresses the need to enhance the quality of Intelligent Environments, recognizing their unique characteristics and the absence of adequate guidance on quality management during development. It pursues three primary objectives: proposing a novel quality-in-use model, presenting an enhanced version of the User-Centered Intelligent Environment Development Process, and reporting on the application of these approaches through a multiple case study. To embed quality into systems, we advocate for the integration of quality characteristics from ISO/IEC 25000 standards with functional requirements. Stakeholders collaboratively define targets using measures from quality standards, and metrics enable early problem detection and resolution during development. The proposed quality-in-use model provides an insightful and objective perspective on system capabilities, guiding development and ensuring stakeholder involvement. However, challenges such as shortening development cycles for early and regular stakeholder feedback and managing an increased number of system tests were noted. Our study makes a significant contribution to the field of Intelligent Environments by providing a structured approach to embedding and managing quality throughout the development lifecycle. The multiple case study offers empirical evidence of the effectiveness of the proposed approaches, with ongoing considerations for challenges in the development process.

Keywords Intelligent environments · Quality model · Multiple case study · Qualitative study

✉ Adityarajsingh Santokhee
a.santokhee@mdx.ac.mu

Juan Carlos Augusto
j.augusto@mdx.ac.uk

Lindsey Brodie
lindseybrodie@btopenworld.com

[1] R.G. on Intelligent Environments, School of Digital Technologies, Middlesex University Mauritius, Uniciti, Mauritius

[2] R.G. on Intelligent Environments, Department of Computer Science, Middlesex University, London, UK

1 Introduction

An Intelligent Environment (IE) refers to a physical space which is composed of a network of devices such as sensors and actuators which are orchestrated by algorithms which sensibly but proactively support people in carrying out their daily activities (Augusto et al., 2013). IEs inherit concepts and technologies from various domains such as Ubiquitous or Pervasive Computing (Weiser, 1991), Ambient Intelligence (Aarts & Roovers, 2003), Smart Environments (Rashidi et al., 2011) and Internet of Things (Atzori et al., 2010). In fact, these are closely related domains. IEs are complex systems comprising of sensing technology and are expected to perform a wide variety of personalised functions in various fields such as independent living (Martirano & Mitolo, 2020), education (Rizk & Hillier, 2022), smart home, healthcare (World Health Organization, 2022), ambient assisted living (Memon et al., 2014), agriculture, and smart factory. They are deployed in physical environments and react on real-time data according to contexts defined for specific stakeholders (Banijamali et al., 2020). For instance, the key goal of an ambient assisted living system is to "*proactively, but sensibly, support people in their daily lives*" (Augusto, 2007).

However, since IEs are highly user-centric systems, it is imperative that they are not only effective but also ethically viable for their end users (Augusto et al., 2013). Additionally, developers also need to cater for important IE specific technical challenges such as context-awareness, tracking preferences of users, implementing reasoning and dealing with hardware malfunctions (ibid.). Traditional software engineering methods and tools lack the maturity needed to effectively engineer these systems and address their specific challenges (Ahmad et al., 2015). Thus, the User-Centred Intelligent Environments Development Process (U-CIEDP) was proposed to develop these types of systems (Augusto et al., 2018). To date, U-CIEDP has been applied to mostly research oriented projects. This led to identification of limitations based on lessons learnt. It has been reported that each phase in U-CIEDP requires strong planning to avoid uncoordinated system development (Augusto et al., 2018; Ogbuabor et al., 2021; Santokhee et al., 2019). Secondly, there is no clear strategy to assess the quality of the systems, during or after development, which is a major limitation (ibid.).

As our reliance on IEs grows, there is an undeniable increase in the demand for higher quality systems. Historically, subpar quality has been a primary cause of failure in software-based products (Jones & Bonsignour, 2011). Given the diverse and critical applications IEs are expected to handle, low quality could result in severe consequences, potentially even endangering lives in medical contexts. System failures lead to the cessation of services, and the complexity of IEs, with their numerous components and intricate human-computer interactions, makes them susceptible to unforeseen issues (ibid.). For instance, upgrading firmware for one hardware component may alter an IE's behaviour, presenting challenges in anticipating and testing such scenarios. Complete control over interfaces is essential for successful system integration, especially as the interactions among components contribute to the emergence of effects unattainable by individual elements alone. Human-machine interfaces add complexity, as these systems must function in dynamic operational conditions involving diverse hardware devices, intricate interactions, unpredictable resource availability, unforeseen usage scenarios, and the occurrence of hard-to-predict errors.

Therefore, this study is motivated by the necessity to engineer higher quality IEs. It aims to contribute to the broader understanding of challenges in the process of developing higher quality IEs. U-CIEDP has been enhanced by incorporating new activities to manage quality to each of its three core development stages: initial scoping, main development, and installation. The overall goal is to produce systems which meet users' or stakeholders' expectations

while satisfying core quality requirements. A novel IE specific quality model to evaluate IEs is also being proposed. The model is derived from the generic ISO/IEC 25010 quality-in-use model and has been adapted according to the nine guiding principles of IEs, as elaborated in Section 2 (Augusto et al., 2013; ISO/IEC 25010, 2021). A multi case-study approach was applied as to explore applicability of the enhanced U-CIEDP methodological framework (Kurtel & Ozemre, 2013; Runeson & Höst, 2009; Sicari et al., 2019; Scott et al., 2021; Staron et al., 2011; Tröls et al., 2021; Yin, 2018). Design of the case study was formulated following Yin's five steps (Yin, 2018). One more motivation for using case study research is that we would like to investigate whether the framework would apply to real-world settings. We report on two case studies in this paper. By incorporating quality characteristics from ISO/IEC 25000 (2021) standards into functional requirements, our approach ensures the inherent integration of quality into system development (Brodie & Woodman, 2009; Gilb & Brodie, 2012). Collaborative efforts between stakeholders and the project team facilitate the definition of targets using measures derived from quality standards. Utilising quality metrics enables developers to monitor deviations from quality targets, addressing issues early in the development stage and enhancing the overall assessment of application quality. The proposed quality-in-use model offers valuable insights, guiding the development process by providing an objective perspective on the system's expected capabilities. Continuous stakeholder involvement throughout ensures the delivery of systems that offer optimal value.

The structure of this article is as follows. Section 2 highlights some background concepts underpinning this study. In Section 3, we report on a Systematic Literature Review which was carried out to inform this research study and discusses the research question and propositions pertaining to this study based on gaps identified through analysis of the literature. The proposed quality-in-use model and UCIEDP2 are described in Section 4. Design and details of the multiple case study are explained in Section 5. In Section 6, we discuss the findings of the case studies by aggregating their results. Threats to validity are analysed in Section 7. The paper ends with a conclusion, limitations, and areas for future work in Section 8.

2 Background

The development of IE requires a multidisciplinary team that has to be not only capable of applying a combination of techniques and methods coming from software engineering to improve its reliability, but also from other Computer Science disciplines, such as Artificial Intelligence, Ubiquitous/Pervasive Computing and Human-Computer Interaction, among others, that make the resulting IE less intrusive, while being smarter, more proactive and usable for the user (Augusto et al., 2013; Dyba et al., 2007; Salvi et al., 2015). Table 1 summarises the nine principles proposed in an article by Augusto et al. (2013). The idea is that every IE should aspire to possess these core principles to ensure systems which are developed are technically and ethically viable. However, some of the principles are entangled, in the sense, achieving one may impact on the degree of fulfilment of other principles. For instance, developing a high-performance IE may require reduced security checks. Therefore, the key challenge is in implementing and managing these core principles during the development lifecycle whilst taking into consideration important quality attribute trade-offs. IEs also pose specific technical challenges such as context-awareness, tracking preferences of multiple users in an environment, implementing reasoning, and dealing with hardware malfunctions (Augusto et al., 2013). More importantly, since these systems are highly user centric, human decision may not be rational, repeatable, or testable.

Table 1 Intelligent Environments Manifesto (Augusto et al., 2013)

Principle	Description
PE1	to be intelligent to recognize a situation where it can help
PE2	to be sensible to recognize when it is allowed to offer help
PE3	to deliver help according to the needs and preferences of those which is helping
PE4	to achieve its goals without demanding from the user/s technical knowledge to benefit from its help
PE5	to preserve privacy of the user/s
PE6	to prioritize safety of the user/s at all times
PE7	to have autonomous behaviour
PE8	to be able to operate without forcing changes on the look and feel of the environment or on the normal routines of the environment inhabitants
PE9	to adhere to the principle that the user is in command and the computer obeys, and not vice versa

IEs can also be considered as complex software intensive systems, which are described as "*systems where the software contributes essential influences to the design, construction, deployment and evolution of the system as a whole*" (Sommerville, 2011). There are diverse perspectives towards quality according to some of the reputable references on quality (Anurag & Kamatchi, 2019; Cote et al., 2006). Although quality as a topic has been widely studied in software engineering for different types of systems (Anurag & Kamatchi, 2019; Benghazi et al., 2012; Kara et al., 2017; Kurtel & Ozemre, 2013; Regan et al., 2020; Vogel et al., 2021), we argue that there is lack of consensus regarding how to develop higher quality IEs.

Unfortunately, traditional software engineering methods and tools are not mature enough to engineer high quality IEs and to deal with the design challenges posed by these types of systems (Ahmad et al., 2015). In a recent study, Olianas et al. (2022) reflected that assuring quality in IoTs is challenging and they reported results of applying a prototype tool to perform system level testing of these systems. Therefore, we argue that more specific quality models, methodologies, and tools are required to develop and evaluate high quality IEs. As a first step in this study, it was deemed necessary to investigate more thoroughly how IEs are developed from a quality perspective.

3 Literature review

It was deemed imperative to conduct a more comprehensive investigation into the development of IEs from a quality perspective. A Systematic Literature Review (SLR) was conducted following the guidelines proposed by Kitchenham and Charters (Kitchenham, 2007) in three phases: planning, conducting, and reporting. The planning phase specified the necessity for conducting the analysis and the review framework. In the second phase, research was identified, primary studies were chosen based on criteria, their quality assessed, and data extracted for synthesis. The third phase involved reporting documents and data obtained during the review. Two iterative processes were also implemented during the review process to minimise the introduction of biases in the research (Wohlin, 2014).

3.1 Planning the SLR

The main goal of the Systematic Literature Review (SLR) was to investigate the current state of the art regarding the quality of IEs. To achieve this, the main research question was broken down into more specific inquiries grouped into three broad areas: definition, measurement, challenges, to comprehensively address the inquiry. Table 2 lists the specific research questions for the SLR.

RQ1 sought to distinguish the definition of quality for IEs. It aimed to explore researchers' perspectives on quality, the commonly employed quality characteristics, how conflicting quality requirements are addressed, and if there exists any system development methodology specifically focusing on quality within the domains of IEs. The second research question (RQ2) delved into the various approaches used to evaluate the quality of IEs, examining the application of quality models, the system development stage(s) for measuring quality requirements. The third research question (RQ3) investigated the prevailing challenges and identified areas for future work as documented in the literature.

3.2 Search strategy

A crucial step in a SLR involves identifying pertinent studies capable of addressing the research questions. Thus, the selection of appropriate search terms and keywords becomes paramount. In this study, we adopted an iterative approach, facilitating the analysis and gradual refinement of the search string. The search strings were devised by incorporating synonyms and abbreviations, connected through Boolean expressions, with the aim of retrieving a comprehensive set of publications. For this systematic literature review, five distinct databases (ACM, Web of Science, IEEE, Science Direct, and Springer Link) were chosen due to their extensive coverage of topics in software engineering. Table 3, presented below, outlines the executed search strings for each database, with the user guide for each database consulted to enhance the syntax of the search strings.

3.3 Defining inclusion and exclusion criteria

To determine the primary studies for further consideration, inclusion and exclusion criteria were established following the guidelines of Kitchenham and Charters (Kitchenham, 2007). A study underwent further analysis only if it met all the inclusion criteria and none of the exclusion criteria. The inclusion criteria are defined as follows:

IC1: The study is related to an aspect of quality for IEs.
IC2: The study is a peer-reviewed journal, conference or workshop proceeding.
IC3: The study addresses one or more of the review questions.

The exclusion criteria consist of:

EC1: The study is written in a language other than English.
EC2: The focus of the study is not related to any aspect of quality for IEs.
EC3: Duplicate studies.

Table 2 Research Questions for SLR

ID	Questions	Sub Questions
RQ1	How is quality defined for IEs?	RQ1.1: Which aspects of quality have been considered by researchers in IE domains?
		RQ1.2: How are quality requirements specified?
		RQ1.3: Which quality characteristics have been proposed for IEs?
		RQ1.4: How were the quality characteristics derived?
		RQ1.5: How are conflicting quality requirements managed and resolved?
		RQ1.6: Which system development methodology is used to develop IEs with a focus on quality?
RQ2	How is quality of IEs evaluated?	RQ2.1: How are quality requirements measured?
		RQ2.2: How is quality of IEs evaluated?
		RQ2.3: During which phase of system development are the quality requirements measured?
RQ3	What are the challenges and future research directions?	

Table 3 Search Strings

Database	Search Strings
ACM	"query": { Fulltext:("intelligent environment" OR "ambient intelligence" OR "ambient assisted" OR "pervasive" OR "ubiquitous" OR "smart environment" OR "internet of things") AND Fulltext:("quality evaluation" OR "quality assessment" OR "quality model" OR "quality attribute" OR "quality characteristic" OR "quality requirement" OR "nonfunctional requirement") AND Fulltext:("metric*" OR "measure*")} "filter": { Media Format: PDF, Article Type: Research Article, E-Publication Date: (01/01/2003 TO 01/31/2023)}
Web of Science	TS = ("intelligent environment" OR "ambient intelligence" OR "ambient assisted" OR "pervasive" OR "ubiquitous" OR "smart environment" OR "iot" OR "internet of things") AND TS = ("quality evaluation" OR "quality assessment" OR "quality model" OR "quality attribute" OR "quality characteristic" OR "quality requirement" OR "nonfunctional requirement") AND TS = ("metric" OR "measure" OR "measurement")
IEEE	("intelligent environment" OR "ambient intelligence" OR "ambient assisted" OR "pervasive" OR "ubiquitous" OR "smart environment" OR "iot" OR "internet of things") AND ("quality evaluation" OR "quality assessment" OR "quality model" OR "quality attribute" OR "quality characteristic" OR "quality requirement" OR "nonfunctional requirement") AND ("metric" OR "measure" OR "measurement")
Science Direct	("intelligent environment") AND ("quality characteristic" OR "quality requirement" OR "quality attribute" OR "nonfunctional requirement") AND ("metric" OR "measure" OR "measurement") ("smart environment" OR "internet of things") AND ("quality characteristic" OR "quality requirement" OR "quality attribute" OR "nonfunctional requirement") AND ("metric" OR "measure" OR "measurement") ("ambient intelligence" OR "ambient assisted") AND ("quality characteristic" OR "quality requirement" OR "quality attribute" OR "nonfunctional requirement") AND ("metric" OR "measure" OR "measurement") ("pervasive" OR "ubiquitous") AND ("quality characteristic" OR "quality requirement" OR "quality attribute" OR "nonfunctional requirement") AND ("metric" OR "measure" OR "measurement")
Springer Link	("intelligent environment" OR "ambient intelligence" OR "ambient assisted" OR "pervasive" OR "ubiquitous" OR "smart environment" OR "iot" OR "internet of things") AND ("quality evaluation" OR "quality assessment" OR "quality model" OR "quality attribute" OR "quality characteristic" OR "quality requirement" OR "nonfunctional requirement") AND ("metric" OR "measure" OR "measurement")'

EC4: Paper less than four pages in length.
EC5: Magazine, dissertation, tutorial, editorial, book, poster or not peer-reviewed publication.
EC6: Systematic mapping or literature reviews.
EC7: The study is published before 2003.

Following the application of inclusion and exclusion criteria, the main author reviewed the abstract of each shortlisted paper to determine its eligibility for further screening. This process was iterated at least twice on separate occasions to minimize bias. Four duplicate studies were identified and subsequently excluded. Upon completion of the initial screening, the full text of each paper was obtained and comprehensively examined by the main author. Table 4 provides a summary of the number of papers screened at each stage.

Table 4 Number of Retrievals Per Database

Database	Number of papers retrieved	Number of papers after first scan	Number of papers after second scan
ACM	459	25	6
Web of Science	178	62	3
IEEE	129	21	6
Science Direct	3221	32	5
Springer Link	2137	43	13
Total			**33**

3.4 Conducting the SLR quality assessment

The remaining 33 papers underwent screening in an Excel sheet, with each paper assessed against the following three quality criteria:

Q1: Are aims and scope of the study clearly stated?
Q2: Are all the study questions answered?
Q3: Are the data source, contexts and conclusions described appropriately for future references?

To establish these quality criteria, we adhered to the guidelines provided by Kitchenham and Charters (Kitchenham, 2007). The following scale-point was applied to each question:

(i) A study fully meets a given quality criterion –1 point.
(ii) A study partially meets a given quality criterion –0.5 points.
(iii) A study does not meet a given quality criterion –0 point.

A total score was calculated by using the following formula:

$$\text{Quality score} = Q1 + Q2 + Q3$$

A study needed to attain a total quality score equal to or greater than 1.5 to qualify for further analysis. All 33 papers successfully met the quality criteria.

3.5 Data extraction

After evaluating the quality of each primary study, the subsequent step involved the extraction of data. A data extraction form was created in Microsoft Excel, structured as follows:

(i) Reference details
(ii) Concept of quality
(iii) Contribution
(iv) Domain
(v) Quality characteristic(s)
(vi) Specification
(vii) Methodology

(viii) When measured?
(ix) Type of study
(x) Challenges
(xi) Future work

The extraction of data was significantly streamlined by initially downloading and saving each full paper individually on disk. The researcher examined each paper to extract pertinent data, recording it in the Excel file. Appendix A provides a summary of the contribution of each paper, identified by a Paper ID attribute.

3.6 Reporting the SLR

Descriptive statistics were employed to analyze notable patterns in the publications, exploring trends such as the annual publication count, domains under consideration, empirical methodologies utilized, and the number of citations for each study. As illustrated in Fig. 1, the total number of publications on the topic has been consistently limited to one-two per year over the last 15 years, with a notable peak observed in 2018. In the early stages of growth around 2015, this sector witnessed a surge in interest, particularly in smart homes and cities, improved security, and an enhanced quality of life by 2018. The acceleration of research in this sector is attributed to a growing emphasis on interdisciplinary collaboration across various fields. Furthermore, the upsurge in funding opportunities and backing from academic institutions, government agencies, and industry could have contributed to the escalation in publications on IEs (European Commission, 2009). As shown in Fig. 2, the highest number of studies targeted IoT Systems (13), followed by AAL (7) and Ubiquitous Systems (5) during the last 15 years. Recent technological advancements and the widespread adoption of devices,

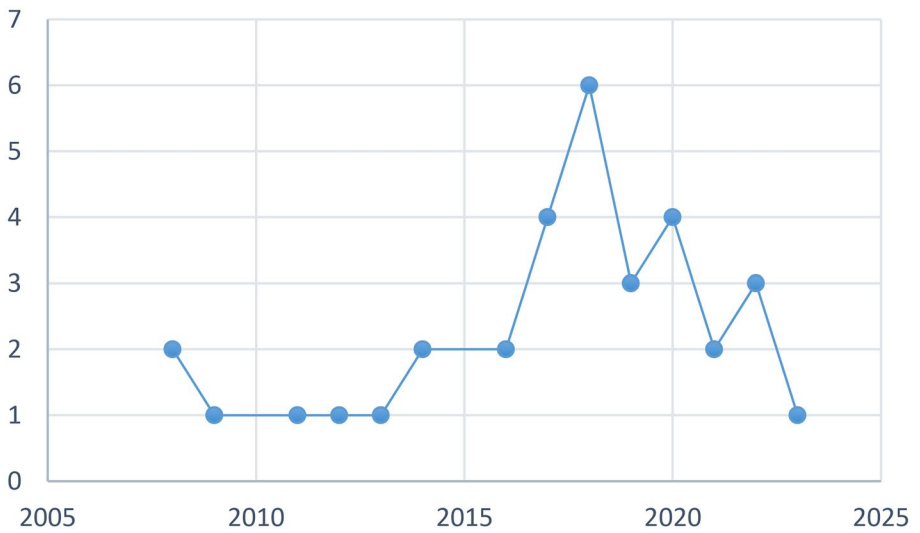


Fig. 1 Frequency of Publications Per Year

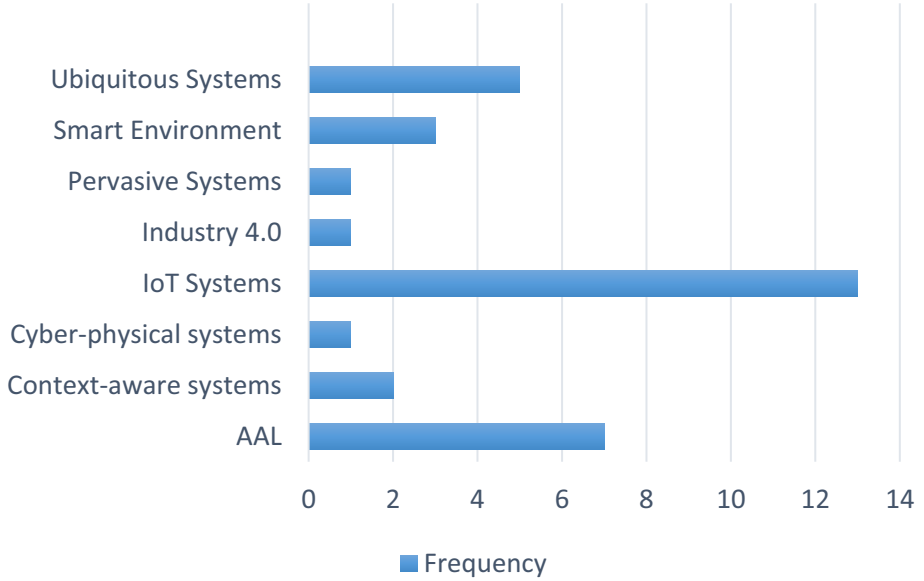
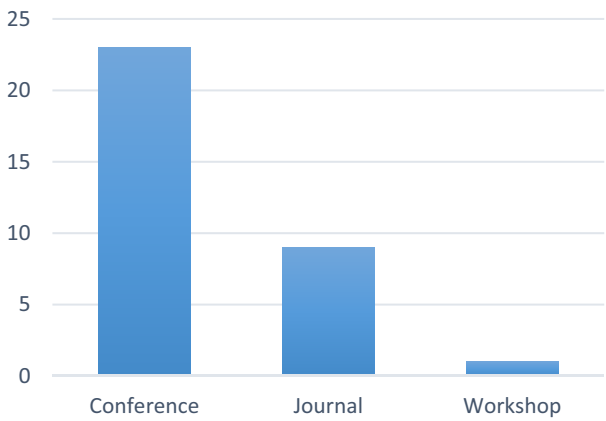


Fig. 2 Frequency of Publications Per IE Domain

especially in areas like AI, IoT, and sensor networks have contributed to proliferation of these systems (Reggio et al., 2020; McKinsey, 2021). Figure 3 depicts a notable prevalence of conference papers compared to journals in this field. While conferences offer a valuable platform for disseminating research, journal publications remain crucial for delivering more comprehensive and in-depth studies, rigorous peer review processes, and long-term archival of research in the domain of IEs. This observation highlights the relative novelty and rapid evolution of this field. Researchers also often present their initial findings and innovative ideas at conferences to receive early feedback and establish their presence in the field. We noted a rise in the number of citations per paper from 20 to 120 which again shows a growing interest of researchers in this field leading to an increase in number of publications, as shown in Fig. 4.

Fig. 3 Types of Publications



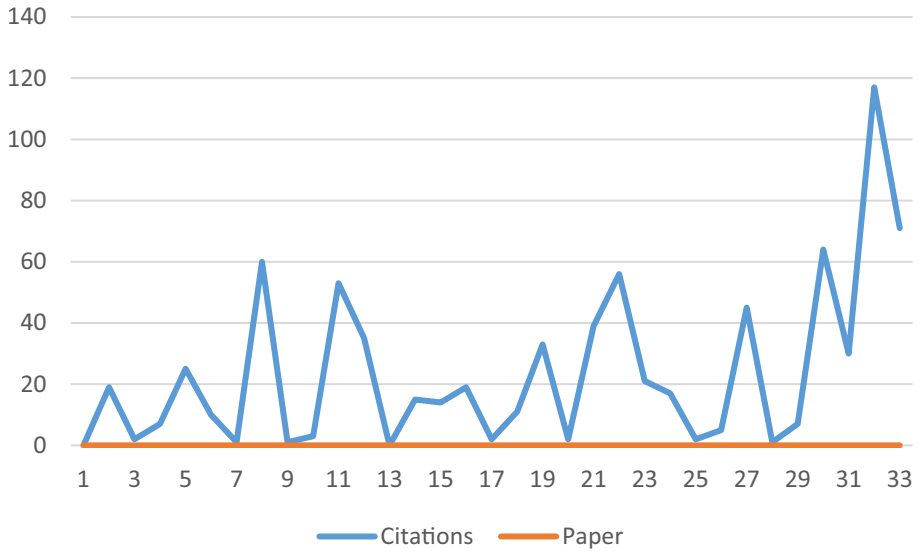


Fig. 4 Number of Citations Per Paper

3.7 Findings

3.7.1 RQ1 – how is quality defined for IE systems?

RQ1.1 Which aspects of quality have been considered by researchers in the domains of IE?

We noted that researchers adopted a non-functional requirements perspective when addressing the concept of quality. Sommerville (2011) emphasises the importance of non-functional requirements in shaping the overall quality of attributes of a software system. Pressman (2014) discusses the significance of non-functional requirements in determining the success or failure of a software project. In essence, the incorporation of non-functional requirements into the definition and assessment of quality is a recognised and widely accepted approach in Software Engineering literature. Specific quality characteristics were implied [P1, P4, P5, P7, P9, P12, P13, P17, P20, P21, P22, P24, P26, P28, P29]. Aspects of systems which were investigated are: usability [P2, P8, P18, P19, P25], data quality [P3, P31], quality of experience [P14, P15, P33], quality of experience [P6, P16, P27, P32], trust [P10] and quality of context [P23, P30]. This implies that in most of the studies under consideration, researchers approached the assessment and consideration of quality by looking beyond the functional features of the system. Instead, they paid attention to the broader characteristics that contribute to the overall usability, quality of experience, quality of context and reliability of the system. This approach acknowledges that a system's quality is not solely determined by its ability to perform specific tasks but is also influenced by how well it meets criteria related to its overall performance, user experience, security measures, and other non-functional attributes.

RQ1.2 How are quality requirements specified?

There is a scarcity of specific information concerning the specification of quality requirements. The utilisation of metrics emerged as a notable trend in several studies [P1, P5, P6, P17, P29, P30, P31]. In one instance [P3], quality requirements were gathered through communication with AAL service providers, while another study [P8] consulted caregivers for this purpose. In [P4], the definition and measurement of software quality factors were explicitly outlined.

The specification of quality requirements is pivotal in software engineering as it serves as the foundation for designing, developing, and evaluating a system (Pressman, 2014; Sommerville, 2011). Clear and precise quality requirements provide a roadmap for the development team, outlining the essential characteristics and attributes the system must possess to meet user expectations (Sommerville, 2011). This specification not only guides the development process but also forms the basis for subsequent testing and validation activities. It enables stakeholders to establish measurable criteria for success, facilitates effective communication between different project participants, and ultimately ensures that the delivered software aligns with user needs and organizational objectives. In essence, both Pressman (Pressman, 2014) and Sommerville (Sommerville, 2011) assert a well-defined specification of quality requirements is fundamental for achieving a successful and high-quality software product. Oram and Wilson (2010) discovered that deficiencies in precision and completeness within requirements and design documentation resulted in persistent design and requirements faults. These issues continued to be identified throughout the entire testing process, as indicated by their comprehensive survey of challenges encountered during the evolution of a large-scale real-time system.

RQ1.3 Which quality characteristics have been proposed for IEs?

We identified and grouped the quality characteristics according to IE domains discovered through the SLR to answer this question. These are summarised in Appendix B. We note that even within the same IE domain, different quality characteristics have been proposed in different studies. This shows that the choice of quality characteristics depends on the context where the system will be used. These findings corroborate with previous studies. For instance, in their evaluation of six AAL platforms (Alhambra, Hydra, OASIS, OpenAAL, PERSONA and UniversAAL), Antonino et al. (2011) selected quality attributes such as maintainability, efficiency and trustworthiness from ISO/IEC 9126, ISO/IEC 14598, and ISO/IEC 25000 Square standards. They believed that these attributes were critical to AAL systems. Memon et al. (2014) argued to study characteristics which lead to interoperability, usability, security, and accuracy rather than concentrating on isolated aspects of AAL. In their study of Smart Cities, Kakarontzas et al., 2014) identified interoperability, usability, authentication, authorization, availability, recoverability, maintainability, and confidentiality as the most prominent quality drivers. They also suggested that quality requirements could be defined using ISO/IEC 25010 (2021) standard. Garcés et al. (2017) stressed on the necessity of managing critical attributes such as security, freedom of risk, reliability, and performance efficiency since the start of AAL systems development. For Washizaki et al. (2020), interoperability was proposed as key attribute due to many participating entities which need to interact with one another. They also revealed that performance, usability, and scalability are central to certain IoT patterns for IoT systems and software while highlighting the need to study other quality attributes. Ashouri

et al. (2021) highlighted that performance, efficiency, time behaviour and resource utilization were the most popular quality attributes. They also reflected that few studies have studied critical attributes for IoT such as security, compatibility, portability, and maintainability. Fizza et al. (2023) proposed a new metric called quality of actuation to quantify correctness of actuation. The study also reiterated the importance of developing a generic model for measuring quality of autonomic applications and a framework to support their development. Thus, the overall picture which emerges is that this is an evolving field of study and researchers have focused on addressing specific quality goals of systems within IE domains.

RQ1.4 How were the quality characteristics derived?

In ten studies, the quality characteristics were derived from ISO/IEC 25010 (2021) or ISO/IEC 25000 (2021) standards [P12, P13, P17, P19, P20, P22, P26, P28, P29, P30], covering IE domains such as AAL, IoT, Smart Environments and Ubiquitous Systems. ISO/IEC 25010 (2021) is a universally recognised standard which provides comprehensive definitions for various quality attributes. However, only half of the cited studies [P13, P19, P22, P26, P30] provide empirical evidence of applying their quality characteristics. According to Hron and Obwegeser (2022) and Humble and Farley (2010), the definitions of these quality attributes have found acceptance across diverse industries including automative, naval, avionics and medical devices. These definitions have been embraced by industries aligning with Industry 4.0 technologies (Abdelouahid & Marzak, 2018; ISO/IEC 25010, 2021). In the remaining studies, the quality characteristics stemmed from prior research, literature reviews, and systematic mapping studies. Nevertheless, a few notable exceptions deviated from this trend. In the case of [P7], field interviews were undertaken after a thorough literature review, providing a unique perspective. Similarly, [P8] incorporated concerns and issues voiced by caregivers into the determination of quality aspects. However, it's noteworthy that the extent of engagement with end users remained relatively limited across most of the studies. Understanding how quality characteristics are derived is vital in software engineering (Sommerville, 2011). It offers transparency into the selection process, enabling stakeholders to assess the reliability and validity of chosen attributes (Pressman, 2014). Whether stemming from empirical studies, literature reviews, or direct user interactions, this knowledge helps gauge the relevance and robustness of quality criteria (Kitchenham & Charters, 2007). Such understanding ensures alignment with user needs, organizational goals, and the broader context (Bass et al., 2012). In essence, a clear grasp of the derivation process enhances the credibility and effectiveness of quality considerations in software development (Sommerville, 2011).

RQ1.5 How are conflicting quality requirements managed and resolved?

IEs are complex systems and present significant challenge due to the intricate trade-offs and dependencies inherent in these systems (Augusto et al., 2013). Quality attributes, such as performance, security, usability, and reliability, are interconnected and often trade-off against each other. Enhancing one attribute may inadvertently compromise another (Rodríguez-Domínguez et al., 2022). For example, optimizing performance by increasing system speed might lead to increased resource utilization and potential security vulnerabilities. IEs typically involve a diverse set of stakeholders with varying needs and expectations. End-users may prioritise usability and ease of use, while administrators may emphasise

security and robustness. Balancing these conflicting stakeholder requirements becomes challenging, requiring careful negotiation and compromise. IEs operate in dynamic and evolving contexts, where requirements may change over time. Adapting to new user needs, technological advancements, or emerging security threats may necessitate adjustments in quality attributes. Enforcing a rigid stance on certain attributes may hinder the system's ability to evolve and meet changing demands. IEs often encounter unforeseen challenges and uncertainties during operation, commonly referred to as "unknown unknowns" (Jones & Bonsignour, 2011). Enforcing specific quality attributes without anticipating and addressing these unknowns may lead to system vulnerabilities and unexpected failures. Therefore, we argue that striking a balance between conflicting quality attributes is key to success of IEs. However, we noted a paucity of studies focused on conflicting quality attributes. [P7] highlights that timeliness, reliability and ease of use need to be managed carefully. [P31] discusses trade-off between data usefulness and privacy protection. Maciel et al. (2022) highlight that edge devices which are commonly found in IoT environments contribute to reliability and availability. However, there remains a need to investigate methods for enhancing security and privacy on these resource-constrained devices, or to find a balance with energy consumption to address these attributes effectively. Mohammadi and Javidan (2022) propose a tool to tackle quality of service issues in software defined networks found in IEs by managing efficiency and survivability.

RQ1.6 Which system development methodology is used to develop IEs with a focus on quality?

Out of the 33 scrutinised studies, only a limited subset has delved into system development methodologies with a specific emphasis on quality considerations. Notably, [P16] contends that enhancing the elicitation of UX requirements necessitates the adoption of an Agile or iterative approach, underscoring the complexity of defining these requirements in comparison to usability aspects. The authors advocate for active involvement of end-users in this iterative process. In [P21], an innovative methodology grounded in Unified Modelling Language is introduced to facilitate the accurate design and analysis of AAL solutions. Additionally, [P23] proposes a model-driven approach for effectively modelling the quality of context information in pervasive systems. The literature collectively highlights lack of a tailored methodology to guide developers in engineering higher quality IEs. Prioritising quality in the system development methodology for IEs is essential to ensure their functionality, reliability, and positive impact on society, while addressing the unique challenges posed by the complexity and dynamic nature of these environments.

3.7.2 RQ2 – How is quality of IEs evaluated?

RQ2.1 How are quality requirements measured?

The analysed articles collectively offer a multifaceted exploration of how quality is measured in IEs, encompassing diverse methodologies and perspectives. In [P1], the authors identified the top ten most popular metrics for Object-Oriented Programming (OOP) based on a study by Nuñez-Varela et al. 2017. This provides a quantitative lens on code quality and offers insights into the industry's prevalent practices. However, this approach might be limited in capturing the full spectrum of software quality, especially considering the evolving nature of programming paradigms beyond OOP. The proposal to build a specific approach for usability testing in ubiquitous systems showcases a commitment to a holistic

evaluation process [P2]. The establishment of a software process for context-awareness testing, the definition of interoperability measures, the design of context-awareness test cases, and the development of support tools collectively aim to address various dimensions of software quality. However, the comprehensive nature of this approach may introduce challenges such as increased complexity, resource demands, and potential resistance from development teams. A noteworthy theme is the evaluation of AAL technology, focusing on measuring the efficacy through the quality of data generated by AAL systems [P3]. This involves assessing structured or semi-structured data across dimensions like accuracy, completeness, timeliness, and interpretability. The use of both quantitative and qualitative methods demonstrates a consolidated approach to data quality, acknowledging the need for a multifaceted evaluation. However, the exclusive focus on specific data dimensions may overlook other aspects crucial for AAL system performance.

The adoption of the Goal-Question-Metric approach to map low-level code-based metrics to high-level software quality factors signifies a bridge between detailed code analysis and overarching quality assessment [P4]. This method enables quantifiable measurements, facilitating the comparison of quality across different software systems and components. Nevertheless, challenges may arise in defining metrics that accurately represent the desired quality factors, potentially introducing biases. The exploration of software product quality metrics for Context-aware Computing extends the measurement scope to consider context-aware applications [P5]. The proposed quality-aware cross-layered framework for IoT applications acknowledges the intricate layers involved in IoT development. However, the challenge lies in the practical implementation of these frameworks, as achieving modularity, distribution, seamless integration, and transparency may be context-specific and difficult to generalize. The ENACT DevOps Framework introduces novel solutions to address challenges in developing, operating, and assuring the quality of distributed smart IoT systems [P10]. However, the effectiveness and adaptability of this framework across diverse infrastructures require empirical validation (White et al., 2017). The proposed quality in use model for AAL systems aligning with ISO/IEC 25010 (2021) underscores the importance of user-centric quality assessment [P12]. However, the application of predefined standards may limit the model's ability to capture the uniqueness of AAL contexts and user experiences. The refinement of the ISO/IEC 25010 (2021) quality model for Industry 4.0 needs signifies an attempt to tailor existing standards to specific industrial requirements [P20]. While providing actionable support for software engineers, the applicability and generalisability of the model may be contingent on industry-specific conditions. The exploration of data quality characteristics for AAL systems, guided by ISO/IEC 25012 (2021) and ISO/IEC 25010 (2021) standards, seeks to establish relevant quality characteristics [P24]. However, the challenge lies in identifying universally applicable characteristics given the diverse nature of AAL systems.

In the comparison of quality models, [P25] suggests applying measures originally defined for ubiquitous systems to IoT applications. This points to an effort to adapt established metrics to different contexts, recognizing the commonalities between IoT and ubiquitous applications. However, challenges may arise in ensuring the relevance and accuracy of these metrics when applied outside their original domain. The emphasis on a data-driven approach to improve User Experience (UX) through a case study signals an industry-oriented effort to integrate data analytics into UX evaluations [P16]. However, the effectiveness of this approach in diverse UX contexts and its generalizability require validation. Similarly, proposing a quality in use model for AAL systems, focusing on effectiveness, efficiency, satisfaction, freedom from risk, and context coverage, demonstrates an attempt to align software quality assessment with specific application domains. However,

the adaptation of the model to diverse AAL contexts may be a potential challenge. [P18] presents heuristics for evaluating the usability of ubiquitous systems. This adds a qualitative dimension to the measurement of software quality. The heuristics offer guidelines for assessing effectiveness, efficiency, and satisfaction, emphasizing a user-centric approach. However, the subjectivity inherent in qualitative assessments may introduce variability in the interpretation of usability.

The proposal of a comparative study of existing quality models of interoperability and the introduction of a hierarchic quality model for interoperability in IoT reflect an effort to standardise and define metrics for assessing interoperability [P29]. The challenge lies in establishing universally applicable criteria for interoperability, given the diverse nature of IoT applications. Additionally, the consideration of metrics based on ISO standards adds a level of standardisation but may introduce challenges in adapting these metrics to specific IoT contexts [P30]. The acknowledgment of limitations, such as the potential loss or alteration of information during the translation of quotes, emphasises the importance of considering reliability in the measurement process. This highlights the need for standardised approaches in data collection and reporting to ensure the consistency and accuracy of measurements.

In summary, the exploration of how quality is measured in the articles reflects a dynamic landscape, encompassing quantitative and qualitative methodologies, context-specific adaptations, and ongoing efforts to standardise metrics. The diverse perspectives and approaches underscore the complexity of software quality assessment, necessitating a balanced consideration of multiple factors and potential challenges in the measurement process.

RQ2.2 How is quality of IEs evaluated?

In [P1, P19, P20, P23, P24, P25, P26, P29, P30, P33], we note that the focus was on developing and proposing quality models for evaluating various aspects of IoT, ubiquitous systems, and cloud services. This included security, interoperability, context quality, and the adaptation of quality models to specific domains like Cloud IoT and Industry 4.0. We note the focus is more on measuring certain aspects of IEs. Some studies have presented quality models, evaluation methods, and indices specifically for AAL systems, aiming to assess their efficacy, quality-in-use, and data quality [P3, P7, P12, P13, P17, P24, P28]. [P2, P4, P5, P18, P19, P25] discuss challenges, metrics, and heuristics for usability testing and HCI quality evaluation in ubiquitous systems with a focus on modularity, context-aware computing, and the design of usability tests that consider context-awareness factors. Quality-of-Experience in IoT systems has been investigated in [P6, P27, P31, P32]. [P8, P10, P11, P14, P15, P16, P21, P22] encompass the development, management, and the evaluation of smart IoT systems and explore methods to ensure their reliability, dependability, and user experience requirements. A quality-in-use model grounded in ISO/IEC 25010 (2021) for AAL systems is presented in [P17]. It was utilised as a guiding framework during the assessment of an intelligent solution. Conversely, other proposals, as indicated by [P5, P7], are still in their early developmental stages. We note there is a paucity of studies which investigate how quality of IEs is evaluated.

RQ2.3 During which phase of system development are the quality requirements measured?

In most studies, quality requirements are typically measured post development or during runtime. Sommerville (2011) reflects that this is done to reflect the actual user experience and to study system behaviour in production environment. Pressman (2014) argues that this

leads to a comprehensive assessment of the final product quality characteristics. However, the main limitations of this approach are that it may result in late discovery of quality issues and when significant resources have been invested (Boehm, 1981). [P4, P21] suggest incorporating quality measurements during the design phase. Budgen (2003) believes that this favours early identification and mitigation of potential quality issues during design stage. Both Boehm (1981) and Jones and Bonsignour (2011) concur that addressing issues early in the development process is generally more cost-effective than later. Post delivery software change was about 100 times as expensive as requirements-phase software system for large systems and a ratio of 5:1 for smaller systems (Boehm, 1981). More recent evidence also seems to support these findings and it is recommended that higher investments in early requirements and architecture verification and validation can significantly reduce the high ratio of 100:1 (Oram & Wilson, 2010). It is worth noting that [P4] lacks empirical evidence, and the work was still ongoing. On the other hand, [P5, P8] advocate for measurements during the development process, while [P15] proposes incorporating them during the modelling stage. There is evidence that measuring quality during the development stage leads to early identification and resolution of quality concerns (Sommerville, 2011). Kan (2002) highlights that this encourages continuous refinement and improvement of the product as development progresses. However, the main drawback as pointed out by Kan (2002) is that it may require additional resources and effort. Similarly, incorporating quality measurements during design ensures that the quality considerations are well integrated into the system. However, measurement during design may be less grounded in practical implementation.

3.8 RQ3 – What are the challenges and future research directions?

Several key challenges were identified in the reviewed studies. Bezerra et al. (2014) argue for usability testing in real-life environments for ubiquitous systems, emphasizing the need for meticulous test case design to anticipate various potential contexts. Weyns et al. (2018) stress risks in automated decision-making, calling for improved UX requirements and cautioning against late hardware changes in agile processes. The literature review reveals a scarcity of research in specific areas (Hamzah et al., 2018), prompting a call for experimental testing and the development of an effective Quality of Experience (QoE) framework (Shin, 2017). A study in China emphasizes the need for more humanistic care for the elderly, suggesting that technology falls short in meeting their spiritual needs (Chen et al., 2023). Goncalves et al. (2022) plan to use the Technology Acceptance Model for assessing the usability of a proposed tool among professional developers. There is a persistent challenge in accurately assessing smart systems both functionally and in terms of usability, with a recommendation for developing standard evaluation frameworks (Amiribesheli & Bouchachia, 2018). Communication about quality among team members lacks a standardized approach, with common strategies like unit tests and test cases posing uncertainty about exhaustiveness and potential cost implications. A need for more evaluations in real life scenarios has also been identified.

3.9 Research question and proposition

Based on our analysis of the literature, we note that the state of the art in software technology does not yet present a well-established and widely accepted framework or methodology

for engineering high quality IEs. There is lack of support and guidance during the systems development lifecycle. The definition, measurement, and management of quality during the development process remain unclear. While various studies propose methods for evaluating IEs, empirical validation of these methods in industry is limited. In response to these identified gaps, we propose a framework that includes a quality-enhanced methodology and an IE-specific quality-in-use model. This framework aims to provide guidance for engineering higher-quality IEs. Consequently, the research question of this study was formulated as follows:

> *How can the process of engineering higher quality intelligent environments be improved through the integration of a framework, with a specific focus on improving the specification of quality requirements and the evaluation of quality throughout development?*

The SLR emphasises the lack of clarity in defining, measuring, and managing quality during the development process. Drawing on the insights gained from the SLR, we proceeded to formulate propositions and related questions aligned with the research question. These propositions serve to refine and guide the research focus.

The literature analysis underscores the ambiguity surrounding the specification of quality. In the context of complex systems, developers must explicitly address the specification, prioritisation, and metrication of quality characteristics (Gilb, 2005). Consequently, the initial proposition of this study is designed to explore the current practices in specifying quality requirements for IEs within ongoing projects.

Proposition 1 *Current projects in IE domains do not capture quality requirements adequately in their specifications.*

Questions were then formulated to examine comprehensively the first proposition by addressing various dimensions of quality requirements in IE domain projects. These included initiation, documentation, stakeholder collaboration, monitoring, conflict resolution, historical context, and evaluation phases.

- Are quality requirements captured prior to the case study?
  This question seeks to understand the timing of capturing quality requirement. It aims to explore whether these requirements are considered since inception of a project or later during development.
- How are quality requirements specified in the current system specifications?
  This question delves into the methods and processes for specifying quality requirement in the current system specifications. It provides insights into the documentation and communication practices.
- Are stakeholders' vision taken into consideration when specifying the quality requirements?
  This is a crucial question which addresses the consideration of stakeholder involvement. It aims to understand whether the expectations of key project stakeholders are considered when defining quality requirements.
- How is quality tracked during the development process?
  The rationale for this question is to uncover the mechanisms and tools used for monitoring and tracking quality aspects throughout the development lifecycle. It provides insights whether and how quality standards have been maintained.

- What is the strategy for managing conflicting quality requirements?
  This questions sheds light on strategies employed to handle conflicting quality requirements and how these were reconciled.
- Was any previous benchmark data available on quality aspects of the system?
  The main purpose for this question was to investigate whether historical benchmark data related to quality aspects of the system exist. It seeks to understand the project reliance on past performance metrics and benchmarks.
- How is system evaluated during and post development?
  This question looks at the evaluation processes employed both during and post development. It provides insight into the ongoing assessment of the quality of the system.

The literature exposes the absence of a suitable methodology for engineering higher quality IEs. As a result, the second proposition seeks to investigate the effects of implementing a quality-oriented methodology in the development of IEs and whether such an approach contributes to an enhancement in the overall quality of these systems. It is emphasised in the literature that continuous monitoring of quality characteristics throughout the system development lifecycle is essential for creating systems that align with their stringent quality requirements (Gilb, 2005; IPA, 2010). The lack of a dedicated tool for managing quality characteristics during the development of IEs is a notable observation, leading to the formulation of the second proposition.

Proposition 2 *A quality enhanced methodology will lead to development of higher quality IEs.*

To examine the second proposition, various aspects such as stakeholder engagement, resource implications, developer perspectives, conflict resolution, and the ultimate impact on the quality of the developed IEs were investigated. These questions cover both the qualitative and quantitative dimensions and tries to offer a well-rounded understanding of the effectiveness of the proposed quality-enhanced methodology.

- How are stakeholders' feedback captured during the development process?
  This question studies the involvement of stakeholders throughout the development process and focuses on how their feedback is collected. It aims to provide insights into the responsiveness of the methodology to stakeholders needs and expectations.
- What is the impact of specifying quality requirement(s) for every functional requirement on development time, cost, and overall quality?
  This question explores the potential trade-offs involved in specifying quality requirements for each functional requirement and assesses the impact on development resources, time and the overall quality of developed IEs.
- How effective is the methodology to developers?
  This question assesses the perception of developers regarding the effectiveness of the proposed methodology. It aims to provide insights into its practicality and feasibility in the development environment.
- What is the impact on development cost and time using the proposed methodology?
  This question builds on the second question and seeks to quantify and understand the specific effects of the methodology on development costs and timelines, helping to evaluate its economic implications.
- How are conflicting quality requirements managed?

This question addresses how the enhanced methodology deals with potential conflicts while ensuring alignment of diverse quality expectations.

- Does application of the methodology result in development of higher quality IEs?
  This question aims to determine whether the application of the quality-enhanced methodology indeed leads to the development of higher quality IEs.

The ISO/IEC 25010 (2021) quality-in-use model has gained prominence as a widely adopted approach for assessing overall system quality, as revealed in the literature. This model is versatile and can be customized to align with the specific characteristics of the system under examination. Consequently, the third proposition delves into the examination of the influence of an IE-specific quality-in-use model in evaluating the quality of a system throughout both its development and post-development phases.

Proposition 3 *An IE specific quality-in-use model is beneficial to evaluate quality of IEs during and post development.*

To address key aspects of the third proposition, questions which focused on the specific qualities and benefits associated with the proposed IE specific quality-in-use model were formulated. These investigated the relevance of quality characteristics, examined stakeholder visibility, and assessed the effectiveness of the model to developers. The questions aim to provide a comprehensive understanding of the impact of the quality model on the evaluation of IE quality during and after development.

- How relevant is the proposed mandatory list of quality characteristics?
  This question evaluates the relevance of the mandatory list of quality characteristics proposed by the IE specific quality-in-use model. This is important because it helps to understand their effectiveness in capturing essential aspects of IE quality.
- Does the quality-in-use model provide more visibility about the quality of the system to stakeholders?
  This question seeks to explore the communicative aspect of the quality-in-use model and whether it enhances visibility for stakeholders. It provides insight into its effectiveness in conveying the system quality.
- How effective is quality-in-use model to developers?
  This question assesses the practicality and utility of the quality-in-use model from the perspective of developers. It aims to understand whether the model is effective in guiding development efforts.

These propositions aim to address the research question by exploring different aspects of how the integration of a framework can contribute to improving the engineering of higher quality IEs, with a specific focus on quality requirement definition, specification, and measurement throughout development.

4 Proposed quality-in-use model and methodological framework

In this section, we present the methodological framework and quality-in-use model to develop and evaluate higher quality IEs.

4.1 A quality-in-use model for IEs

We propose to evaluate the quality of Intelligent Environments (IEs) through a refined "quality-in-use" framework, which we define specifically for IEs as " the degree to which a product or system can be used by specific users to meet their needs and accomplish specific goals." This definition is rooted in the universally recognised ISO/IEC 25010 (2021) standard, which we adapt to address the unique demands of IEs. Building on the work of Erazo-Garzon et al. (2021) and Salomón et al. (2023), who respectively adapted ISO/IEC 25010 (2021) for AAL systems and context-aware software systems, we propose a novel quality-in-use model tailored for IEs. Our adaptation process is detailed, ensuring clarity in how metrics are defined and applied:

1. Adaptation of the Generic ISO/IEC 25010 (2021) Model:
   - We assessed each ISO/IEC 25010 (2021) characteristic for its relevance to IEs.
   - We decided to retain characteristics like effectiveness, efficiency, and satisfaction.
   - We removed or updated certain attributes that were less applicable to the IE context.
2. Integration of IE Principles:
   - We mapped nine fundamental IE principles directly to relevant ISO/IEC 25010 (2021) quality characteristics.
   - For principles without a direct ISO/IEC 25010 (2021) counterpart, we introduced new sub-characteristics.
   - Each new or updated quality characteristic was clearly defined, ensuring relevance to IEs.
3. Definition of Specific Metrics:
   - For each quality characteristic and sub-characteristic, we established clear, measurable metrics from ISO/IEC 25023:2016 official documentation (ISO/IEC 25023, 2022).
   - These metrics were developed to be IE-specific, ensuring they are tangible and relevant.

We followed an iterative process involving multiple refinement cycles to ensure comprehensive and consensus-based metric development. Various models were presented to the two co-authors. This process continued until consensus was reached among all three authors regarding the quality-in-use model depicted in Fig. 5. The finalised model incorporates adjustments to context completeness and flexibility while maintaining all sub-characteristics within freedom from risk. Additionally, seven new sub-characteristics, aligned with the remaining seven principles, were integrated under the categories of effectiveness, efficiency, and satisfaction. The measurement functions for these quality characteristics were then rigorously defined, adhering to the ISO/IEC 25023 (2022) Quality Measurement Framework standards. Each metric was constructed to be measurable, relevant, and specific to the IE context, ensuring practical applicability and clarity. We employed a clear scale, from 0 to a defined maximum, where values closer to the maximum indicate higher quality-in-use. The proximity to 1.0, for instance, signifies exceptional performance. Table 5 provides a summary of each metric, its scale, and the target values for higher-quality performance. By incorporating these details directly into the text, we aim to provide a clear, actionable, and transparent framework for evaluating the quality-in-use of Intelligent Environments, offering stakeholders a detailed and practical tool for assessment.

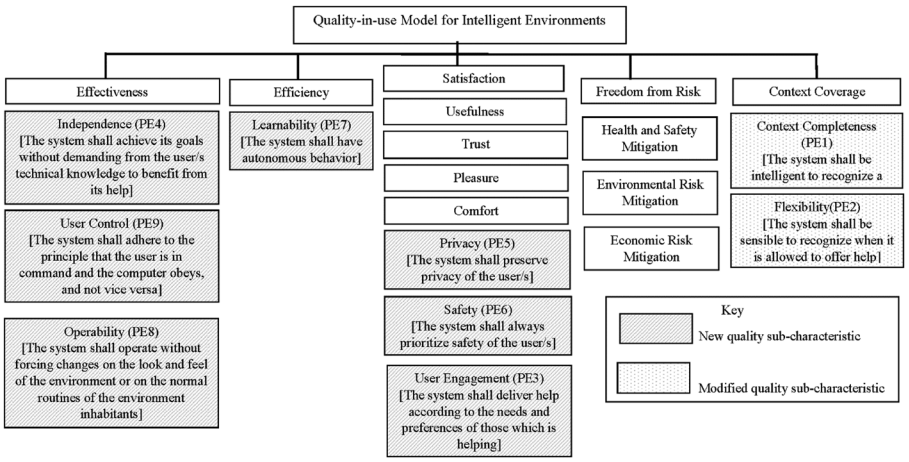


Fig. 5 Quality-in-use Model for IEs

Table 5 Measurement Functions for Proposed Quality-in-use Model

Principle	Measurement function
PE1	$\mathbf{X = A/B}$
	A-Number of acceptable contexts correctly identified by the system
	B-Total number of distinct contexts of use
PE2	$\mathbf{X = (X_1 + X_2 + X_3 + X_4)/4}$
	$\mathbf{X_1 = A_1/B_1}$
	A_1-Number of situations of help correctly identified by the system.
	B_1-Total number of distinct situations when help is needed.
	$\mathbf{X_2 = A_2/B_2}$
	A_2-Number of situations when system correctly recognised the user.
	B_2-Total number of distinct situations when user recognition is required.
	$\mathbf{X_3 = A_3/B_3}$
	A_3-Number of situations when system correctly recognised the user's preferences.
	B_3-Total number of distinct situations when user recognised user preferences.
	$\mathbf{X_4 = A_4/B_4}$
	A_4-Number of situations when system exhibits empathy with the user's mood and overall situation
	B_4-Total number of distinct situations
PE3	$\mathbf{X = A/B}$
	A-Number of situations when system correctly offered assistance to the user
	B-Total number of distinct situations when user assistance is required
PE4	$\mathbf{X = A/B}$
	A-Number of situations when system correctly achieved its goals without user intervention.
	B-Total number of distinct situations when system should achieve its goals without user intervention.

Table 5 (continued)

Principle	Measurement function
PE5	**$X = A/B$**
	A-Number of privacy features configurable by the user
	B-Total number of distinct user configured privacy features
PE6	**$X = 1 - A/B$**
	A-Number of safety failures
	B-Total number of distinct safety features
PE7	**$X = (X_1 + X_2 + X_3)/3$**
	$X_1 = A_1/B_1$
	A_1-Number of successful cases
	B_1-Total number of distinct scenarios
	$X_2 = 1$ or 0
	1 – System adapts and behaves correctly.
	0 – System does not adapt.
	$X_3 = 1\text{-}A_3/B_3$
	A_3-Number of times system had to be re-programmed to achieve autonomy under given scenario.
	B_3-Total number of distinct scenarios
PE8	**$X = 1$ or 0**
	1 – no changes were required
	0 – changes were required
PE9	**$X = (X_1 + X_2 + X_3 + X_4)/4$**
	$X_1 = A_1/B_1$
	A_1-Number of times when system correctly achieved its goals without user intervention.
	B_1-Total number of distinct situations when system should achieve its goals without user intervention.
	$X_2 = A_2/B_2$
	A_2-Number of preferences successfully imposed by the user.
	B_2-Total number of distinct preferences provided by the system.
	$X_3 = A_3/B_3$
	A_3-Number of times when users successfully managed to undo decisions from the system.
	B_3-Total number of distinct situations when users can undo decisions from the system.
	$X_4 = 1$ or 0
	1 - can disconnect
	0 – cannot disconnect

4.2 Proposed methodological framework: UCIEDP2

The initial choice for guiding the development of Intelligent Environments (IEs) was the User-Centred Intelligent Environment Development Process (U-CIEDP), as introduced by Augusto (2014). However, it has been observed that U-CIEDP lacks a well-defined

Table 6 Enhancements to Initial Scoping Stage

Sub phase	Proposed new Activities
Interview stakeholders	1. Establish vision of the project. 2. Gather critical success factors from stakeholders: time, cost and scope.
Define required services	3. Prepare an initial draft of the requirements specification. 4. For each functional requirement, propose quality characteristics and measures from ISO/IEC quality standards. 5. Agree on the mandatory quality requirements defined in the proposed quality-in-use model.
Define required infrastructure	6. Develop an IET to record past, goal and actual values for every quality characteristic. 7. Validate the requirements, quality characteristics and goals with stakeholders. 8. Compile a requirements specification document.
Initial design and prototyping	9. Propose design ideas in the IET. Use data from previous projects as guide. 10. Evaluate and agree on a design idea with stakeholders. 11. Update requirements specification document.

strategy for effectively managing quality requirements throughout the development process, as noted by (Augusto et al., 2018; Ogbuabor et al., 2021; Santokhee et al., 2019). In response to this limitation, we have enriched the U-CIEDP with new activities tailored to each of its core phases: Initial Scoping, Main Development, and IE Installation, as detailed in Tables 6, 7, and 8. This enhanced methodology, dubbed UCIEDP2, is systematically illustrated in Fig. 6 and embodies a more quality-centric approach:

1. Initial Scoping: UCIEDP2 commences with collecting stakeholders' visions and requirements via interviews, alongside documenting crucial project characteristics — time, cost, and scope. An initial set of functional requirements is established, inviting stakeholders to define specific quality characteristics. These characteristics are anchored at the functional level to enable precise quantification and control, informed by the research of Brodie and Woodman (2009) and Fenton and Bieman (2014), and measured according to standards like ISO/IEC 25010 and ISO/IEC 25012 (ISO/IEC 25010, 2021;

Table 7 Enhancements to Main Development Stage

Sub phase	Proposed new Activities
Design	1. Prepare design specifications for selected design. 2. Check and address design defects. 3. Prepare detailed test specifications. 4. Validate with stakeholders.
Implementing and testing	5. Run simulations on proposed system in a test environment. 6. Record actual values in IET for every quality characteristic. 7. Check and address code defects.
Verify correctness	8. Evaluate system using proposed quality-in-use model. 9. Validate IET with stakeholders. 10. Update requirements and design specifications, as necessary.

ISO/IEC 25012, 2021). We identified that in U-CIEDP, this phase lacked concrete steps for integrating quality requirements from the outset. In UCIEDP2, we have introduced specific activities such as 'Establishing Project Vision' and 'Gathering Critical Success Factors' which ensure that quality is considered from the earliest stages. Each activity is directly linked to overcoming U-CIEDP initial shortcomings, offering a detailed methodology for capturing and documenting stakeholder quality expectations based on ISO/IEC quality standards.

2. Quality and Requirements Documentation: Targets for each quality characteristic, per functional requirement, are defined in collaboration with stakeholders and captured in a customized Impact Estimation Table (IET), adhering to Gilb's principles (2005). An example of an IET can be found in Appendix C, enhancing reproducibility, and ensuring clear, actionable guidelines for quality assessment. Following this, prototypes and design concepts are developed, rigorously evaluated against the set quality targets, cost, and time constraints. Moving beyond the broad guidelines of U-CIEDP, UCIEDP2 recommends precise quality metrics, drawing from ISO/IEC 25010 (2021) and ISO/IEC 25012 (2021) standards. The IET is designed for tracking and evaluation of these metrics throughout the development lifecycle.
3. Main Development: This phase involves developing detailed designs and test cases using appropriate design methodologies, with iterations as needed. Implementation follows, preferably in short, incremental cycles, prioritising high-importance requirements and integrating regular stakeholder feedback to ensure alignment with quality objectives. Code verification employs methods such as test-driven development and code reviews (Jones & Bonsignour, 2011). Post-implementation, the system undergoes comprehensive testing and quality evaluations against the IE quality-in-use model, with results documented in the IET to identify any discrepancies from set targets, as outlined in Table 7.
4. Post-Deployment Assessment: The final phase focuses on assessing the deployed system performance from the users' perspective, incorporating user acceptance testing based on the established quality-in-use model. Findings, along with any variances in quality attributes, are systematically documented in the IET. This stage's specific activities are enumerated in Table 8.

By integrating these modifications into UCIEDP2, our aim is to present a model that is not only robust and quality-focused but also transparent and replicable. We are committed to fostering a development lifecycle for IEs that meets functional requirements while

Table 8 Enhancements to IE Installation Stage

Sub phase	Proposed new Activities
Equipment validation	1. Test hardware behaviour in deployed environment. 2. Validate against values recorded in IET.
Software validation	3. Test software behaviour in deployed environment. 4. Validate against values recorded in IET.
Services validation	5. Stakeholders perform user acceptance testing and log all issues. 6. Stakeholders record data in IET.
Interview stakeholders	7. Review all data collected from the validation activities. 8. Team validates data collected with stakeholders. 9. Discuss problem areas and devise strategy to address them.

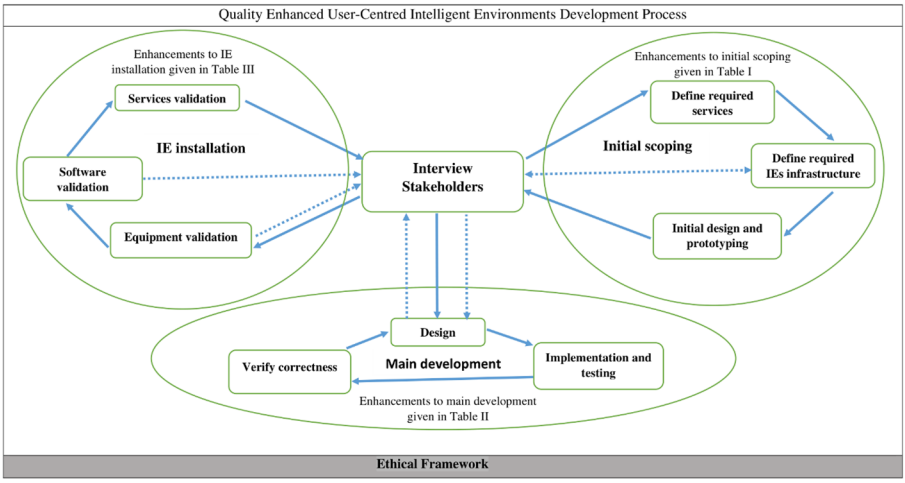


Fig. 6 Quality Enhanced U-CIEDP (UCIEDP2)

excelling in user satisfaction and quality assurance. The enhancements introduced here are designed with the broader research community in mind, offering a blueprint for quality-driven development in this dynamic field.

5 Case study research design

In this study, an enhanced methodological framework, UCIEDP2, has been proposed for engineering high quality IEs. To explore applicability of UCIEDP2 in different contexts, a multiple case study research was deemed more appropriate to allow for generalisation (Murzi, 2007; Staron et al., 2011; Kurtel & Ozemre, 2013; Runeson & Höst, 2009; Sicari et al., 2019; Scott et al., 2021; Tröls et al., 2021; Yin, 2018). Another motivation for using case study research is that we would like to investigate whether the framework would apply to real-world settings (Dalcher & Brodie, 2007). We report on two case studies in this paper: a final year undergraduate project and the other in industry respectively. Design of the case studies was largely inspired by Yin (2018) and consisted of the following five steps:

1. "a study's questions,
2. its propositions, if any,
3. its unit(s) of analysis,
4. the logic linking the [collected] data to the propositions; and
5. the criteria for interpreting the findings.''

The main research question and corresponding propositions underpinning this study are defined in Section 4. The unit of analysis in each project was development of a system using UCIEDP2. As far as the fourth component is concerned, the literature review revealed several questions which were linked to the propositions. Regarding the fifth

component of the case study design, the questions defined for each proposition were instrumental to identify the types of data which had to be collected and strategies to analyse the data. Data for this study was mostly collected from interviews, project documentation reports, test scripts for user acceptance testing and bug reports. Employees in different roles were interviewed. The interviews were carried out as semi- structured interviews and recorded as audio files. The data which was collected was mostly qualitative with some quantitative data such as time, cost, and budget. The main researcher's role in both case studies was to provide support for the application of the UCIEDP2 methodology and to provide an IET for data collection (Alasuutari et al., 2008; Saunders et al., 2016). Informed consent was obtained from all participants involved in the study. Collaboration with the industry partner was made possible due to an existing memorandum of understanding with Middlesex University Mauritius. In the next sections, we describe the two case studies.

5.1 Case study I: smart home monitoring system

The first case study is development of a smart home monitoring system. Figure 7 shows the architecture diagram of the implemented system. Three sensors were connected to an Arduino Nano board. Data collected by the sensors were transmitted to a MySQL database hosted on a Raspberry Pi 4 computer using radio frequency. An Apache Web Server was also installed on the computer. The Web Server hosted a web-based application to monitor energy consumption, and provide recommendations, such as alerts and tips energy for

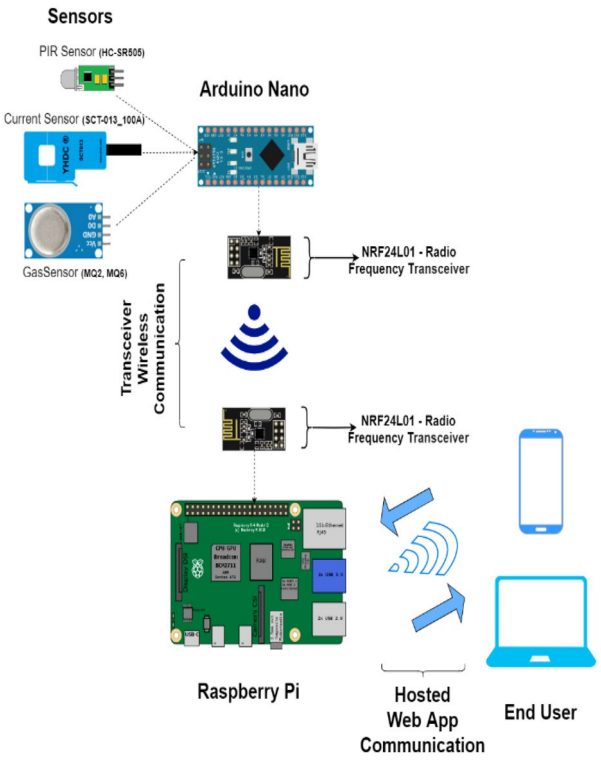


Fig. 7 Architecture Diagram of Smart Home Monitorin System

Table 9 List of functional requirements

Code	Requirement
FR01	The system shall accept input signals from motion sensors
FR02	The system shall accept input signals from gas sensors
FR03	The system shall accept input signals from electric sensors
FR04	The system shall allow wireless connectivity in a local area network setup
FR05	The system shall provide a responsive web-based user interface
FR06	The system shall provide a registration page
FR07	The system shall provide a live graphical display of electricity consumption

saving, using an adapted k-means clustering algorithm. The project stakeholders were a third-year student as developer, a project supervisor, and an experienced business consultant from a software development company as customer. First, we discuss development of the system using prototyping. A kick-off meeting was scheduled at the beginning of May 2020. Since Mauritius was under lockdown during that period due to the Covid-19 pandemic, the meeting was held online. All the project stakeholders participated. The project specification document was examined to determine how the requirements were specified. The initial list of functional and non-functional requirements is given in Tables 9 and 10 respectively. It is worth mentioning that the non-functional requirements were lacking specific metrics and were specified rather vaguely. Prototypes for the web-based application were developed by following Nielsen Heuristics (Nielsen, 1994). Low fidelity mockups were designed by the developer, and these were validated by the business consultant. General feedback and improvements were recorded on paper. Quality requirements were specified as non-functional requirements for the whole system. However, no metrics were specified for the non-functional requirements as given in Table 10. The developer also indicated that test cases were defined for each functional requirements and these were executed towards the end of development. The system was evaluated using the Technology Acceptance Model (TAM) with six constructs and a score of 4.47 was recorded (Sharma et al., 2022).

Table 10 List of non-functional requirements

Code	Requirement
NFR01	The system shall be secure and safe with appropriate security measures
NFR02	The system shall be efficient and fast
NFR03	The system shall be optimised to run all the time
NFR04	The system should be accessible to user everywhere with internet connection
NFR05	The web application shall be intuitive to use

5.1.1 Application of UCIEDP2

An online half-day workshop was held in mid-May 2020 during which the UCIEDP2 methodology and the quality-in-use model were explained to all stakeholders using PowerPoint slides. The stakeholders agreed that the vision of the system was to develop a low cost and easy-to-use web-based application to monitor energy consumption and provide accurate recommendations on energy usage. Upon consultation of the ISO/IEC 25010 (2021) and ISO/IEC 25012 (2021) models, the stakeholders unanimously agreed on a set of quality sub-characteristics for each functional requirement. The details were compiled in an IET, as summarised in Table 11. For brevity, a cut-down sample of the new system specification is given below. In this example, functional requirement FR07 is now quantified as follows:

Functional requirement: FR07
The system shall provide a live graphical display of electricity consumption.
Quality sub characteristic: Usability. Appropriateness recognizability
Measure: Description completeness
Measurement function: X = A/B
A = Number of usage scenarios described in the product description or user documents
B = Number of usage scenarios of the product
0 < = X < = 1
Quality sub characteristic: Usability.Learnability
Measure: User guidance completeness
Measurement function: X = A/B
A = Number of functions described in user documentation and/or help facility as required
B = Number of functions implemented that are required to be documented
0 < = X < = 1

Table 11 Quality measures for functional requirements

Code	Quality sub-characteristics	Measure	Prototyping	Goal	UCIEDP2
FR01	Interoperability	Data formats exchangeability	0.5	1	0.75
	Functional completeness	Functional coverage	0.5	1	0.8
FR02	Interoperability	Data formats exchangeability	0.25	1	0.55
	Functional completeness	Functional coverage	0.63	1	0.83
FR03	Interoperability	Data formats exchangeability	0.75	1	1
	Functional completeness	Functional coverage	0.75	1	0.88
FR04	Interoperability	Data formats exchangeability	1	1	1
	Functional completeness	Functional coverage	0.5	1	0.75
FR05	Time behaviour	Mean response time	600 ms	300 ms	457 ms
FR06	Appropriateness recognizability	Description completeness	0.65	1	0.87
FR07	Learnability	User guidance completeness	0.14	1	0.57
	Appropriateness recognizability	Description completeness	0.43	1	0.79

A second online meeting was convened a week later. This time the developer presented the results recorded for each functional requirement, including the mandatory quality requirements defined in the quality-in-use model. Using this data as baseline, the project stakeholders discussed target values for each of the measures and updated the IET accordingly. One iteration of phase one of UCIEDP2 was sufficient to complete the initial scoping. During the second phase of UCIEDP2, the developer focused on improving the existing system based on available data. Priority was given to functional requirements with very low measure scores. For instance, description completeness was initially measured at 0.14 while user guidance completeness was 0.43 for FR007. These improved to 0.57 and 0.79 respectively in the second prototype. The developer progressively improved the functional requirements. However, significant time was required to research and apply new concepts. Completion of phase two necessitated four iterations. The consultant was involved at the end of each iteration for feedback. Development time increased to 18-person days from 10-person days for the first prototype. The system was then evaluated using the quality-in-use model. Figure 8 shows a comparison of the measurements of the mandatory quality characteristics. All three project stakeholders were then interviewed for feedback at the completion of the project.

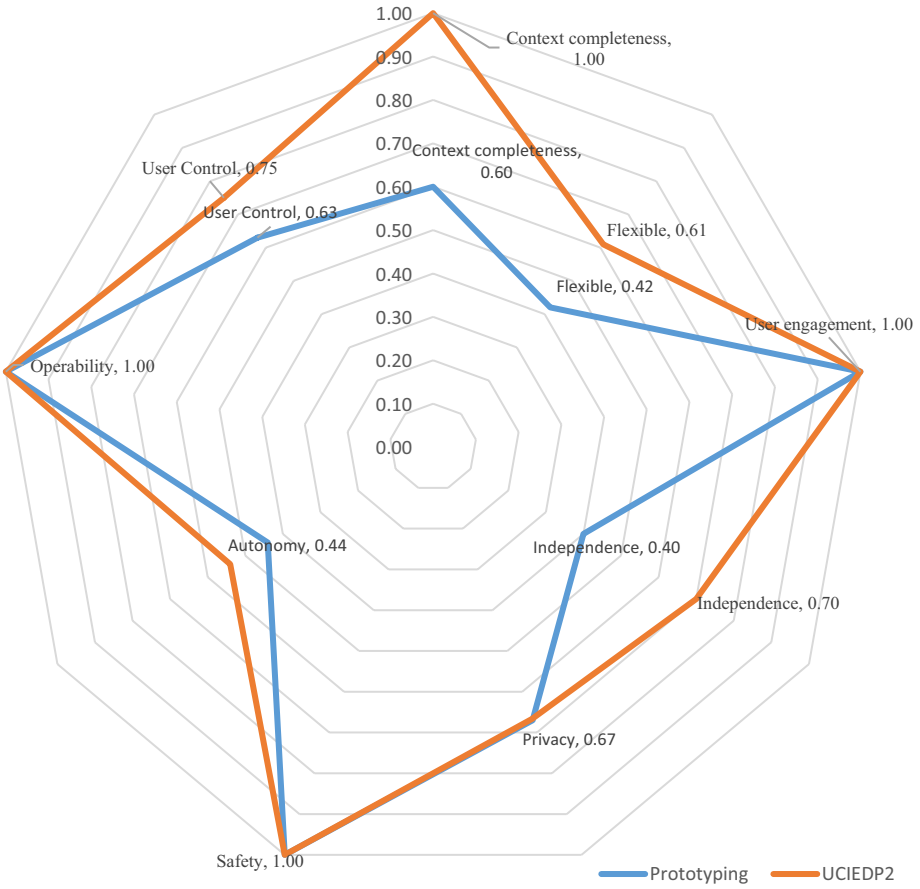


Fig. 8 Evaluation of the Two Systems Using the Quality-in-use model

5.1.2 Findings

Next, we discuss the findings of this case study against the research propositions.

Proposition 1 *Current projects in IE domains do not capture quality requirements adequately in their specifications.*

- Are quality requirements captured prior to the case study?
  Partly
- How are quality requirements specified in the current system specifications?
  They are expressed quite vaguely as non-functional requirements for the overall system.
- Are stakeholders' vision taken into consideration when specifying the quality requirements?
  No
- How is quality tracked during the development process?
  Quality is not tracked during the development process.
- What is the strategy for managing conflicting quality requirements?
  No strategy.
- Was any previous benchmark data available on quality aspects of the system?
  No data was previously available.
- How is system evaluated during and post development?
  User acceptance testing and TAM were administered post development only.

The findings highlight notable deficiencies in the existing method of capturing and defining quality requirements. Preceding the case study, quality requirements were only partially documented and were expressed in a vague manner as non-functional requirements for the entire system. Furthermore, the vision and priorities of stakeholders were disregarded during the specification of these requirements. The tracking of quality throughout the development process was absent, and there was an absence of a strategic framework for handling conflicting quality requirements. Additionally, no foundational data existed regarding the quality aspects of the system, impeding the ability to accurately assess and enhance the system during and after development. The sole evaluations conducted were User Acceptance Testing (UAT) and the Technology Acceptance Model (TAM), administered post-development, potentially insufficient for uncovering all system issues. In summary, these deficiencies underscore the necessity for a more comprehensive and systematic approach to capturing, defining, and assessing quality requirements within IE domains.

Proposition 2 *A quality enhanced methodology* (*UCIEDP*2) *leads to development of higher quality IEs.*

- How are stakeholders' feedback captured during the development process?
  Specifying the vision was identified as a key step. The stakeholders agreed that it helped them to consider relevant quality characteristics which would contribute towards meeting the vision of the system. However, picking the right quality characteristics from the ISO/IEC 25000 (2021) standards and agreeing on the quality targets were quite challenging. The researcher had to schedule a session to explain the ISO/IEC 25010 (2021) and ISO/IEC 25012 (2021) quality models. The developer then measured each functional requirement and recorded the results in an IET (Table 11). Stakeholders

agreed that they were more involved during each stage of development. The consultant highlighted that giving feedback based on the IET worked well and they were able to monitor progress more objectively.

- What is the impact of specifying quality requirement(s) for every functional requirement on development time, cost and overall quality?

  According to the developer, more time was required during each stage of development as more checks and tests were required to ensure quality expectations are being met for each functional requirement. He claimed the number of test cases almost tripled due to increase in number of test conditions. But importantly, the IET gave them clearer picture of realization of quality characteristics for each requirement for the existing version of the application. This allowed the developer to better focus on critical areas which needed improvement. It was also possible to measure the functional requirements using the quality characteristics during design time and monitor progress during implementation. However, since this was a final year project, it was difficult to estimate true cost of development effort as budget was capped at $100 to cover for hardware expenses only.
- How effective is UCIEDP2 to developers as a methodology?

  The developer claimed that the activities for each phase were clearly defined and conducive to manage the quality requirements during development. His main challenge was picking quality characteristics based on the ISO/IEC 25000 (2021) standards. To address this challenge, the main author arranged a session involving all three stakeholders. During this session, the ISO/IEC 25010 (2021) and ISO/IEC 25012 (2021) quality models were explained, drawing reference from official documentation Due to time constraints, stakeholders opted to focus on a select few quality characteristics from the model that could be realistically implemented. Subsequently, a second session was scheduled a week later, during which the stakeholders actively participated and achieved unanimous consensus on the chosen quality characteristics. The requirements were re-adapted by incorporating quality characteristics, as summarised in Table 11. Since functional requirements already existed for the system, only one cycle was required during the initial scoping stage. The developer also evaluated the existing system using the quality-in-use model to obtain initial data for each mandatory quality characteristic. Thereafter, target and current values for each measure were defined and recorded in the IET. During the main development stage, the developer admitted measuring each functional requirement independently during implementation using the quality characteristics. A second round of measurements was carried after all requirements were implemented to check for big deviations from expected results. The developer highlighted that although this strategy worked for this project, clearer guidance could be provided to explain the process of measuring the functional requirements.
- What is the impact on development cost and time using UCIEDP2?

  The main impact was on development time which increased from 10-person days to 18-person days.
- How are conflicting quality requirements managed?

  There were no conflicting quality requirements.
- Does application of UCIEDP2 result in development of higher quality IEs?

  There is a consensus among all those involved in this project that having a list of well-defined functional and quality requirements with target baselines helped throughout the project. The developer highlighted that the baselines contributed to track progress during the development stages. However, fixing some of the design and coding issues alone was very challenging and time consuming for the developer. Significant amount of time was

spent learning and testing new coding concepts to meet some quality expectations. The stakeholders agreed that application of UCIEDP2 resulted in development of a higher quality system compared to the previous version, as evidenced in Table 11. They were also able to choose the solution which delivered the best value in terms of quality, cost and development effort during design and monitor its subsequent implementation.

Proposition 3 *An IE specific quality-in-use model is beneficial to evaluate quality of IEs.*

- How relevant is the proposed list of mandatory quality characteristics?

  The stakeholders were particularly interested in privacy, context coverage and user control out of the nine mandatory quality characteristics. In addition, they chose trust based on the objectives of the project. Nevertheless, the developer evaluated the existing system using the quality-in-use model and recorded data for every mandatory quality characteristic in the IET. In the second version developed using UCIEDP2, priority was given towards improving the selected four quality characteristics. Overall, the developer and consultant agreed that the measures were relevant to measure each quality characteristic. They acknowledged the importance of the remaining mandatory requirements towards developing more acceptable systems. However, due to time and resource constraints these would be improved in future versions. The radar map in Figure 8 captures the degree to which each quality characteristic has been realized using the two methodologies.
- Does the quality-in-use model provide more visibility about the quality of the system to stakeholders?

  Both the consultant and the programmer revealed that the quality-in-use model was instrumental to have a better appreciation of the system quality. They agreed that the data helped to evaluate the system more effectively and objectively from a user perspective. In the previous version, the user acceptance testing, and TAM were only performed after development of the system. The quality-in-use model also allowed for iterative development using data collected as feedback to make more incremental development of hitting the quality targets.
- How effective is the quality-in-use model to developers?

  According to the developer, the quality-in-use model was relatively easy to use. More importantly, it was useful as a planning tool to manage complexity of these types of systems and better manage conflicting quality requirements.

5.2 Case study II: planning solution

The second case study was carried in a mid-sized local company. It has a workforce of 30 employees with nine developers and three testers. The company delivers telematics solutions including geographical platforms for measuring asset performance in real time, fleet management and IoT solutions. Its customer base includes public and private sector authorities, including organisations within the African subcontinent. It made a transition to Agile Scrum from waterfall methodology in 2016. However, the management of the company is keen to explore alternative methodologies which would improve their time to market, allow earlier identification and correction of defects, and reduce volume of regressions and reworks post deployment. The project consisted of developing a planning solution for a new customer. Its purpose was to optimize daily operation of

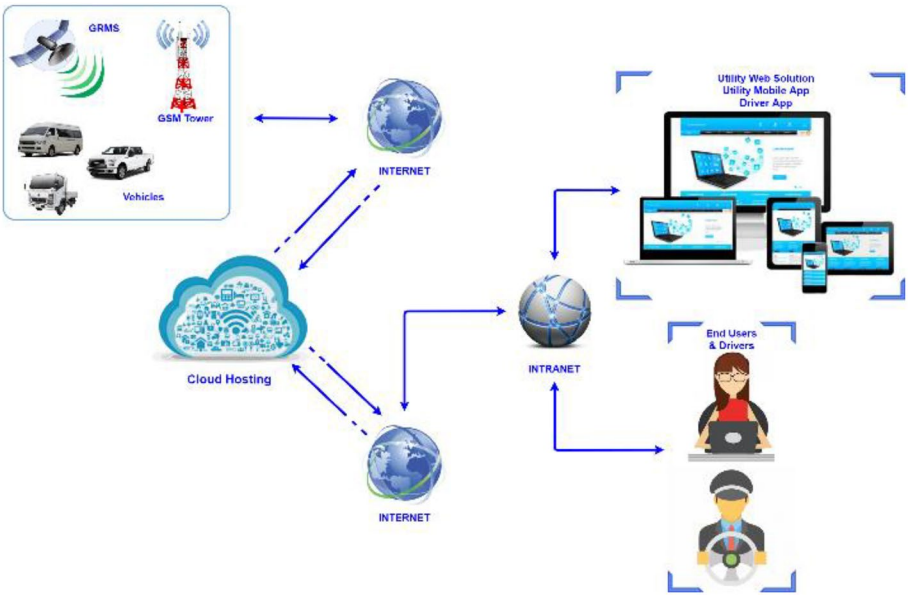


Fig. 9 High level Architecture Diagram of Planning Solution

vehicles, spot route deviation, optimize fuel efficiency and ensure customer satisfaction. The application provides real time information for connected assets. At the core of the platform is a tracking engine. GPS trackers, fuel and machine sensors are installed on clients' vehicles. Positional and engine related information is then transferred over the mobile networks to the tracking engine which corrects any positional discrepancies and false coordinates which might occur due to dropped network connections during transmission. The low-level information is then translated to a database format capable of being used by other applications. The information is rendered in real time using graphical representations that allow users to monitor the performance of the asset. Figure 9 shows the high-level architecture of the solution. This project was selected by the company since a similar one was delivered in March 2019 to a different customer and which would thus serve as baseline for comparisons (Fig. 9).

5.2.1 Application of agile scrum

A kick-off meeting was held in May 2021 with key stakeholders such as the general manager, account manager, chief programmer, and lead tester. It was decided that the study would be conducted in two phases. The first phase consisted of interviewing each team member to obtain background information on current practices regarding system development at the company. Due to time constraints and availability, the interviews were scheduled over a period of one month and were approximately one hour in length each. Following this initial round of interviews, the main observations were summarised as follows. The company has adopted Agile Scrum as development methodology and the team engages in daily stand-up meetings. Customers, normally a project sponsor and an IT manager or technical person, are involved during every stage of the project since the start. Requirements gathering consists of capturing the needs of the clients first. These are translated

to functional requirements by the consultants. Once all requirements have been obtained, the next step is for the project manager to define all the fields, process flows, data capture and screen flows in a software design document, which is then validated by the customer. Development starts following approval from the customer. Test scripts are also written by a test manager for each screen. When development is completed, testing is carried out by the testers. Following this, the system is deployed for user acceptance testing by the actual users. Quality is basically assessed through the testing process. The company does not enforce any kind of best practice with regards to quality. It all comes down to individual expertise of the development team. However, although customers are satisfied with the support and engagement provided, the main limitation is volume of regressions that are needed post deployment, and which affects timely delivery of other projects. Bugs are maintained in the ClickUp (2022) bug tracking software. A decision was also made to recruit the same Scrum team on the new project: two senior developers (5 + years of experience), three junior developers (1–3 years) and three testers.

5.2.2 Application of UCIEDP2

The second phase of the study started in mid-June 2021 and was complete by end of September 2021. It focused on application of UCIEDP2 to a pilot project of similar complexity and size as the previous one. A half-day virtual meeting was conducted to explain UCIEDP2, the new quality-in-use model and the ISO/IEC 25000 (2021) standards to the key stakeholders and the development team. They were then tasked to propose quality characteristics based on the ISO/IEC 25000 (2021) standards. Upon consultation of bug reports of the existing project, performance efficiency was identified as key quality characteristic. Customers also complained that performance of some queries for the first project degraded over time. Therefore, the measures selected unanimously by the team were mean response time and response time adequacy. Out of the 75 functional requirements, 28 were identified as critical. The two measures were defined for each of these 28 requirements. Below is an example for one critical requirement:

Functional requirement: User shall be able to insert planning request either on screen or by mass upload.
Quality sub characteristic: Performance Efficiency. Time Behaviour
Measure: Mean response time
Measurement function: $X = \sum (Ai)/n$
$i = 1$ to n
Ai = Time taken by the system to respond to a specific user task or system task at i-th measurement
n = Number of responses measured
Measure: Response time adequacy
Measurement function: $X = A/B$.
A = Mean response time
B = Target response time specified
$0 <= X <= 1$

The project team consulted the previous product backlog but prioritised the 28 critical requirements first. However, all the developers were first assigned to record baseline values for the two measures for every functional requirement from the first planning project. They were supported by the two testers. Data was recorded in an IET (Table 12)

Table 12 Partial Impact Estimation Table for Planning Solution

Functional Requirement		**Agile Scrum**			**UCIEDP2**		
	Quality characteristic	***Past***	***Goal***	***Actual***	***Past***	***Goal***	***Actual***
User shall be able to insert planning request either on screen or by mass upload.	Performance Efficiency. Time Behaviour. Mean response time (seconds)	-	5s	10s	10s	5s	2s
	Performance Efficiency. Time Behaviour. Response time adequacy	-	1	0.6	0.6	1	0.9

and shared among the entire team. A second meeting, led by the senior developers, was carried a week later to reach a consensus on the quality targets for each functional requirement, including the mandatory requirements for the quality-in-use model. They presented the recorded baseline values and proposed targets for the two measures for each functional requirement. There were some divergences of opinions with regards to the targets among the team members. However, these were resolved though discussions of the discrepancies and identification of any trade-offs that was deemed necessary. The team members also explored compromises and collectively arrived at a consensus on the quality targets. That concluded the initial scoping phase in UCIEDP2. The team decided to adopt a combination of iterative and incremental approach to development. The 15 highest priority requirements were implemented first during the main development phase. The artefact produced was then evaluated using the quality-in-use model in production environment. The IET was updated with data collected and inconsistencies were easily identified, which led to further improvement. Feedback was then sought from the general manager and account manager before work started on the next increment. Overall, after four cycles during development, a complete version of the system was ready for system testing. The two testers then proceeded to release testing of the system using the proposed quality-in-use model. Data on number of regressions and reworks was collected for three releases. Figure 10 shows a comparison between the two planning systems developed using Agile Scrum and UCIEDP2 respectively.

5.2.3 Findings

The project team members were individually interviewed to gather their feedback on UCIEDP2 and the quality-in-use model at the completion of the project. In this section. we discuss the findings of this case study against the research propositions.

Proposition 1 *Current projects in IE domains do not capture quality requirements adequately in their specifications.*

- Are quality requirements captured prior to the case study?
  No
- How are quality requirements specified in the current system specifications?
  Not specified. However, performance testing of the application is done if time permits. Usability is validated during the user acceptance testing. The lead tester provides test scripts to the customers and assists them whenever required.
- Are stakeholders' vision taken into consideration when specifying the quality requirements?
  No
- How is quality tracked during the development process?
  Quality is not tracked during the development process.
- What is the strategy for managing conflicting quality requirements?
  No strategy.
- Was any previous benchmark data available on quality aspects of the system?
  No data was previously available.
- How is system evaluated during and post development?
  User acceptance testing post development only.

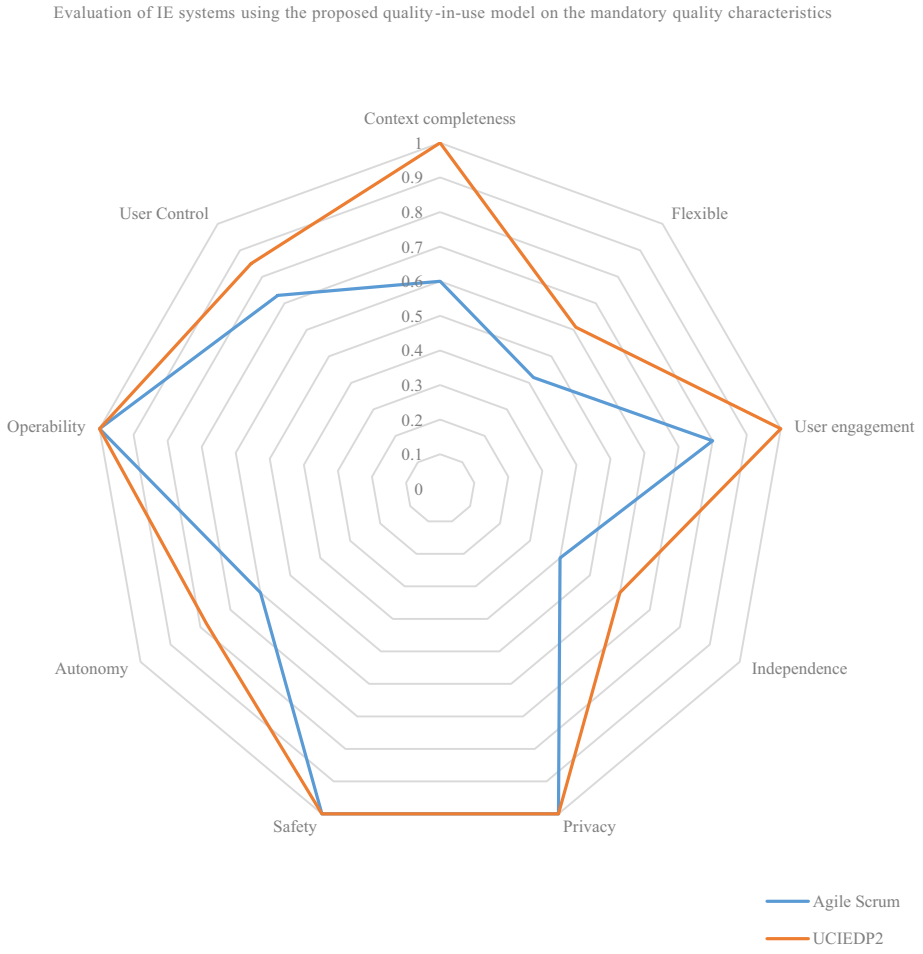


Fig. 10 Evaluation of the Two Planning Solutions Using the Quality-in-use model

Proposition 2 *A quality enhanced methodology* (*UCIEDP*2) *leads to development of higher quality IEs*.

- How are stakeholders' feedback captured during the development process?

  The general manager confirmed that customers are involved during every stage of a project. The team generates the requirements specification, design document and test scripts after capturing the full customised demands of the clients. With the adoption of UCIEDP2, the requirements were quantified. Quality targets were defined following consultation with the project team. The researcher had to schedule a session to explain the ISO/IEC 25010 (2021) and ISO/IEC 25012 (2021) quality models. The accounts manager recorded these metrics in an IET which was then used as basis for assessing potential design solutions more likely to meet the quality targets along with estimated development effort, cost, and time. This session was facilitated by the team leaders,

Table 13 Comparison Between Planning Projects

Characteristic	**Planning Project**	
Methodology	Agile Scrum	UCIEDP2
Number of staff	• General Manager (projects and operations) • Account Manager • Chief Programmer • Lead Tester • 1 person for documentation • 5 developers • 3 Testers	
Number of features	15	
Number of requirements	75	
Number of system tests	334	427
Total effort	21-person days	32-person days
Estimated development cost (Rs)	200 K	275 K
Number of releases	3	
Number of regressions	126	85
Number of reworks	15	9

senior developers, and testers. The potential design options were then presented to the general manager and customers and a consensus was reached on the solution which will likely provide the best value in terms of quality, time, and cost. Following this phase, all the fields, process flows, data capture and screen flows were captured in a design document, which was again validated by the general manager and the customers. Customers' feedback was instrumental to drive implementation of the new system. After each increment, the artefact was demonstrated and resulting quality related data was used as basis for decision making.

- What is the impact of specifying quality requirement(s) for every functional requirement on development time, cost, and overall quality?

  Looking at the Table 12, the functional requirement is placed on the left-hand column. Goal represents the expected target value while *actual* captures the value which was measured. For instance, mean response time for the functionality was noted to be 10s for the system developed using Agile Scrum. A 50% improvement was proposed and the target to achieve was set to 5s. However, after refactoring the codes, the response time the response time further improved 2s. However, the number of system tests increased considerably from 334 to 725 with the introduction of quality characteristics since new test cases were defined. Total effort also increased from 21-person days to 32-person days for the new project. Estimated development cost rose by Rs 75K. Table 13 summarises the two projects on key characteristics.
- How effective is UCIEDP2 to developers as a methodology?

  The three senior developers agreed that while using Agile Scrum, the focus was more on delivering system functionalities after every sprint. Testing was always performed by the quality assurance team at the end. However, with the adoption of UCIEDP2, quality measures were now incorporated with every functional requirement. They reflected that this encouraged them to perform tests early and check quality targets are being met during development. This was also beneficial since it allowed detection of errors early on. On the other hand, since the team was

composed of junior and senior developers, there were some divergences in effort and time required to complete tasks of similar levels of complexity. The two junior developers needed more consultations and support from seniors to resolve technical issues. The two testers reflected that previously they were running test scripts to validate the functional requirements only. However, with UCIEDP2, they also had to measure the quality characteristics for each functional requirement which led to a significant increase in effort and time. However, the general perception was that UCIEDP2 was very effective because everybody had a clearer picture of the project's expectations. However, on the downside, significant time and effort were required and it was tricky to manage multiple projects.

- What is the impact on development cost and time using UCIEDP2?

  Total effort increased from 21-person days to 32-person days. The estimated development cost also was higher, as summarised in Table 13. The general manager highlighted that additional time was required to apply UCIEDP2 and cover for the comparatively higher number of system tests. However, on the positive side, bugs were discovered early during development and there was a decrease in number of reworks over three releases. The number of regressions had also fallen, especially on issues related to response times of queries. According to the general manager, UCIEDP2 offered a better return on investment compared to Agile Scrum on this planning project.
- How are conflicting quality requirements managed?

  Managing performance (response time) and usability in the planning solution was challenging previously. It requires availability of data in real time. However, performance related issues increased as the user interfaces grew more complex. Unfortunately, a lot of these problems emerged during testing and lead to reworks or quick fixes to ship the system on time. With the introduction of UCIEDP2, since performance is now quantified, the senior developers were able to find problems earlier during development. Crucially, they had collected critical data to explain to customers what was technically feasible.
- Does application of UCIEDP2 result in development of higher quality IEs?

  The consensus from the team is that they had more confidence in the quality of the system that was developed. By measuring quality throughout the development process, all five developers agreed that quality was embedded into the system. This is also supported by a reduction in major reworks and regressions post deployment for the new planning project (Table 13). The general manager and accounts manager highlighted that UCIEDP2 gave a sense of direction in terms of design and quality of choices and were able to give more objective feedback to customers. They were thus able to choose the solution which would deliver the best value in terms of quality, cost, and development effort.

Proposition 3 *An IE specific quality-in-use model is beneficial to evaluate quality of IEs.*

- How relevant is the proposed list of mandatory quality characteristics?

  The team acknowledged that evaluation of systems was always performed using user acceptance testing at the end of development. The introduction of mandatory quality characteristics served as a reminder of the critical attributes that systems of this nature should embody. To ensure a comprehensive assessment without resource dispersion

across numerous quality dimensions, the team deliberately chose to concentrate on these nine mandatory quality characteristics for the current project. The quality-in-use model was first applied to the existing planning solution by the developers and testers. The results provided useful insights into the quality of the system. The team noted that quality targets for three out of the nine characteristics were already met.

The developers tried to improve the remaining six characteristics and were successful to some extent. The radar maps in Figure 10 show the degree to which each mandatory quality have been realized using the two methodologies for the planning solutions.

- Does the quality-in-use model provide more visibility about the quality of the system to stakeholders?

  The senior developers pointed out that the quality characteristics revealed critical insights into the quality of their first planning solution. Previously, quality was always left to the quality assurance team. Testing during development was sporadic. However, the quality-in-use model offered tangible and measurable indicators by quantifying the quality aspects. They were also able to assess more objectively their implementation choices, allowing more iterative refinement of the system as development progressed. Moreover, they argued that metrics enabled a comparative analysis over time or across different projects. By establishing benchmarks and comparing metrics across various iterations or projects, teams can identify trends, assess the impact of process improvements, and make informed decisions for future development efforts.
- How effective is the quality-in-use model to developers?

  The implementation of the quality requirements, using the quality-in-use model, allowed the developers to perform a more comprehensive evaluation of the system. The senior developers agreed this level of granularity was particularly beneficial as it allowed them trace problems back to the functional level, gaining insights into the root causes of issues.

  The comprehensive evaluation made possible by the quality-in-use model empowered them take targeted and effective measures to enhance the overall quality of the system. By understanding the specific aspects that require attention, they were able to prioritise their efforts and implement changes that had a meaningful impact on the system performance. However, the two novice developers highlighted that they faced some challenges in applying the quality-in-use model concretely. While they appreciated the level of support which was provided by the senior developers, they expressed a need for more explicit guidelines and practical exercises to enhance their understanding and application of the quality-in-use model. This feedback highlighted the importance of providing adequate resources and guidance, especially for less experienced team members, to maximize the model's effectiveness across the entire development team.

6 Discussions

In this section, we aggregate the findings from the two case studies.

Proposition 1 *Current projects in IE domains do not capture quality requirements adequately in their specifications.*

Ensuring a common understanding of system quality requirements among stakeholders is crucial, given the various interpretations of quality (Jones & Bonsignour, 2011). Both

case studies highlight shortcomings in effectively addressing quality requirements. They were expressed quite vaguely as non-functional requirements encompassing the entire system which is a common practice (Ali et al., 2022; Ruiz-López et al., 2013; Werner, 2022). However, the "one definition fits all" approach doesn't aptly apply to non-functional requirements (Chung et al., 2009). Each IE system has certain unique characteristics and stakeholder expectations which may require more tailored approach to defining and implementing quality requirements. Unfortunately, we note that stakeholders' vision was often disregarded in both case studies. Werner (2022) further argues against applying quality requirements uniformly for the entire system due to potential higher costs. This again highlights the need of adapting quality requirements to the specific characteristics and demands of each system.

Chung and do Prado Leite (2009) further draw attention to a disproportionate emphasis on functional requirements. There is a risk that functional requirements may be neglected or overlooked. The study by Oriol et al. (2020) adds weight to this concern by highlighting that in Agile based developments, quality requirements are often given less consideration compared to their functional requirements. This again raises concerns about the effectiveness of development methodologies in ensuring a balanced focus on both functional and non-functional aspects of a system.

The argument presented by Brodie and Woodman (2009) against separating non-functional or quality requirements from functional requirements further reinforces the idea that these two types of requirements should be interrelated. Integrating quality characteristics with functional requirements, as advocated by Edward Deming (Ghobadian & Speller, 1994), ensures that quality becomes an inherent aspect of the system. System development aligns with developer specifications, but customer expectations must also be met (Jones & Bonsignour, 2011). This approach discourages dissociation of quality and functional aspects, promoting a holistic perspective for effective system development. Both case studies also highlight that the stakeholders' vision was disregarded, and there was a lack of quality monitoring during the development process. The strategic management of conflicting requirements was insufficient, exacerbated by the absence of benchmark data and evaluation methods.

Thus, there is a need to improve how quality requirements are currently addressed in IEs domain projects. The highlighted deficiencies overlook stakeholders' vision and the imbalance in emphasis between functional and non-functional requirements collectively support the proposition that current projects in IE domains do not capture quality requirements adequately in their specifications.

Proposition 2 *A quality enhanced methodology* (*UCIEDP*2) *leads to development of higher quality IEs*.

In the first case study, Prototyping was initially used as methodology (Sommerville, 2011). According to the developer, it gave him the flexibility to explore ideas and seek feedback from the consultant and supervisor. On the other hand, the company had adopted Scrum, which is considered as a prevalent Agile method, as methodology by the company (Hron et al., 2022). The general manager pointed out that this offered several advantages over conventional approaches such as prioritisation of requirements based on business value and delivery of potentially shippable product increments after a sprint (Hanslo et al., 2019). In addition, utilisation of Scrum metrics such as velocity, sprint length and retrospective meetings were used to improve on project delivery objectives (Greening, 2015; Hanslo et al., 2019). The Scrum team members agreed that they were working very closely with customers to refine the requirements and were following recommendations from the Scrum

Body of Knowledge (SCRUMstudy™, 2016) to ensure early value delivery. However, the main issues were related to quality although projects were being delivered on time according to the general manager. In both case studies, testing was the predominant method to evaluate the systems post development. However, it is important to note that testing can only reveal the presence of errors, not their absence. Consequently, some requirements and design bugs persisted even after testing. For example, usability issues lingered in the first case study. In the second case study, reworks and regressions occurred after each release, primarily due to performance-related problems, incurring substantial overhead costs, time, and resources. Management of the company expressed concerns about the excessive time and resources spent on post-development bug fixing and regression handling, impacting other projects due to resource unavailability. Surprisingly, a quantitative analysis of the Scrum Framework found no significant positive correlation between quality and Scrum adoption (Hanslo et al., 2019).

The introduction of the UCIEDP2 methodology mitigated these shortcomings by prioritising stakeholders' feedback during development. Most importantly, the vision of the stakeholders was translated to measurable quality characteristic(s) from the ISO/IEC 25000 (2021) standards. Despite challenges in selecting appropriate characteristics and targets, the use of ISO/IEC 25000 (2021) standards facilitated the specification of quality requirements. Agreed baseline and target values for each functional requirement was recorded in an IET. This also influenced the way the most optimal solution was selected based on cost, time, and quality expectations rather than subjectively. All participants agreed that UCIEDP2 successfully managed quality requirements, aided by the IET for progress tracking. Moreover, the systematic incorporation of stakeholders' feedback during development facilitated the integration of quality measures into each functional requirement, enabling early testing and tracking of quality targets. This was a departure from conventional end-of-development testing which served as the predominant method to assess systems in both case studies.

UCIEDP2 departs from the inflexible nature of plan-based methods and addresses the absence of metrics for measuring quality in Scrum (Sommerville, 2011). It facilitates progress tracking by assessing quality measures during development, encouraging immediate problem resolution rather than deferring it to the end of development. Additionally, it offers flexibility to accommodate various lifecycle approaches, including incremental, prototyping, iterative, or hybrid methodologies. The recommended iterative approach involves using feedback for incremental development to meet the established targets (Humble & Farley, 2010). However, a key challenge is shortening the development cycle to obtain early and regular feedback from customers. Both case studies highlighted that novice developers required more time and support to enhance code compared to their senior counterparts, leading to an increase in testing efforts that could potentially impact development schedules (Gordon & Bieman, 1993). Despite these challenges, the methodology demonstrated advantages such as early bug detection during development and a reduction in rework instances across various releases. These benefits allowed for improved planning when dealing with multiple projects concurrently.

The arguments provide a compelling case for the second proposition suggesting that the UCIEDP2 methodology positively contributes to the development of higher quality IEs compared to initially employed methodologies (Prototyping and Scrum) by prioritising stakeholder feedback during development, specification of quality requirements, progress tracking throughout development, and early testing. While this methodology led to increased development cost and time due to additional tests and efforts, it proved effective for developers generally, offering a comprehensive evaluation of the system and targeted improvements. This approach provided a clearer understanding of project expectations.

Proposition 3 *An IE specific quality-in-use model is beneficial to evaluate quality of IEs.*

As already discussed, quality was initially overlooked in both case studies. Efforts were mostly concentrated towards implementing the functional requirements. IEs are personalised and user-centric systems. They depend on the specific physical environments in which they operate since different users will have different requirements (Banijamali et al., 2020). However, quality aspects related to the personalised nature of IEs such as user control, context coverage, and privacy were not explicitly considered in the early stages. Testing primarily revealed functional errors but lacked comprehensive insights leading to a discrepancy between user expectations and system capabilities.

The application of this adapted quality-in-use model in the two case studies proved to be instrumental in guiding the development processes. The model explicitly outlines quality characteristics relevant to IEs. Developers reported that the defined quality characteristics and metrics facilitated the selection of optimal design ideas, considering factors such as cost, time, and quality. They could also weigh the trade-offs between adapting the system to diverse physical environments. The stakeholders expressed appreciation for the informed decision-making enabled by the UCIEDP2 methodology at each phase. Developers in both case studies acknowledged that the quality-in-use model provided a more objective perspective on the anticipated capabilities of the systems and allowed for system development tailored to specific customer contexts. Moreover, the IE-specific quality-in-use model proved beneficial for evaluating IE quality. Stakeholders prioritised mandatory quality characteristics, such as privacy, context coverage, user control, and trust. The quality-in-use model provided transparency, enabling informed decisions and incremental improvements throughout the development process.

Furthermore, the collection of critical benchmark data represented a notable shift in the company's approach, providing more accurate insights than their prior method of obtaining feedback through user interface demonstrations. Although this change meant increased efforts for developers and testers, involving more efficient code writing and extensive testing throughout development, it proved beneficial in meeting and managing quality levels. Constant monitoring of selected quality attributes allowed developers in both projects to navigate quality trade-offs and minimise the risk of accruing technical debt. A similar observation aligns with the findings of Sas and Avgeriou (2020), who, in their study interviewing developers across various companies developing embedded systems, noted the importance of actively managing quality attributes to mitigate technical debt.

The analysis supports the third proposition and suggests that an IE specific quality-in-use model is indeed beneficial for evaluating the quality of IEs. The tailored approach addresses the unique challenges posed by these personalised and user-centric systems and guides both developers and stakeholders in making informed decisions throughout the development process. However, it required a substantial investment of time and effort, presenting challenges in managing multiple projects simultaneously.

7 Threats to validity

Yin (2018) highlighted several common criticisms of case study research, including lack of rigor, bias, difficulty in generalisation, and extensive, cumbersome documentation. In response to these concerns, it is emphasised that improving the quality of a case study involves adhering to four empirical research tests: construct validity, internal validity, external

validity, and reliability (Yin, 2018). To enhance the robustness and credibility of the research findings, a systematic scrutiny of potential validity threats was conducted at every stage of the case study, drawing on the validity perspectives proposed by Gibbert et al. (2008) and Runeson and Höst (2009). The adoption of these perspectives facilitated a comprehensive evaluation of the study's design, execution, and the generalisability of its findings.

The formulation of the main research question and propositions was informed by a thorough analysis of the literature. However, there is the possibility that the literature review might not have been exhaustive. Data collection, stemming from research questions for each proposition, involved multiple sources in both case studies, including interviews, meetings, project reports, system specification documents, bug reports, and test reports. One challenge to the construct validity is potential misinterpretation of interview questions by participants. To mitigate this concern, the main author conducted a pilot test with the two co-authors to identify ambiguous questions or potential misinterpretations (Creswell & Creswell, 2017). The interview questions were then further refined and are detailed in Appendices D and E, respectively. In both case studies, the main author also carried training and information clarification meetings to ensure that the participants are adequately trained and briefed on the interview protocol (Rubin & Rubin, 2012). After completing each interview, the primary author emailed a summary of the findings to each participant, following the approach recommended by Morse (1994) to ensure an accurate representation of their perspectives.

Additionally, an established coding scheme and a systematic data analysis plan were implemented to ensure objective and thorough data analysis. Validity perspectives proposed by Gibbert et al. (2008) and Runeson and Höst (2009) were applied into the coding scheme development and data analysis. Gibbert et al. (2008) offer a structured approach to address threats to construct validity, emphasizing the importance of ensuring that selected variables accurately represent underlying theoretical constructs. This was particularly crucial in the context of a case study. The coding scheme for each proposition was assessed to ensure that it aligns with the theoretical constructs underlying it. To validate relevance and completeness of the coding categories, the main author conducted a pilot project and sought feedback from the two co-authors who are domain experts. Meanwhile, Runeson and Höst (2009) contribute valuable insights into addressing concerns related to internal and external validity in empirical research, guiding the examination of potential biases, confounding variables, and the generalisability of findings beyond the specific case study context. Despite the qualitative nature of the study, quantitative analysis was incorporated, with results interpreted in consideration of internal validity threats. However, we acknowledge that the metrics proposed for the quality-in-use model have undergone limited empirical testing, primarily through initial pilot studies. First, they could be re-evaluated and refined using the Goal-Question-Metric (GQM) approach. This will involve clearly defining specific goals for each principle, formulating precise questions to reflect these goals, and ensuring our metrics directly answer these questions. While these studies provided valuable insights, they fall short of a comprehensive empirical validation needed to firmly establish the metrics reliability and validity across diverse IE contexts. More comprehensive empirical validation across a broader range of IE domains needs to be conducted. A larger and more diverse group of domain experts should be engaged in the review process. Their feedback could be obtained on the relevance, clarity, and comprehensiveness of each metric.

The choice of a multiple case study methodology aimed to explore the applicability of UCIEDP2 in two distinct settings and assess the similarity of findings. One case study was conducted in a real-world industrial context to minimise threats to external validity. However, external validity regarding the generalisability of study results to companies beyond the scope

of the research cannot be ignored. For the industrial case study, only one small and medium sized local company participated due to the novelty of IE technologies. Therefore, the inclusion of only one local company introduces a potential limitation, as our findings predominantly reflect its perspectives. In both case studies, the main researcher explained UCIEDP2, the proposed quality-in-use model, and ISO/IEC 25000 (2021) quality characteristics. Data recorded in impact estimation tables were collected through tests designed and written by developers/testers. These were validated by the consultant in the first case study and the chief developer in the second one. Multiple data collection techniques, such as interviews, observation, and document analysis, were utilised to bolster data reliability (Yin, 2018). Triangulation of data, involving the collection of information from various sources, including interviews with different stakeholders, was employed to cross-verify findings. UCIEDP2 was also compared against two methodologies namely prototyping and Agile Scrum through the two case studies respectively. Ethical considerations were addressed by conducting all study procedures in accordance with Middlesex University's Ethics Framework. Prior consent was obtained from individual participants, safeguarding their rights throughout the study. These measures collectively aimed to improve the overall reliability of the study.

8 Conclusion and future work

Engineering higher quality IEs is challenging. This study highlights the prevailing issues, including the absence of suitable methodologies. These challenges are further compounded by:

- Insufficient guidance on tracking and measuring quality throughout the development process.
- Lack of a dedicated quality model for assessing the quality of IEs.
- Separation of quality characteristics from functional requirements.
- Limited empirical research focusing on quality aspects for IEs.

To tackle these issues, our study introduces the UCIEDP2 methodology, designed to define, measure, and monitor quality aspects for IEs during their development. This approach involves specifying quality characteristics and measures, drawn from the ISO/IEC 25000 (2021) family of standards, for every functional requirement. Additionally, a novel quality-in-use model, adapted from the generic ISO/IEC 25010 (2021) model and aligned with the nine guiding principles of IEs, is presented for IE evaluation.

Through a multiple case study involving projects from different domains, our proposed methodologies were investigated, revealing that integrating quality characteristics from the ISO/IEC 25000 (2021) standards with functional requirements builds quality into systems. Stakeholder collaboration in defining targets using quality standards' measures and metrics enables developers to proactively address deviations from quality targets during development.

The effectiveness of the proposed quality-in-use model was evident in guiding system development by offering an objective perspective on expected capabilities. Involving stakeholders throughout the process ensures the delivery of systems that provide optimal value. However, challenges remain, such as the need to shorten development cycles for more frequent stakeholder feedback and managing the increased number of system tests. Current efforts focus on applying the methodology to a broader range of industrial projects, aiming to refine and adapt our approaches for wider applicability.

Appendix A – list of selected articles following SLR

Paper ID	Title	Contribution
P1	An analysis of quality of model driven development solutions applied to cyber-physical devices	This study evaluates the quality of Cyber Physical Systems using a model driven development platform. (Goncalves et al., 2022)
P2	Challenges for usability testing in ubiquitous systems	This paper discusses some challenges for doing usability testing in ubiquitous systems based on a literature analysis and the authors' own experience towards testing these types of systems. The authors also discuss ongoing research for how to design usability testing by considering context-awareness factors. (Bezerra et al., 2014)
P3	Data quality-oriented efficacy evaluation method for AAL	This paper presents a data quality model for assessing the efficacy of AAL technologies. The authors highlight that the system could be evaluated during different stages of its life cycle by measuring efficacy to determine its functional and non-functional performance (Beevi et al., 2016).
P4	Infrastructure for Ubiquitous Computing: Improving Quality with Modularisation	In this paper, the authors define a set of infrastructural concerns in the domain of ubiquitous computing and outline an aspect-oriented design to improve software modularity. They also discuss how improvement in quality can be quantifiably measured using the Goal-Question-Metric approach (Munnelly & Clarke, 2008).
P5	Metrics Evaluation for Context-Aware Computing	This study presents a software product quality metric for context-aware computing. The metrics have also been updated based on the technological, social, user and environment dimensions (Mantoro, 2009).
P6	Quality Provisioning in the Internet of Things Era: Current State and Future Directions	This paper proposes to extend quality of experience of IoT systems by including factors such as quality of data, quality of information and presents a quality aware framework (Pal et al., 2018).
P7	A study on the quality evaluation index system of smart home care for older adults in the community -based on Delphi and AHP	This paper presents a quality index system for smart home care for older adults (Chen et al., 2023).
P8	A tailored smart home for dementia care	This paper discusses development of a smart home solution for dementia care through the systems development process (Amiribesheli & Bouchachia, 2018).
P9	Evaluate and control service and transaction dependability of complex IoT systems	This study designed and evaluated an artifact to define and evaluate service and transaction dependability from a consumer-centric view in an action research project (Niedermaier et al., 2022).
P10	Development and Operation of Trustworthy Smart IoT Systems: The ENACT Framework	This paper presents a framework to provide DevOps support for smart IoT systems (Bruel et al., 2020).

Paper ID	Title	Contribution
P11	Applying architecture-based adaptation to automate the management of internet-of-things	The authors introduce an architecture-based adaptation approach to automate the management of IoT (Weyns et al., 2018).
P12	LNCS 7040 - Evaluation of AAL Platforms According to Architecture-Based Quality Attributes	This paper presents an evaluation of relevant AAL platforms based on a selection of quality attributes that are important for AAL systems (Antonino et al., 2011).
P13	Quality Model for CloudIot Applied in Ambient Assisted Living (AAL)	This paper presents a quality model with a particular focus on CloudIoT layers in AAL (Salgado Guerrero et al., 2021).
P14	Evaluating an IoT Application Using Software Measures	The aim of this paper is to investigate the applicability of software measures from ubiquitous to IoT systems (Carvalho et al., 2017).
P15	QoS-based formation of software architectures in the Internet of Things	This paper models and analyses QoS-related concerns in IoT architectures (Bures et al., 2019).
P16	An exploratory study on how internet of things developing companies handle user experience requirements	This qualitative study explores how companies elicit UX requirements in the context of IoT (Kamsties et al., 2018).
P17	A Quality in Use Model for Ambient Assisted Living (AAL) Systems	The main contribution of this paper is a quality in use model for AAL systems (Botto-Tobar et al., 2021).
P18	Heuristics to evaluate the usability of ubiquitous systems	This work proposes specific heuristics to evaluate the usability of ubiquitous systems (Streitz & Markopoulos, 2017).
P19	A Quality Model for Human-Computer Interaction Evaluation in Ubiquitous Systems	This paper presented a model for the HCI quality evaluation in ubiquitous systems (Santos et al., 2013).
P20	A Quality 4.0 Model for architecting industry 4.0 systems	This paper proposes a refined version of ISO/IEC 25010 quality model based on I4.0 needs (Antonino et al., 2022).
P21	Enabling correct design and formal analysis of Ambient Assisted Living systems	This paper proposes a verification approach and methodology to check fulfilment of non-functional requirements (Benghazi et al., 2012).
P22	Euphoria: A Scalable, event-driven architecture for designing interactions across heterogeneous devices in smart environments	This paper proposes a new software architecture design and implementation which facilitates prototyping, deployment and evaluation of interactions across devices in a smart environment (Schipor et al., 2019).
P23	A model-driven approach for quality of context in pervasive systems	This study focuses on quality of context information (Hoyos et al., 2016).
P24	A Quality Model for the Evaluation AAL Systems	This paper underlines the need to define a data quality model based on quality characteristics of AAL systems (Kara et al., 2017).
P25	Assessing Usability of Ubiquitous Systems Using Quality Model	In this study, the authors identified quality characteristics and measures for assessing usability of ubiquitous systems. A usability measurement model is also proposed (Hamzah et al., 2018).
P26	CloudIoTSecurity: Evaluating the Security in Cloud IoT Applications	This paper presents a quality model based on ISO/IEC 25010 model and a method aligned to ISO/IEC 25040 to evaluate the security of cloud IoT applications (Cedillo et al., 2020).

Paper ID	Title	Contribution
P27	Modelling Quality of IoT Experience in Autonomous Vehicles	This study proposes a new architecture which takes into consideration quality of data, quality of network and quality of context to determine overall quality of IoT in the domain of autonomous vehicles (Minovski et al., 2020).
P28	Quality in use measures for an AAL system for older adults	This paper proposes a set of quality measures to evaluate quality in use of an AAL platform that monitors and supports elderly people (Cristescu et al., 2020).
P29	Towards a New Interoperability Quality Model for IoTs	This paper presents a quality model to evaluate the interoperability of IoT platforms (Abdelouahid & Marzak, 2018).
P30	Trustworthiness and Quality of Context Information	This study proposes a quality of context model and introduces trustworthiness as measure of reliability of the context information provider (Neisse & Wegdam, 2008).
P31	A Real-time PPG Quality Assessment Approach for Healthcare Internet-of-Things	This paper proposes a novel PPG quality assessment approach for IoT-based health monitoring system with the goal of removing unreliable data (Naeini et al., 2019).
P32	Conceptualizing and measuring quality of experience of the internet of things: Exploring how quality is perceived by users	This study explores how quality is perceived by users and proposes a user experience model for IoT (Shin, 2017).
P33	CLOUDQUAL: A Quality Model for Cloud Services	This study is inspired from SERVQUAL and the e-service quality model and initiate a quality model named CLOUDQUAL for cloud services (Xianrong et al., 2014).

Appendix B – quality characteristics per domain

IE Domain	Quality Characteristic	Paper ID
AAL	Accuracy, completeness, timeliness and interpretability	P3
	Reliability, security, maintainability, efficiency, safety	P12
	Security, usability, reliability	P13
	Effectiveness, efficiency, satisfaction, freedom from risk, context coverage	P17
	Safety, timeliness	P21
	Accuracy, completeness, recoverability, confidentiality, efficiency, precision, reliability, security	P24
	Effectiveness, trust, usefulness, freedom from risk	P28
Ubiquitous systems	Context-awareness, transparency, attention, mobility, calmness	P2
	Comprehensibility, manageability, maintainability, scalability, testability, reusability, usability	P4
	Usability	P18
	Trust, resource-limitedness, usability, ubiquity	P19
	Context awareness, calmness, transparency, attention, mobility	P25

IE Domain	Quality Characteristic	Paper ID
Context-aware systems	Functionality, reliability, usability, efficiency, maintainability, portability	P5
	Trustworthiness of context providers	P30
Cyber-physical systems	Analysability, changeability, stability, testability	P1
IoT Systems	Quality of Data, Quality of Information	P6
	Effectiveness, efficiency, satisfaction	P9
	Trustworthiness	P10
	Quality goals are set	P11
	Calmness	P14
	Power consumption, sensing accuracy, execution time	P15
	User Experience	P16
	Confidentiality, integrity, non-repudiation, accountability, authenticity, availability	P26
	Quality of experience	P27
	Confidentiality, efficiency, mobility, flexibility, distributivity, functionality	P29
	Accuracy of health parameters	P31
	Satisfaction, involvement, affordance, coolness, enjoyment, hedonicity, content quality, reliability of services, system	P32
	Usability, availability, reliability, responsiveness, security, elasticity	P33
Industry 4.0	Reliability, maintainability (modifiability), maintainability (testability)	P20
Pervasive Systems	Trustworthiness, comparability, completeness, relevance, understandability	P23
Smart environment	Timeliness, reliability, ease of use, tangibility, empathy	P7
	Functionality, usability	P8
	Adaptability, modularity, flexibility, interoperability	P22

Appendix C – example of an impact estimation table

Functional Requirement	Quality Requirement	Design Idea	
		D1	D2
FR001	Reliability 300 <-> 3000 Hours (Mean Time Before Failure)	1950 h (1650 h) ±0 61% ± 0	1140 h (840 h) ±240 31% ± 9%
	Usability 20 <-> 10 min	19 min (1 min) ±4 min 10% ± 40%	14 min (6 min) ±9 min 60% ± 90%
	Maintenance 1.1 M <-> 100 K US dollars/year	1.1 M $/Y (0 K$/Y) ±180 K 0% ± 18%	100 K $/Y (1 M$/Y) ±720 K 100% ± 72%
Sum of Quality Requirements		71%	191%

Resources			
	Capital 0 <-> 1 M US dollars	500 K (500 K) ±200 K 50% ± 20	100 K (100 K) ±200 K 10% ± 20
Sum of Costs		50%	10%
Quality to Cost Ratio		1.42 (71/50)	19.10 (191/10)

The above Impact Estimation Table (IET) is an example of how it could be used to estimate the impact of implementing three quality requirements (reliability, usability, and maintenance) for a functional requirement denoted by FR0001 (Gilb, 2005). The estimates are based on experience data and the estimated project duration is one year. Baseline and target values are represented using (Baseline <-> Target) pair notation. In this example, we are comparing two design ideas. The first idea is expected to improve usability by 1 min whereas the second idea is projected to improve usability by 6 min. To account for potential errors or uncertainties, we estimate ranges of ± 4 and ± 9 min respectively. The percentage impact serves as a measure of the proportional change from an initial baseline to a specified target. For instance, percentage impact for the second idea is determined as follows:

$$(1\text{M}/(1.1\text{M} - 100\text{K})) \times 100 = 100\%$$

100% indicates that the target has been fully achieved, while 0% signifies no change at all. There is also a possibility of defining an overall percentage uncertainty. The total percentage impact of quality requirements serves as the determinant of how "good" a design idea is, while the aggregate of costs provides an estimation of its financial implications. The quality-to-cost ratio measures the cost-effectiveness of a given idea, and according to this ratio, the second design idea emerges as a better choice compared to the first idea. Despite the existence of more sophisticated versions of the IET, this simplified iteration captures essential data such as quality and cost. Moreover, it can be employed throughout the development lifecycle for the assessment of design ideas or the comparison of multiple versions of an application in terms of cost-effectiveness.

Appendix D – first interview questions

Introduction

1. Explanation about the research study, what we are looking for, how they will benefit from the results and that they will be anonymous.
2. Details of the organization

    (a) Mission statement
    (b) Vision

3. Subject's personal history in the company

    (a) Role
    (b) Experience

(c) Daily tasks
(d) Responsibility

General information about projects at the organisation

4. What types of projects are developed at the organisation?
5. Provide details about the following:

(a) What are the timescales for the project?
(b) How many people are involved and their roles?
(c) Which development methodologies are adopted?
(d) What are innovative about the projects?

Questions about quality management

6. What quality management processes are used within a project?

(a) Do you follow any specific standard for managing quality at the project level?
(b) Are there any best practices which you are encouraged to follow with regards to quality?
(c) Which units and/or roles are involved in quality management?

7. How is quality of a system assessed?
8. How are functional and quality requirements managed?
9. What quality requirements (top 5) are most important in your work?
   (a) Efficiency/performance/capacity, reliability, safety, security, maintainability, usability, others
10. During which phase(s) are the quality requirements evaluated?

(a) By whom?
(b) How (measurement, reviews, testing etc.)?

11. What is the course of action if results are not as expected?
12. How has the quality of systems evolved using the current approach?
13. What are the pros and cons working with the current approach?

Ending

14. Other observations? Anything we may have not covered?
15. Any relevant observations with regards to your specific perspective in the project.

Appendix E – second interview questions

Introduction

1. What is the aim of the project?

2. What is the vision of the stakeholders?
3. What are the timescales of the project?
4. What is the budget allocated for the project?
5. How many people were involved in the project and what were their roles?
6. Which quality characteristics and attributes were selected from ISO/IEC 25010?
7. Who were involved in defining target values for the functional requirements, including the core set of requirements based on IE principles, of the system?
8. Were quality requirements specified in the system specifications document previously?
9. What previous benchmark data was available?

Questions about UCIEDP2 methodology

10. Describe three most significant aspects of UCIEDP2:

    (a) What worked well?
    (b) What did not work as well?

11. State any adaptations to your current setup specifically due to UCIEDP2?
12. How was the project (time, cost, resources) affected by UCIEDP2?
13. Your overall experience with UCIEDP2 compared to your previous way of working.

    (a) On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate your previous method?
        (i) Usefulness
        (ii) Power of clarification
        (iii) Ease of use
        (iv) Effectiveness
        (v) Time efficiency
    (b) On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate UCIEDP2?
        (i) Usefulness
        (ii) Power of clarification
        (iii) Ease of use
        (iv) Effectiveness
        (v) Time efficiency

14. Was the team able to measure and monitor accomplishment of target values of quality attributes for each functional requirement during analysis, design, implementation, evaluation?
15. How did you incorporate stakeholders' feedback during development process?
16. Will you use UCIEDP2 in its current form in other projects?
17. Do you have any recommendations what to improve?

Questions about proposed quality-in-use model to evaluate the system

18. How did you evaluate quality of your system previously?
19. How did the quality-in-use model work? Describe the three most significant aspects:

(a) What worked well?
(b) What did not work as well?

20. State any adaptations required specifically due to this new quality model?
21. How was the project (time, cost, resources) affected by this new quality model?
22. Your overall experience with this quality model compared to your previous way of working.

(iii) On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate your previous method?
(i) Usefulness
(ii) Power of clarification
(iii) Ease of use
(iv) Effectiveness
(v) Time efficiency

(iv) On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate the quality-in-use model?
(i) Usefulness
(ii) Power of clarification
(iii) Ease of use
(iv) Effectiveness
(v) Time efficiency

23. How often did your team use the quality-in-use model to evaluate the system during development and after development?
24. Who were involved in the evaluation process?
25. Did you note any improvement in quality of the system? What data do you have to prove this?

Ending

26. Other observations? Anything we may have not covered?
27. Any relevant observations with regards to your specific perspective in the project.

Author contribution This study was carried out as part of a PhD study pursued by Mr. Adityarajsingh Santokhee (corresponding author) at Middlesex University UK. Prof. Dr. Juan Carlos Augusto and Dr Lindsey Brodie reviewed and approved the final manuscript.

Funding This project is not funded at all.

Data availability The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare no competing interests.

References

Aarts, E., & Roovers, R. (2003). IC design challenges for ambient intelligence. *2003 Design, Automation and Test in Europe Conference and Exhibition* (pp. 2–7). Munich, Germany. https://doi.org/10.1109/DATE.2003.1253578

Abdelouahid, R. A., & Marzak, A. (2018). *Towards a New Interoperability Quality Model for IoTs* (p. 1). Institute of Electrical and Electronics Engineers.

Ahmad, M., Belloir, N., & Bruel, J. M. (2015). Modeling and verification of functional and non-functional requirements of ambient self-adaptive systems. *Journal of Systems and Software, 107*, 50–70. https://doi.org/10.1016/j.jss.2015.05.028

Alasuutari, P., Bickman, L., & Brennan, J. (2008). *The Sage handbook of social research methods*. Sage.

Ali, A., Khalil, I., Ahmad, I., Parveen, I., & Uz Zaman, U. K. (2022). Role of Non-functional Requirements in projects' success. In *2022 2nd International Conference on Digital Futures and Transformative Technologies (ICoDT2)*, pp. 1–7. https://doi.org/10.1109/ICoDT255437.2022.9787463

Amiribesheli, M., & Bouchachia, H. (2018). A tailored smart home for dementia care. *Journal of Ambient Intelligence and Humanized Computing,9*(6), 1755–1782. https://doi.org/10.1007/s12652-017-0645-7

Antonino, P. O., Capilla, R., Pelliccione, P., Schnicke, F., Espen, D., Kuhn, T., & Schmid, K. (2022). Quality 4.0 Model for architecting industry 4.0 systems. *Advanced Engineering Informatics,54*, 101801. https://doi.org/10.1016/j.aei.2022.101801

Antonino, P. O., Schneider, D., Hofmann, C., & Nakagawa, E. Y., et al. (2011). Evaluation of AAL Platforms According to Architecture-Based Quality Attributes. In D. V. Keyson (Ed.), *Ambient Intelligence. AmI 2011. Lecture Notes in Computer Science* (vol 7040). Heidelberg: Springer, Berlin.

Anurag, A., & Kamatchi, R. (2019). (2019) Designing a "software quality model" based on RCCA of defects and validating based on "quality algorithm." *J Softw Evol Proc.,31*, e2210. https://doi.org/10.1002/smr.2210

Ashouri, M., Davidsson, P., & Spalazzese, R. (2021). Quality attributes in edge computing for the Internet of Things: A systematic mapping study. *Internet of Things,13*, 100346. https://doi.org/10.1016/j.iot.2020.100346. ISSN 2542-6605.

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks (Amsterdam, Netherlands: 1999),54*(15), 2787–2805. https://doi.org/10.1016/j.comnet.2010.05.010

Augusto, J. C. (2007). Ambient Intelligence: The Confluence of Ubiquitous/Pervasive Computing and Artificial Intelligence. In A. J. Schuster (Ed.), *Intelligent Computing Everywhere*. London: Springer. https://doi.org/10.1007/978-1-84628-943-9_11

Augusto, J. C. (2014). User-centric software development process. In *2014 international conference on intelligent environments* (pp. 252–255). Shanghai, China. https://doi.org/10.1109/IE.2014.47

Augusto, J., Callaghan, V., Kameas, A., Cook, D., & Satoh, I. (2013). *Intelligent Environments: a manifesto Human - centric. Computing and Information Sciences* (3rd ed., p. 12). Springer.

Augusto, J., Kramer, D., Alegre, U., Covaci, A., & Santokhee, A. (2018). The user-centred intelligent environments development process as a guide to co-create smart technology for people with special needs. *Universal Access in the Information Society, 17*, 115–130. https://doi.org/10.1007/s10209-016-0514-8

Banijamali, A., Pakanen, O. P., Kuvaja, P., & Oivo, M. (2020). Software architectures of the convergence of cloud computing and the Internet of Things: A systematic literature review. *Information and Software Technology,122*, 106271. https://doi.org/10.1016/j.infsof.2020.106271. ISSN 0950-5849.

Bass, L., Clements, P., & Kazman, R. (2012). *Software Architecture in Practice* (3rd ed.). Addison-Wesley.

Beevi, F. H. A., Wagner, S. R., Pedersen, C. F., & Hallerstede, S. (2016). *Data Quality Oriented Efficacy Evaluation Method for Ambient Assisted Living Technologies*. Association for Computing Machinery.

Benghazi, K., Hurtado, M. V., Hornos, M. J., Rodríguez, M. L., Rodríguez-Domínguez, C., Pelegrina, A. B., & Rodríguez-Fórtiz, M. J. (2012). 'Enabling correct design and formal analysis of Ambient Assisted

Living systems. *The Journal of Systems and Software,85*(3), 498–510. https://doi.org/10.1016/j.jss.2011.05.022

Bezerra, C., Andrade, R., Santos, R., Abed, M., de Oliveira, K., Monteiro, J., Santos, I., & Ezzedine, H. (2014). *Challenges for usability testing in ubiquitous systems* (p. 183). ACM.

Botto-Tobar, M., Montes León, S., Camacho, O., Chávez, D., Torres-Carrión, P., & Zambrano Vizuete, M. (2021). *' Quality in Use Model for Ambient Assisted Living (AAL) Systems Applied Technologies* (pp. 643–660). Switzerland: Springer International Publishing AG.

Boehm, B. (1981). *Software Engineering Economics*. Prentice-Hall.

Budgen, D. (2003). *Software Design* (2nd ed.). Pearson Education.

Brodie, L., & Woodman, M. (2009). Using metrics to express quality. In *17th International Conference on Software Quality Management (SQM 2009)*. Southampton.

Bruel, J., Mazzara, M. and Meyer, B. (2020) 'Development and Operation of Trustworthy Smart IoT Systems: The ENACT Framework. In *Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment.* Switzerland: Springer International Publishing AG, pp. 121–138.

Bures, T., Duchien, L., & Inverardi, P. (2019). QoS-Based Formation of Software Architectures in the Internet of Things. In *Software Architecture*. Switzerland: Springer International Publishing AG, pp. 178–194.

Carvalho, R. M., Andrade, R. M. C., Barbosa, J., Maia, A. M., Junior, B. A., Aguilar, P. A., Bezerra, C. I. M., & Oliveira, K. M. (2017). *Distributed, Ambient and Pervasive Interactions : 5th International Conference, DAPI 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9–14, 2017, Proceedings*. Springer International Publishing.

Cedillo, P., Bermeo, A., Piedra-Garcia, D., & Tenezaca-Sari, P. (2020). *CloudIoTSecurity: Evaluating the Security in Cloud IoT Applications* (p. 1). IEEE.

Chen, H., Zhang, Y., & Wang, L. (2023). ' study on the quality evaluation index system of smart home care for older adults in the community - based on Delphi and AHP. *BMC Public Health,23*(1), 411. https://doi.org/10.1186/s12889-023-15262-1

Chung, L., & do Prado Leite, J.C.S. (2009). On Non-Functional Requirements in Software Engineering. In A. T. Borgida, V. K. Chaudhri, P. Giorgini, & E. S. Yu (Eds.), *Conceptual Modeling: Foundations and Applications. Lecture Notes in Computer Science* (vol. 5600). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-642-02463-4_19

Clickup. (2022). Available at: https://clickup.com/. Accessed 1 Jan 2022.

Cote, M. A., Suryn, W., & Georgiadou, E. (2006). Software quality model requirements for software quality engineering. *14th International Conference on Software Quality Management,* 31–50.

Creswell, J. W., & Creswell, J. (2017). *Research design: Qualitative, quantitative, and mixed methods approaches* (4th ed.). Newbury Park: Sage.

Cristescu, I., Balog, A., & Bajenaru, L. (2020). *Quality in use measures for an AAL system for older adults* (p. 1). IEEE.

Dalcher, D., & Brodie, L. (2007). *Successful IT Projects*. London (UK): Thomson Learning.

Dyba, T., Dingsoyr, T., & Hanssen, G. (2007). Applying systematic reviews to diverse study types: An experience report. In *First international symposium on empirical software engineering and measurement (ESEM 2007)* (pp. 225–234.s).

Erazo-Garzon, L., Illescas-Peña, L., & Cedillo, P. (2021). A Quality in Use Model for Ambient Assisted Living (AAL) Systems. In M. Botto-Tobar, S. Montes León, O. Camacho, D. Chávez, P. Torres-Carrión, & M. Zambrano Vizuete (Eds.), Applied Technologies. ICAT 2020. *Communications in Computer and Information Science*, vol 1388. Cham: Springer. https://doi.org/10.1007/978-3-030-71503-8_50

European Commission. (2009). Ambient Assisted Living - Preparation of an Art. 169-Initiative. https://cordis.europa.eu/project/rcn/71922/factsheet/en. Accessed 19 May 2020.

Fenton, N., & Bieman, J. (2014). Software Metrics: A Rigorous and Practical Approach. *Third Edition*. https://doi.org/10.1201/b17461

Fizza, K., Banerjee, A., Jayaraman, P. P., Auluck, N., Ranjan, R., Mitra, K., & Georgakopoulos, D. (2023). A Survey on Evaluating the Quality of Autonomic Internet of Things Applications. *IEEE Communications Surveys and Tutorials,25*(1), 567–590. https://doi.org/10.1109/COMST.2022.3205377

Garcés, L., Ampatzoglou, A., Avgeriou, P., & Nakagawa, E. (2017). Quality attributes and quality models for ambient assisted living software systems: A systematic mapping. *Information and Software Technology,82*, 121–138. https://doi.org/10.1016/j.infsof.2016.10.005. ISSN 0950-5849.

Ghobadian, A., & Speller, S. (1994). Gurus of quality: A framework for comparison. *Total Quality Management,5*(3), 53–70. https://doi.org/10.1080/09544129400000025

Gibbert, M., Ruigrok, W., & Wicki, B. (2008). What passes as a rigorous case study? *Strategic Management Journal,29*(13), 1465–1474.
Gilb, T. (2005). Competitive Engineering. In L. Brodie (Ed.), *A Handbook for Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage.* Buterworth-Heinemann. ISBN 0750665076.
Gilb, T., & Brodie, L. (2012). What's fundamentally wrong? Improving our approach towards capturing value in requirements specification. In *Proceedings of the 22nd Annual INCOSE International Symposium* (IS 2012). Rome, Italy. California: International Council on Systems Engineering (INCOSE), 1, p. 1010.
Gordon, V. S., & Bieman, J. M. (1993). Reported effects of rapid prototyping on industrial software quality. *Software Quality Journal,2*, 93–108. https://doi.org/10.1007/BF00590438
Goncalves, R. F., Menolli, A., & Dionisio, G. M. (2022). *An Analysis of the Quality of Model Driven Development Solutions Applied to Cyber-Physical Devices*. ACM.
Greening, D.R. (2015). Agile Enterprise Metrics. In *2015 48th Hawaii International Conference on System Sciences*, pp. 5038–5044. https://doi.org/10.1109/HICSS.2015.597
Hamzah, N., Mageswaran, G., Nagappan, S. D., & Chuprat, S. (2018). *Assessing Usability of Ubiquitous Systems Using Quality Model* (p. 1). IEEE.
Hanslo, R., Mnkandla, E., & Vahed, A. (2019) Factors that contribute significantly to Scrum adoption. In: *FedCSIS 2019 Proceedings*. IEEE, pp. 821–829. https://doi.org/10.15439/2019F220
Hoyos, J. R., García-Molina, J., Botía, J. A., & Preuveneers, D. (2016). A model-driven approach for quality of context in pervasive systems. *Computers & Electrical Engineering,55*, 39–58. https://doi.org/10.1016/j.compeleceng.2016.07.002
Hron, M., & Obwegeser, N. (2022). Why and how is Scrum being adapted in practice: A systematic review. *Journal of Systems and Software,183*, 111110. https://doi.org/10.1016/j.jss.2021.111110. ISSN 0164–1212.
Humble, J., & Farley, D. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Pearson Education.
ISO/IEC 25010. (2021). *ISO/IEC 25010:2011: Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (Square).* System and Software Quality Models. Geneva: ISO 2021.
ISO/IEC 25012. (2021). *ISO/IEC 25012:2008 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Data quality model*. System and Software Quality Models. Geneva: ISO 2021.
ISO/IEC 25000. (2021). *ISO/IEC 25000:2014 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*. System and Software Quality Models. Geneva: ISO 2021.
ISO/IEC 25023. (2022) *ISO/IEC 25023:2016 Quality measurement framework*. System and Software Quality Models. Geneva: ISO 2022.
IPA. (2010). *Embedded System development Quality Reference guide, Software Engineering Center, Technology Headquarters*. Japan: Information-technology Promotion Agency.
Jones, C., & Bonsignour, O. (2011). *The Economics of Software Quality* (1st ed.). Addison-Wesley Professional.
Kamsties, E., Horkoff, J. and Dalpiaz, F. (2018) An Exploratory Study on How Internet of Things Developing Companies Handle User Experience Requirements. In: *24th International Working Conference on Requirements Engineering Foundation for Software Quality, REFSQ 2018,Utrecht, Netherlands.* Switzerland: Springer International Publishing AG, pp. 20–36.
Kan, S. H. (2002). *Metrics and Models in Software Quality Engineering* (2nd ed.). Addison-Wesley.
Kara, M., Lamouchi, O., & Ramdane-Cherif, A. (2017). 'A Quality Model for the Evaluation AAL Systems. *Procedia Computer Science,113*, 392–399. https://doi.org/10.1016/j.procs.2017.08.354
Kakarontzas, G., Anthopoulos, L., Chatzakou, D., & Vakali, A. (2014). A conceptual enterprise architecture framework for smart cities: A survey-based approach. In *2014 11th International Conference on e-Business (ICE-B)*, pp. 47–54.
Kitchenham, B., & Charters, S. (2007). Guidelines for Performing Systematic Literature Reviews in Software Engineering. In: *Technical Report EBSE 2007–001*. Keele University and Durham University Joint Report.
Kurtel, K., & Ozemre, M. (2013). Cohesive software measurement planning framework using ISO standards: A case study from logistics service sector. *Journal of Software: Evolution and Process, 25*, 663–679. https://doi.org/10.1002/smr.1557
Maciel, P., Dantas, J., Melo, C., Pereira, P., Oliveira, F., Araujo, J., & Matos, R. (2022). A survey on reliability and availability modeling of edge, fog, and cloud computing. *Journal of Reliable Intelligent Environments, 8*, 227–245. https://doi.org/10.1007/s40860-021-00154-1
Mantoro, T. (2009). *Metrics evaluation for context-aware computing* (p. 574). USA: ACM.

McKinsey. (2021). The Internet of things: Catching up to an accelerating opportunity. McKinsey & Company. November 2021. Available from: https://www.mckinsey.com/~/media/mckinsey/business%20functions/mckinsey%20digital/our%20insights/iot%20value%20set%20to%20accelerate%20through%202030%20where%20and%20how%20to%20capture%20it/the-internet-of-things-catching-up-to-an-accelerating-opportunity-final.pdf. Accessed 1 May 2023.

Martirano, L., & Mitolo, M. (2020). Building Automation and Control Systems (BACS): a Review. In: *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe)*. Madrid, Spain, pp. 1–8. https://doi.org/10.1109/EEEIC/ICPSEurope49358.2020.9160662

Memon, M., Wagner, S. R., Pedersen, C. F., Beevi, F. H., & Hansen, F. O. (2014). Ambient assisted living healthcare frameworks, platforms, standards, and quality attributes. *Sensors (Basel),14*(3), 4312–41. https://doi.org/10.3390/s140304312. PMID: 24599192; PMCID: PMC4003945.

Minovski, D., Ahlund, C., & Mitra, K. (2020). Modeling Quality of IoT Experience in Autonomous Vehicles. *IEEE Internet of Things Journal,7*(5), 3833–3849. https://doi.org/10.1109/JIOT.2020.2975418

Mohammadi, R., & Javidan, R. (2022). EFSUTE: A novel efficient and survivable traffic engineering for software defined networks. *Journal of Reliable Intelligent Environments, 8*, 247–260. https://doi.org/10.1007/s40860-021-00139-0

Morse, J. M. (1994). Designing funded qualitative research. In N. K. Denzin & Y. S. Lincoln (Eds.), *Handbook of qualitative research* (pp. 220–235). Sage Publications Inc.

Munnelly, J., & Clarke, S. (2008). *Infrastructure for ubiquitous computing* (p. 1). ACM.

Murzi, M. (2007). The philosophy of logical positivism [Online]. Available: http://www.murzim.net/LP/LP.pdf. Accessed 31 Jan 2023.

Naeini, E. K., Azimi, I., Rahmani, A. M., Liljeberg, P., & Dutt, N. (2019). A Real-time PPG Quality Assessment Approach for Healthcare Internet-of-Things. *Procedia Computer Science,151*, 551–558. https://doi.org/10.1016/j.procs.2019.04.074

Neisse, R., & Wegdam, M. (2008). *Trustworthiness and Quality of Context Information* (p. 1925). IEEE.

Niedermaier, S., Zelenik, T., Heisse, S., & Wagner, S. (2022). Evaluate and control service and transaction dependability of complex IoT systems. *Software Quality Journal,30*(2), 337–366. https://doi.org/10.1007/s11219-021-09556-z

Nielsen, J. (1994). Heuristic evaluation. In J. Nielsen & R. L. Mack (Eds.), *Usability Inspection Methods.* New York City, NY: Wiley & Sons.

Nuñez-Varela, A. S., Perez-Gonzalez, H. G., Martínez-Perez, F. E., & Soubervielle-Montalvo, C. (2017). Source code metrics: A systematic mapping study. *Journal of Systems and Software,128*(2017), 164–197.

Ogbuabor, G., Augusto, J., Moseley, R., & Van Wyk, A. (2021). Context-aware support for cardiac health monitoring using federated machine learning. In M. Bramer, & R. Ellis (Eds.), *41st SGAI International Conference on Artificial Intelligence (AI-2021)* (pp. 267–281). Cambridge, England: Springer. https://doi.org/10.1007/978-3-030-91100-3_22

Olianas, D., Leotta, M., & Ricca, F. (2022). MATTER: A tool for generating end-to-end IoT test scripts. *Software Quality Journal, 30*, 389–423. https://doi.org/10.1007/s11219-021-09565-y

Oram, A., & Wilson, G. (2010). *Making Software: What Really Works, and Why We Believe It*. Sebastopol, CA: O'Reilly Media.

Oriol, M., Martínez-Fernández, S., Behutiye, W., et al. (2020). Data-driven and tool-supported elicitation of quality requirements in agile companies. *Software Quality Journal,28*, 931–963. https://doi.org/10.1007/s11219-020-09509-y

Pal, D., Vanijja, V., & Varadarajan, V. (2018). *Quality Provisioning in the Internet of Things Era* (p. 1). USA: ACM.

Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach* (8th ed.). McGraw-Hill Education.

Rashidi, P., Cook, D. J., Holder, L. B., & Schmitter-Edgecombe, M. (2011). 'Discovering Activities to Recognize and Track in a Smart Environment. *IEEE Transactions on Knowledge and Data Engineering,23*(4), 527–539. https://doi.org/10.1109/TKDE.2010.148

Regan, G., McCaffery, F., Paul, P. C., Reich, J., Armengaud, E., Kaypmaz, C., Zeller, M., Guo, J. Z., Longo, S., O'Carroll, E., & Sorokos, I. (2020). Quality improvement mechanism for cyber physical systems—An evaluation. *Journal of Software: Evolution and Process, 32*, e2295. https://doi.org/10.1002/smr.2295

Reggio, G., Leotta, M., Cerioli, M., Spalazzese, R., & Alkhabbas, F. (2020). What are IoT systems for real? An experts' survey on software engineering aspects. *Internet of Things,12*, 100313. https://doi.org/10.1016/j.iot.2020.100313

Rizk, J., & Hillier, C. (2022). Digital technology and increasing engagement among students with disabilities: Interaction rituals and digital capital. *Computers and Education Open,3*, 100099. https://doi.org/10.1016/j.caeo.2022.100099

Rodríguez-Domínguez, C., Santokhee, A., & Hornos, M. J. (2022). Intelligent environments with entangled quality properties. *Journal of Reliable Intelligent Environments, 8*, 223–226. https://doi.org/10.1007/s40860-022-00182-5

Rubin, H. J., & Rubin, I. S. (2012). *Qualitative Interviewing: The Art of Hearing Data* (3rd ed.). Los Angeles, CA: Sage. ISBN: 978-1-4129-7837-8.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empir Software Eng,14*, 131. https://doi.org/10.1007/s10664-008-9102-8

Ruiz-Lopez, T., Rodriguez-Dominguez, C., Noguera, M., Rodriguez, M., Benghazi, K., & Garrido, J. (2013). Applying model-driven engineering to a method for systematic treatment of NFRs in Aml systems. *J. Ambient Intell. Smart Environ.,5*, 287–310.

Salgado Guerrero, J. P., Chicaiza Espinosa, J., Cerrada Lozada, M., & Berrezueta-Guzman, S. (2021) Quality Model for CloudIot Applied in Ambient Assisted Living (AAL). In: *Information and Communication Technologies.* Switzerland: Springer International Publishing AG, pp. 184–198.

Salomón, S., Duque, R., Montaña, J. L., & Tenés, L. (2023). Towards automatic evaluation of the Quality-in-Use in context-aware software systems. *Journal of Ambient Intelligence and Humanized Computing, 14*, 10321–10346. https://doi.org/10.1007/s12652-021-03693-w

Salvi, D., Montalva Colomer, J. B., Arredondo, M. T., Prazak-Aram, B., & Mayer, C. (2015). *A Framework for Evaluating Ambient Assisted Living Technologies and the Experience of the universAAL Project.* 329–352.

Santokhee, A., Augusto, J. C., & Brodie, L. (2019). Towards a general framework for evaluating intelligent environments methodologies. In *Intelligent Environments (Workshops)* (pp. 17–25).

Santos, R. M., De Oliveira, K. M., Andrade, R. M. C., Santos, I. S., & Lima, E. R. (2013). *A Quality Model for Human-Computer Interaction Evaluation in Ubiquitous Systems*. Springer International Publishing.

Saunders, M., Lewis, P., & Thornhill, A. (2016). *Research Methods for Business Students* (7th ed.). Pearson, Harlow.

Sas, D., & Avgeriou, P. (2020). Quality attribute trade-offs in the embedded systems industry: An exploratory case study. *Software Quality Journal,28*, 505–534. https://doi.org/10.1007/s11219-019-09478-x

Schipor, O., Vatavu, R., & Vanderdonckt, J. (2019). Euphoria: A Scalable, event-driven architecture for designing interactions across heterogeneous devices in smart environments. *Information and Software Technology,109*, 43–59. https://doi.org/10.1016/j.infsof.2019.01.006

Scott, E., Milani, F., Kilu, E., & Pfahl, D. (2021). Enhancing agile software development in the banking sector—A comprehensive case study at LHV. *Journal of Software: Evolution and Process, 33*, e2363. https://doi.org/10.1002/smr.2363

SCRUMstudy™. (2016). *A Guide to the Scrum Body Of Knowledge (SBOK™ Guide)* – (2016th ed.). Arizona, USA.

Sharma, M., Assotally, A., & Bekaroo, G. (2022) RaspiMonitor: A Raspberry Pi Based Smart Home Monitoring System. In: *2022 3rd International Conference on Next Generation Computing Applications (NextComp)*, pp. 1–6. https://doi.org/10.1109/NextComp55567.2022.9932198

Shin, D. (2017). Conceptualizing and measuring quality of experience of the internet of things: Exploring how quality is perceived by users. *Information & Management,54*(8), 998–1011. https://doi.org/10.1016/j.im.2017.02.006

Sicari, S., Rizzardi, A., & Coen-Porisini, A. (2019). How to evaluate an internet of things system: Models, case studies, and real developments. *Software: Practice and Experience, 49*, 1663–1685.

Sommerville, I. (2011). *Software Engineering* (9th ed.). Addison-Wesley.

Staron, M., Meding, W., Karlsson, G., & Nilsson, C. (2011). Developing measurement systems: An industrial case study. *Journal of Software Maintenance and Evolution: Research and Practice, 23*, 89–107.

Streitz, N., & Markopoulos, P. (2017). *'Heuristics to Evaluate the Usability of Ubiquitous Systems'Distributed* (pp. 120–141). Springer International Publishing AG.

Tröls, M., Mashkoor, A., Demuth, A., & Egyed, A. (2021). Ensuring safe and consistent coengineering of cyber-physical production systems: A case study. *Journal of Software: Evolution and Process, 33*, e2308. https://doi.org/10.1002/smr.2308

Vogel, M., Knapik, P., Cohrs, M., Szyperrek, B., Pueschel, W., Etzel, H., Fiebig, D., Rausch, A., & Kuhrmann, M. (2021). Metrics in automotive software development: A systematic literature review. *Journal of Software: Evolution and Process, 33*, e2296. https://doi.org/10.1002/smr.2296

Washizaki, H., Ogata, S., Hazeyama, A., Okubo, T., Fernandez, E. B., & Yoshioka, N. (2020). (2020) Landscape of Architecture and Design Patterns for IoT Systems. *IEEE Internet of Things Journal,7*(10), 10091–10101. https://doi.org/10.1109/JIOT.2020.3003528

Weiser, M. (1991). The Computer for the 21st Century. *Scientific American,265*, 94–104. https://doi.org/10.1038/scientificamerican0991-94

Werner, C. (2022). Towards a theory of shared understanding of non-functional requirements in continuous software engineering. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings (ICSE '22)*. Association for Computing Machinery. New York, NY, USA, 300–304. https://doi.org/10.1145/3510454.3517069

Weyns, D., Iftikhar, M. U., Hughes, D., & Matthys, N. (2018). *Applying Architecture-Based Adaptation to Automate the Management of Internet-of-Things*. Springer International Publishing.

World Health Organization. (2022). Global report on health equity for persons with disabilities. World Health Organization. https://apps.who.int/iris/handle/10665/364834. License: CC BY-NC-SA 3.0 IGO.

Wohlin, C. (2014). *Guidelines for snowballing in systematic literature studies and a replication in software engineering* (p. 1). ACM.

White, G., Nallur, V., & Clarke, S. (2017). Quality of service approaches in IoT: A systematic mapping, *Journal of Systems and Software, Volume 132, 2017. ISSN,186–203*, 0164–1212. https://doi.org/10.1016/j.jss.2017.05.125

Xianrong, Z., Martin, P., Brohman, K., & Da Li, Xu. (2014). 'CLOUDQUAL: A Quality Model for Cloud Services. *IEEE Transactions on Industrial Informatics,10*(2), 1527–1536. https://doi.org/10.1109/TII.2014.2306329

Yin, R. K. (2018). *Case Study Research and Applications: Design and Methods* (6th ed.). Sage.

Adityarajsingh Santokhee is a Senior Lecturer in Computer Science at Middlesex University Mauritius and a fellow of the UK Higher Education Academy. He completed his Ph.D. in Computer Science in April 2024 at Middlesex University, UK, under the supervision of Prof. Juan Carlos Augusto and Dr Lindsey Brodie. Aditya holds a BEng (Hons) in Computer Science and Engineering from the University of Mauritius, an MSc in Network Centred Computing from the University of Reading, and a PGCE in Higher Education from Middlesex University. With extensive experience in software development, teaching, and research in both commercial and academic settings since 2001, Aditya's research interests focus on Software Engineering methods, Data Science, and Intelligent Environments. He is actively involved in knowledge transfer, offering consultancy services based on his expertise and research findings. Additionally, he serves as the co-chair of the annual International Workshop on the Reliability of Intelligent Environments.

Prof. Juan Carlos Augusto is a researcher with expertise in Context-Aware Computing, Ambient Intelligence, Ambient Assisted Living, Intelligent Environments, Smart Environments, Pervasive Computing, Ubiquitous Computing, Person-Centric Computing, Internet of Things, sensor-based computing, Smart Classrooms, Smart Campus, and Smart Cities. Throughout his career, he has advocated for a shift from gadget-centered environment automation to a more user-centered approach. Prof. Augusto has contributed to over 280 peer-reviewed publications, including numerous articles in scientific journals. He has also edited landmark books such as the Handbook on Ambient Intelligence and Smart Environments (Springer, 2009), the Handbook on Ambient Assisted Living (IOS Press, 2012), the Handbook of Smart Cities (Springer, 2021), and the forthcoming Smart Health Handbook (Sage/IOS Press, 2025). He is passionate about training postgraduate researchers and has supervised fifteen Ph.D. candidates (ten as Director of Studies) and directed nine M.Sc. by Research projects. He has also served as an External Examiner for twenty-three Ph.D. candidates in various countries. Prof. Augusto is dedicated to

the dissemination of knowledge and has been invited to deliver keynotes, tutorials, short courses, and research seminars worldwide. He has served on fourteen Advisory Groups/Panels and reviewed for National Funding bodies in eleven different countries. He is an active member of several professional societies, including ACM, IEEE, AAAI, and BCS. Additionally, he holds editorial roles such as Editorial Board member, Editor in Chief of three journals and of one Book Series.

Dr Lindsey Brodie is an academic at Middlesex University, where she teaches MSc courses in Information Systems Quality Management, Engineering Project Management, and Logistics & Supply Chains. Her doctoral research focused on impact estimation, a quantitative method for assessing how well designs meet requirements. Dr Brodie has enjoyed an extensive career in the industry, predominantly at ICL (now Fujitsu Siemens Computers). She began her career working on customer projects in the government, banking, and retail sectors, where she provided technical support for operating systems and developed software for operational support and product support of data management software. She later transitioned to consultancy, specializing in user requirements, IT strategy, and business processes. Beyond her industry roles, Dr Brodie has made significant contributions to academic literature, authoring numerous conference papers, journal articles, and book chapters. She has edited Tom Gilb's books 'Competitive Engineering' and 'Principles of Software Engineering Management.' Additionally, she co-authored the student textbook 'Successful IT Projects' with Professor Darren Dalcher. Dr Brodie is a member of the British Computer Society and holds the title of Chartered IT Practitioner.