Thèse de doctorat
Pour obtenir le grade de Docteur de
l'UNIVERSITÉ POLYTECHNIQUE HAUTS-DE-FRANCE
et de l'INSA HAUTS-DE-FRANCE

Informatique et applications

Présentée et soutenue par Maria Paula CORRÊA ANGELONI.

Le 11/12/2025, à Valenciennes

École doctorale :
École Doctorale Polytechnique Hauts-de-France (ED PHF n°635)
Unité de recherche :
Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielles et Humaines (LAMIH - UMR CNRS 8201)

Évaluation de la qualité d'usage des applications d'environnement intelligent par simulation basée sur des agents

JURY

Président du jury
- M. Tabia, Karim. Professeur des universités. Université d'Artois.

Rapporteurs
- Mme Dugdale, Julie. Professeure des universités. Université Grenoble Alpes.
- M. Winckler, Marco. Professeur des universités. Université Côte d'Azur, Polytech Nice.

Examinateurs
- Mme Martinie, Célia. Maître de conférences HDR. Université de Toulouse.
- M. Tabia, Karim. Professeur des universités. Université d'Artois.

Co-directrice : Mme Marçal de Oliveira, Káthia. Professeure des universités. LAMIH - UMR CNRS 8201.
Co-directrice : Mme Grislin-Le Strugeon, Emmanuelle. Professeure des universités. LAMIH - UMR CNRS 8201; INSA Hauts-de-France.
Co-encadrante : Mme Tîrnăucă, Cristina. Maître de conférences. Université de Cantabria.
Co-encadrant : M. Duque, Rafael. Maître de conférences. Université de Cantabria.

PhD Thesis
Submitted for the degree of Doctor of Philosophy from
UNIVERSITÉ POLYTECHNIQUE HAUTS-DE-FRANCE
and INSA HAUTS-DE-FRANCE

Computer science and applications

Presented and defended by Maria Paula CORRÊA ANGELONI.

On 11/12/2025, Valenciennes

Doctoral school :
Doctoral School Polytechnique Hauts-de-France (ED PHF n°635)
Research unit :
Laboratory of Industrial and Human Automation control Mechanical engineering and Computer science (LAMIH – UMR CNRS 8201)

Quality in use evaluation of smart environment applications by agent-based simulation

JURY

President of jury
- Mr. Tabia, Karim. Full professor. University of Artois.

Reviewers
- Mrs. Dugdale, Julie. Full professor. University Grenoble Alps.
- Mr. Winckler, Marco. Full professor. University Côte d'Azur, Polytech Nice.

Examiners
- Mrs. Martinie, Célia. Associate professor (HDR). University of Toulouse.
- Mr. Tabia, Karim. Full professor. University of Artois.

Co-director : Mrs. Marçal de Oliveira, Káthia. Full professor. LAMIH - UMR CNRS 8201.
Co-director : Mrs. Grislin-Le Strugeon, Emmanuelle. Full professor. LAMIH - UMR CNRS 8201; INSA Hauts-de-France.
Co-supervisor : Mrs. Tîrnăucă, Cristina. Associate professor. University of Cantabria.
Co-supervisor : Mr. Duque, Rafael. Associate professor. University of Cantabria.

Abstract

Titre: Évaluation de la qualité d'usage des applications d'environnement intelligent par simulation basée sur des agents

Mots clefs: qualité d'usage; modélisation et simulation basées sur des agents; exploration de données

Résumé: Les progrès de l'Internet des objets (IoT) ont permis le développement d'environnements intelligents, où des appareils interconnectés assistent les utilisateurs dans leur quotidien. Dans ce contexte, les maisons intelligentes offrent une solution pratique et économique permettant aux personnes âgées et handicapées de préserver leur autonomie tout en bénéficiant d'un soutien technologique. Garantir la qualité logicielle des applications destinées à ces environnements est essentiel, notamment pour répondre aux besoins des utilisateurs vulnérables. Selon la norme ISO/IEC 25019, l'évaluation de la qualité d'utilisation (QinU) doit prendre en compte différents contextes d'utilisation afin de vérifier si un système répond aux besoins des utilisateurs et atteint les objectifs et résultats escomptés. Les approches traditionnelles d'évaluation de la QinU nécessitent la participation directe de l'utilisateur, ce qui peut s'avérer particulièrement difficile pour les personnes souffrant de déficiences motrices. La prise en compte de multiples contextes d'utilisation accroît encore cette complexité. Pour relever ces défis, nous proposons une évaluation préliminaire de la QinU par simulation. Partant de ce principe, nous proposons une méthodologie utilisant la modélisation et la simulation à base d'agents (ABMS) pour reproduire le comportement et les interactions des utilisateurs au sein d'environnements intelligents. En intégrant des techniques d'exploration de données, la méthodologie ancre les simulations dans des modèles réels, permettant ainsi d'évaluer QinU selon divers profils d'utilisateurs et contextes d'utilisation. L'application de cette méthodologie démontre la faisabilité d'une évaluation précoce de la qualité logicielle des applications d'environnement intelligent, avant d'impliquer les utilisateurs réels en interaction directe avec leur environnement.

Title: Quality in use evaluation of smart environment applications by agent-based simulation

Keywords: quality in use; agent-based modeling and simulation; data mining

Abstract: Advances in the Internet of Things (IoT) have enabled the development of smart environments, where interconnected devices assist users in their daily lives. Within this context, smart homes offer a practical and cost-effective solution that allows older adults and people with disabilities to maintain independence while benefiting from technological support. Ensuring software quality in applications for such environments is essential, particularly when addressing the needs of vulnerable users. According to ISO/IEC 25019, quality in use (QinU) evaluation must consider different contexts of use to verify whether a system meets user needs and achieves the intended goals and outcomes. Traditional approaches for assessing the QinU require direct user participation, which can be especially challenging for individuals with motor impairments. Accounting for multiple contexts of use further increases this complexity. To address these challenges, we argue that QinU can be preliminarily evaluated through simulation. Building on this premise, we propose a methodology that employs agent-based modeling and simulation (ABMS) to replicate user behavior and interactions within smart environments. By incorporating data mining techniques, the methodology grounds simulations in real-world patterns, enabling the evaluation of QinU across diverse user profiles and contexts of use. The application of this methodology demonstrates the feasibility of assessing the software quality of smart environment applications at an early stage, prior to involving real users in direct interaction with their surroundings.

Discipline is freedom. Compassion is strength. To be kind is to be brave.

From the original *Disciplina é liberdade. Compaixão é fortaleza. Ter bondade é ter coragem.*

Há Tempos – As Quatro Estações, Renato Russo (Legião Urbana)

Acknowledgements

I would like to express my deepest gratitude to my family for their love, patience, and encouragement throughout this journey. In particular, I am especially grateful to my parents and grandparents, whose constant support, guidance, and belief in me have been a foundation of strength and motivation.

I am also thankful to the friends I met along the way, who stood by my side during both challenging and joyful times. Their companionship and encouragement have been invaluable in helping me stay resilient and focused.

I also owe my sincere appreciation to my four supervisors for their invaluable counseling, guidance, and trust. Their direction, feedback, and encouragement not only shaped this thesis but also gave me the opportunity to grow as a researcher. I am truly grateful for their generosity in sharing their time, knowledge, and expertise.

Last, but certainly not least, I would like to thank the Hauts-de-France Region, the European Union and CNRS (by the REUNICE and Eunice Vitral projects) for the financial of this study.

Contents

Abstract i

Acknowledgements v

List of figures xii

List of tables xiv

List of abbreviations xv

1 Introduction 1
- 1.1 Context and motivation . . . 1
- 1.2 Objectives and challenges . . . 6
- 1.3 Thesis methodology . . . 8
- 1.4 Thesis structure . . . 9

2 Fundamental concepts 11
- 2.1 Quality in use . . . 11
  - 2.1.1 Definitions and characteristics . . . 11
  - 2.1.2 Quality in Use evaluation . . . 14
  - 2.1.3 Context of use influence . . . 15
- 2.2 Agent-Based Modeling and Simulation . . . 18
  - 2.2.1 Agents . . . 18
  - 2.2.2 Agent-based modeling . . . 19
  - 2.2.3 Agent-based simulation . . . 21
  - 2.2.4 ABMS to study human behaviors . . . 24
- 2.3 Combining ABMS and DM . . . 25
  - 2.3.1 Data mining . . . 25
  - 2.3.2 Process mining . . . 26
  - 2.3.3 ABMS with DM techniques . . . 27
- 2.4 Smart environments . . . 28
- 2.5 Conclusion . . . 31

3 State of the art 33
- 3.1 Overview of the performed literature reviews . . . 33
- 3.2 Quality in use prediction with agent-based modeling and simulation or data mining . . . 36

3.2.1 Discussion . . . 36
3.2.2 Threats to validity . . . 39
3.3 Quality in use evaluation with agent-based modeling and simulation or data mining . . . 39
3.3.1 Discussion . . . 40
3.3.2 Threats to validity . . . 50
3.4 Quality in use evaluation of smart environment applications . . . 51
3.4.1 Discussion . . . 53
3.4.2 Threats to validity . . . 62
3.5 Conclusion . . . 62

4 QUESTS: a methodology for quality in use evaluations with agent-based modeling and simulation 65
4.1 Fundamentals of the approach . . . 66
4.2 Agent-based modeling . . . 71
4.2.1 Modeling the context of use . . . 72
4.2.2 Modeling data to evaluate Quality in Use characteristics . . . 78
4.2.3 Data mining . . . 86
4.3 Agent-based simulation . . . 88
4.3.1 Agent-based model and simulation observer . . . 89
4.4 Quality in Use evaluation through measures . . . 92
4.5 Potentially executable contexts of use . . . 94
4.6 Conclusion . . . 96

5 The application of the QUESTS methodology 99
5.1 Introducing the experimental studies . . . 99
5.2 The evaluated software applications . . . 101
5.2.1 The Help Me! application . . . 101
5.2.2 The ParkinsonCom application . . . 102
5.3 User profiles with Parkinson's disease . . . 104
5.4 First experimental study: validating the assumptions for QUESTS . . . 105
5.4.1 Agent-based modeling . . . 107
5.5 Second experimental study: implementing the whole QUESTS approach . . . 112
5.5.1 Agent-based modeling . . . 114
5.5.2 Data mining . . . 114
5.5.3 Modeling the context of use . . . 121
5.5.4 Modeling data to evaluate quality in use characteristics . . . 128
5.5.5 Agent-based simulation . . . 128
5.5.6 Agent-based model and simulation observer . . . 129
5.5.7 Executing the agent-based simulation . . . 131
5.5.8 Quality in use through measures . . . 137
5.6 Third experimental study: applying QUESTS on a third-party application . . . 140
5.6.1 The Overview, Design concepts and Details protocol . . . 143
5.6.2 Generated data . . . 146
5.6.3 QinU evaluation through measures . . . 151
5.7 Discussion and analyses of the results . . . 160
5.7.1 Extending the applicability . . . 160

5.7.2 Validating the results 164
5.8 Threats to validity 165
5.9 Applying the QUESTS methodology: a step-by-step guide 167
5.10 Conclusion 169

6 Conclusion and perspectives **171**
6.1 Final considerations 171
6.2 Contributions 173
6.3 Strengths, constraints and limitations 175
6.4 Future work 178

Appendix **181**

List of figures

1.1 Overview of the use of a smart environment application. 7
1.2 Methodology for QUESTS. 9

2.1 Quality in Use model in ISO/IEC 25010:2011 [40]. 12
2.2 Quality in Use model in ISO/IEC 25019:2023 [21]. 13
2.3 Outcomes from the use of a system in a context of use (adapted from ISO/IEC 25019:2023 [21]). 18
2.4 Classification of interactions between PM and ABMS [109]. 28

3.1 Research steps and their respective results. 38
3.2 Execution overview of the performed search. 42
3.3 Methodologies used for QinU evaluation. 42
3.4 Evaluation methodologies used in different kind of system. 45
3.5 Agents abilities applied in the studies. 47
3.6 Quality in use characteristic considered in the evaluations. 49
3.7 Quality in use characteristic evaluated in each kind of application. 49
3.8 Research steps and their results. 53
3.9 Number of studies published per year. 53
3.10 Types of systems evaluated in the secondary studies. 60

4.1 Targeted outcome for the employment of the approach. 67
4.2 Examples of agents to be modeled in the ABMS. 68
4.3 Actors and their roles in the execution of the methodology. 69
4.4 Components and their flows in the approach. 71
4.5 Agent-based modeling phase and its inputs. 72
4.6 Agent examples and their roles regarding explicit and implicit interactions. 73
4.7 Device agents are connected and share information. 75
4.8 Detailed data mining phase during the agent-based modeling. 87
4.9 Agent-based simulation phase and components. 89
4.10 Simulation and interaction log data and their contents. 91
4.11 QinU evaluation and its components. 94
4.12 QUESTS: a three-phase methodology for the evaluation of the QinU of software systems. 97

5.1 Screenshot of the webpage created for the proof of concept. 102
5.2 Screenshot of the ParkinsonCom application. 103
5.3 Implementation of the first experimental study. 107

5.4 Implementation of the first experimental study in NetLogo. 109
5.5 Activity diagram of the first experimental study. 110
5.6 Interaction (top left) and simulation (top right) log data and the spreadsheet (bottom center) calculating the QinU measure with data that comes from them. 111
5.7 Measured effectiveness and efficiency of the application for each user. . . . 112
5.8 Overview of the second experimental study implementing the QUESTS methodology. 113
5.9 ABM phase and its inputs in this implementation. 114
5.10 Detailed data mining phase during the ABM phase. 117
5.11 Markov chain modeling consecutive sensor activations starting at sensor M009 from the real dataset (#transitions and %). 119
5.12 House floor plan with blue spots to represent all the motion sensors. . . . 121
5.13 Smart environment modeled in the ABM. 125
5.14 ABS phase and its inputs in this implementation. 129
5.15 Activity diagram for the second experimental study. 132
5.16 QinU evaluation and its components. 138
5.17 Charts presenting the results of the evaluated QinU characteristics for eleven simulated individuals. 139
5.18 Overview of the second experimental study implementing the QUESTS methodology. 141
5.19 Main steps to generate the behavior of the user agents. 142
5.20 Activity diagram for the third experimental study. 147
5.21 Mean effectiveness of the completed tasks for each user agent (P1 to P26) and their respective parameters for tremors and falls. 155
5.22 Mean effectiveness for each room in the environment. 157
5.23 Mean efficiency of the completed tasks for each user agent (P1 to P26) and their respective parameters for tremors and slowness. 157
5.24 Mean efficiency for each room in the environment. 158
5.25 Mean user safety risk for each user agent. 159
5.26 Mean user safety risk for each room in the environment. 160

6.1 Motion sensors. 188

List of tables

2.1 Examples of QinU measures from ISO/IEC 25022:2016 [22]. 15

3.1 Structure to define the research goal for RR 1. 34
3.2 Structure to define the research goals for RR 2 and RR 3. 34
3.3 PICOC search string for the three RR. 35
3.4 Research protocol for RR 1. 37
3.5 Research protocol for RR 2. 41
3.6 Research protocol for RR 3. 52
3.7 Methods and techniques found in the primary studies. 55
3.8 Standardized questionnaires found in the secondary studies. 56
3.9 QinU characteristics found in the primary studies (#). 61

4.1 QinU characteristics from the Beneficialness category [21]. 80
4.2 QinU characteristics from the Freedom from risk category [21]. 82
4.3 QinU characteristics from the Acceptability category [21]. 84
4.4 Evaluation of QinU characteristics from ISO/IEC 25019:2023 [21]. 85
4.5 QinU characteristics from ISO/IEC 25019:2023 [21]. 93

5.1 Parameters set up for each user agent (from P1 to P26). 106
5.2 Parameters of the simulation in the first experimental study. 108
5.3 Probability of each message being sent according to the user's location. . . 108
5.4 QinU measures applied in this study (based on the ISO/IEC 25022:2016 standard [22]). 111
5.5 Parameters of the simulation. 122
5.6 Probability of falling according to *falling-frequency* and the user's location. 126
5.7 Sample of simulation log data from NetLogo for user agent P1. 133
5.8 Sample of interaction log data recorded in the Raspberry for user agent P1. 135
5.9 Outcome from the QinU measures for P1. 136
5.10 Sample of the simulation log data from NetLogo. 149
5.11 Sample of interaction log data. 151
5.12 QinU measure applied in the third experimental study (based on the ISO/IEC 25022:2016 standard [22]). 152
5.13 Measures calculated for each user agent (T1 = requesting for help, T2 = deactivating alarm, *imp* = implicit, *exp* = explicit) 154
5.14 Elements of the context of use and the maximum number of potential scenarios. 164

6.1 Proposed measures for QinU characteristics [22]. 183
6.2 Elements from the ODD protocol [84]. 184
6.3 Errors found in the “Sensor state” column of the motion sensors. 185
6.4 Atypical patterns from the temperature sensors. 186
6.5 Atypical patterns found in the dataset. 186
6.6 *pvalues* resulted from different functions. 187
6.7 Matrix comparing the solution outcome and the original source material. . 189

List of abbreviations

ABMS	Agent-based modeling and simulation
ABM	Agent-based modeling
ABS	Agent-based simulation
ADL	Activities of daily living
AmI	Ambient Intelligence
DM	Data mining
GUI	Graphical user interface
IT	Information technologies
IEC	International Electrotechnical Commission
IoT	Internet of Things
ISO	International Organization for Standardization
MAS	Multi-agent system
ML	Machine learning
PM	Process mining
QA	Quality assurance
QinU	Quality in use
UX	User experience

Chapter 1

Introduction

This chapter introduces the context and motivation of the research conducted in this thesis. It outlines the objectives and challenges encountered, describes the methodological approach adopted to address the problem, and concludes by presenting the overall structure of the thesis.

1.1 Context and motivation

As technologies become more ubiquitous in people's daily lives, software-based solutions are provided to make the users' routines easier [1], which makes them progressively rely on interconnected devices, such as the Internet of Things (IoT) [2]. The amount of devices linked to the Internet is increasingly growing [3], and vary from motion and temperature sensors to smart televisions and surveillance cameras [4, 5].

Enabled by the IoT, diverse environments can benefit from smart technologies. In recent years, smart buildings have received increasing attention as they employ advanced solutions to provide sustainable, economical, and comfortable living conditions for occupants [6]. The development of innovative energy management systems has, therefore, become both significant and urgent, seeking to achieve an optimal balance among energy consumption, carbon emissions, operational costs, and user comfort through the intelligent

scheduling of building resources. Importantly, the application of smart technologies extends beyond buildings, encompassing entire urban areas. To fulfill the requirements of a smart city, the efficient use of information and communication technologies is essential for managing data analysis, facilitating data communication, and supporting the implementation of complex strategies that ensure the secure operation of the city [7]. While these examples demonstrate the large-scale impact of IoT, the same principles also extend into the private sphere. One of the most direct ways individuals encounter and benefit from these technologies in daily life is through smart homes [8, 9], where IoT devices integrate seamlessly into everyday routines.

Smart home technologies are particularly relevant for senior citizens and people with disabilities, as they enable access to medical support when needed, promoting independence and facilitating individuals in maintaining their routines at home [10, 11, 12, 13]. Furthermore, they are often more economically viable than relying on in-person caregivers [14]. Living in a smart home is believed to support the well-being of older adults, even though some may be unaware of their health status or misperceive the role of the smart devices around them [15].

The former examples show how IoT technologies are integrated into daily life, connecting devices, spaces, and people in increasingly sophisticated ways [16]. Consequently, understanding the complexity of IoT software applications has become crucial for informing policy, guiding innovation, and supporting decision-making processes [17]. Social phenomena such as collective behavior and policy adoption unfold from the interactions of heterogeneous entities, often in unpredictable ways [2]. For instance, in smart homes, the simultaneous use of connected devices can create unexpected energy consumption patterns, sometimes conflicting with sustainability goals. As a consequence, it is paramount to test and evaluate the software quality of systems before they are available to end users. This is intended not only to ensure the safety of end users, but also to facilitate the approval and adoption of such applications by them [18]. Ensuring the quality in the use of software

applications in smart environments is essential if they are to be effectively utilized: without a solid reference on user evaluation, a system evaluation cannot be tackled well [19]. Guaranteeing the reliability of tools is even more important in cases of senior citizens and people with reduced mobility, considering a software flaw happening with them may be a worse menace than for people with no movement issues. This has become even more crucial as the world population grows older thanks to advancements in different fields, as medicine and technology, which increases the number of elderly citizens (considered as such from the age of 65) [20].

According to the ISO/IEC 25019:2023 [21], quality in use (QinU) is the "extent to which the system, when it is used in a specified context of use, satisfies or exceeds the stakeholders needs to achieve specified beneficial goals or outcomes". In essence, QinU considers the overall quality of the system in its operational environment for specific users and the execution of specific tasks. This is particularly crucial considering smart environments, where the contextual components are multiple and interacting. In these environments, a variety of sensors can be combined in different ways, while users interacting with them may display diverse traits and characteristics. Applications are often linked to specific locations – for instance, thermostats or motion sensors – which further influence how they are experienced. As a result, the context of use becomes a crucial factor and may vary considerably across such settings, a feature particularly evident in ubiquitous IoT applications.

The QinU model of the ISO/IEC 25019:2023 standard is made up of characteristics (such as usability and accessibility) related to the outcome of the human-computer interaction when a product is used in a specific context of use. This model measures the effect of the interaction between the user and the software, taking into consideration the user experience (UX). To that end, it is proposed the use of established measures, for which the ISO/IEC 25022:2016 standard [22] provides a concrete set.

When addressing the evaluation of interactive systems, usability issues are often considered, with effectiveness, efficiency, and satisfaction forming the core tripod of its definition

[23]. QinU evaluation and usability-related topics have been largely explored in literature with different methods and applications, such as mathematical simulations [24, 25], focus groups [26, 27], think aloud [28, 29], surveys [30, 31], and more.

One of the most common ways of evaluating QinU and usability is through questionnaires that can be answered by end users (e.g., [32]) or by domain experts (e.g., [33]). Among the published questionnaires, they may be standardized [34] or non-standardized. The procedures to employ questionnaires consist in either demonstrating the use of the evaluated system, or asking the participants to interact with the system themselves while being placed in the environment (if necessary) in order to evaluate in-vivo their engagement with the surroundings. Then, the participants are asked to answer a set of questions. The person in charge of the evaluation supervises the participants throughout the activities that collect data and later analyzes the results of the applied questionnaire.

Conducting an accurate QinU evaluation through the use of questionnaires requires a sufficient number of participants to represent diverse end users and contexts of use. Including people with disabilities is particularly important to ensure that the system accommodates a wide range of needs. Nevertheless, it can be challenging to group enough people with a wide range of traits, especially in the case of smart environments.

Evaluating the QinU in such spaces is not trivial, because the interaction differs from conventional systems since it should consider different types of input [5, 35]: the users do not necessarily need to make explicit efforts to interact with the computer (use of the mouse, keyboard, etc.), as they also do so while performing day-to-day tasks (moving around, using objects, etc.) that the pervasive system interprets as implicit inputs [36]. These, in turn, trigger proactive responses and implicit outputs, captured by integrated sensors. A conventional approach would involve placing participants directly in smart environments to interact with the systems, but doing so solely for testing purposes is often impractical and may even pose risks and safety concerns.

Moreover, the greater the number of sensors involved in the application under evalua-

tion, the more variations emerge in the environment, and consequently, in the context of use. We argue that, given these limitations, an alternative approach is to reproduce smart environments through simulation. By modeling both the environment and the users' interactions, simulations allow researchers to explore system behavior under realistic conditions without exposing participants to potential risks or logistical constraints. This approach can provide a first evaluation of the QinU in a controlled, safe, and cost-effective manner while still capturing the complexity of implicit interactions. Simulations can allow diverse scenarios to be tested, usability issues to be anticipated, and designs to be refined before implementation in real environments. Most importantly, simulations make it possible to easily replicate multiple contexts of use by adjusting parameters to generate scenarios tailored to specific needs.

In light of these considerations, the main research question that guides this thesis is:

How to evaluate the quality in use of smart environment applications through simulation?

One possible way to perform simulations is through the application of software agents. Agent-based modeling and simulation (ABMS) offers a powerful framework for simulating the behaviors and interactions of individuals within a given environment [37]. By modeling and simulating the interactions among autonomous entities (the agents), ABMS enables the study of emergent phenomena, the exploration of counterfactual scenarios, and the incorporation of heterogeneity and adaptation in a natural and intuitive way.

However, despite its potential, ABMS in the social sciences still faces several methodological challenges [38], including the formalization of qualitative theories, validation of simulation outcomes, and integration of empirical data into agent behavior. These obstacles demand a robust methodological protocol that balances realism with computational tractability.

We argue that, as ABMS allows for modeling a system as a collection of autonomous

agents, we can model users and smart devices that interact with one another. This aligns well with our objective of capturing the complex behaviors typical of smart environments. Furthermore, ABMS operates effectively at a microscale, enabling detailed representation of individual agents' behaviors and their emergent collective dynamics. This granularity is essential to simulate how people interact with applications and devices, which is critical for generating meaningful data.

1.2 Objectives and challenges

The main objective of this thesis is to define a methodology for evaluating the QinU through simulations with software agents and quality measures. In order to reach this goal, three scientific challenges need to be addressed:

i. to model and simulate user interactions in smart environments using agents, and considering the variability of the context of use;
ii. to ground the simulation on real patterns;
iii. to calculate the QinU measures using log data collected from simulations.

To avoid placing potential end users in a testing environment at a first instance of QinU evaluation, and to overcome *scientific challenge (i)*, we argue that we could model users, as well as the smart environment around them, through ABMS to transmit events to the software application under evaluation. This would simulate a human-computer interaction, replacing then the involvement of real people using the evaluated application at this stage. Furthermore, an ABMS makes it possible to reproduce multiple contexts of use with greater ease than through the involvement of real users.

As the simulation is designed to replace the actual utilization of the software application at a certain point during the software quality evaluation, it is essential to provide an adequate "approximation" of the real-world data generation process (see [39] for an overview

of methodological problems arising in the empirical validation of ABMS). To overcome *scientific challenge (ii)*, we apply data mining (DM) techniques in our approach to find patterns in real data and feed them to the ABMS.

For the QinU evaluation, and to overcome *scientific challenge (iii)*, we use log data generated by the agent-based simulation, which gathers information about the simulated events of interaction with the targeted application. In doing so, it is important to evaluate different combinations of context of use across multiple scenarios, varying not only the users and the tasks they perform but also the environment and the resources included in the simulation. This allows for diverse outcomes that can be used to measure the QinU.

In summary, the goal is to define a model for software agents to simulate both a resident inside their smart environment they live in, as well as the interconnected devices around them. For such to be as close to reality as possible, we will feed this model with patterns found in real-world data through data mining techniques. From the log data produced as outcome, we will apply QinU measures suggested by ISO/IEC standards.

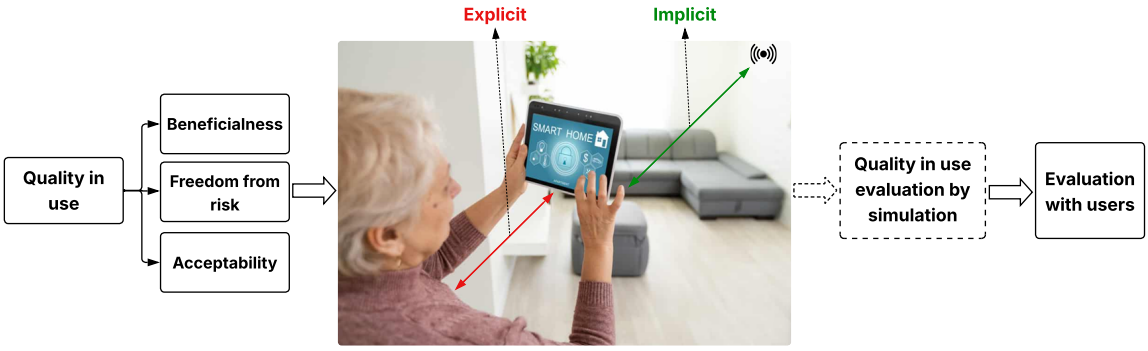


Figure 1.1: Overview of the use of a smart environment application.

Figure 1.1 displays the QinU characteristics [21] – beneficialness, freedom from risk, and acceptability – that impact the use of software applications. It also disposes representations of both types of interactions found in smart environments: *explicit* interaction – with the lady providing input through the tablet's touchscreen – as well as *implicit* interaction – a sensor placed at her house receiving information. The element presented in-between the

dotted line (*Quality in use evaluation by simulation*) shows where the main contribution of this thesis – the proposed methodology – should take place in the overall software engineering process. It creates an additional step, providing an approach that gives quality assurance (QA) analysts the chance to examine the QinU according to their established thresholds before including potential end users in the process.

The main goal of the methodology proposed in this thesis is to support QA analysts when deciding whether the system is ready to go through final evaluations with real-world potential end users, or if it needs to be improved. According to their analyses, the application may go back to the design and development phase, or move forward to the usability tests. This could save time and money for the stakeholders involved, and most important, could avoid potential situations with some hazard for the end users in place.

The proposed methodology has been named QUESTS, which is an acronym for ***Q***uality in ***U***se ***E***valuation of ***S***mart environment ***T***echnologies by agent-based ***S***imulations. This name evokes the idea of a journey or mission, analogous to how the evaluation process is a structured quest for understanding how smart environments perform in real use scenarios.

1.3 Thesis methodology

Figure 1.2 presents the main phases to solve the scientific challenges detailed in the previous section. At the initial stage, literature reviews are performed to study the state of the art and analyze if the problem raised in this thesis had not been explored yet.

Then, to define the QUESTS methodology, we focus on the definition of patterns and possible ways of implementing data mining for this purpose, as discussed in *scientific challenge (ii)*. From there, the *agent-based modeling* is implemented and followed by the *agent-based simulation*. These two steps take place to define software agents that represent a resident of a smart home, as well as the smart devices found in the environment they live in, as brought up in *scientific challenge (i)*.

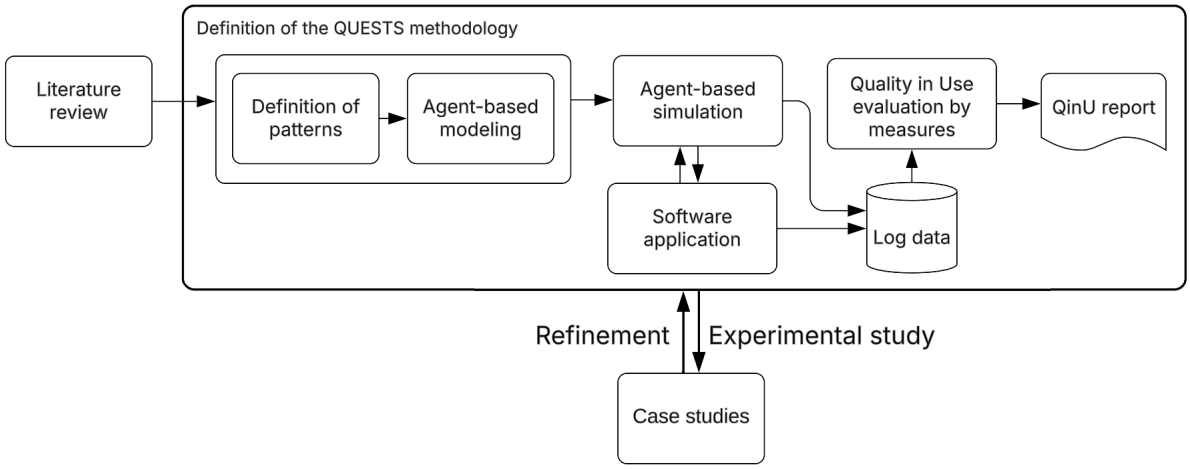


Figure 1.2: Methodology for QUESTS.

In the agent-based simulation, events would take place and trigger interactions on the targeted application. This generates log data to be applied to QinU measurements in order to evaluate pre-established characteristics, which solves *scientific challenge (iii)*. Lastly, a QinU report is created to be used as supporting material by a QA analyst. For that, QinU measures will be used or adapted from the ISO/IEC 25022:2016 standard.

For each iteration of the approach, a case study was performed, which allowed us to refine and improve the QUESTS methodology.

1.4 Thesis structure

The remainder of this thesis is organized as follows.

Chapter 2 presents the fundamental concepts applied to the methodological approach. It explains and discusses concepts of different domains, such as software quality, agent-based modeling and simulation, data mining and smart environments.

Chapter 3 provides literature reviews regarding the QinU evaluation of software applications for smart environments using or not software agents and data mining in order to analyze the state of the art on this subject.

Chapter 4 establishes the approach of the QUESTS methodology with a generalization of all steps and features involved presented in the definition of the QUESTS methodology in Figure 1.2.

Chapter 5 presents the implementation of the QUESTS methodology in three different case studies according to the approach defined in the previous chapter. Each experimental study allowed then the refinement and improvement for the one that followed. Furthermore, it also provides possible threats to the validity of the experimental studies and includes a step-by-step description to apply the proposed approach.

Lastly, Chapter 6 concludes with final considerations, contributions, and a discussion of the strengths and limitations of the QUESTS methodology. These, in turn, provide a basis for reflecting on alternative perspectives and directions for future work.

Chapter 2

Fundamental concepts

This chapter introduces the fundamental concepts involved in this thesis. It includes definitions from: (i) quality in use (QinU), according to published standards that categorize software quality and its elements; (ii) agent-based modeling and simulation; (iii) data mining, and specifically, process mining; (iv) agent-based modeling and simulation combined with data mining techniques; and, (v) smart environments.

2.1 Quality in use

This section details important elements for the concept of QinU. Section 2.1.1 presents the definitions and the characteristics that are a part of it; Section 2.1.2 outlines how QinU can be measured and evaluated; and Section 2.1.3 discusses how the context of use impacts the QinU.

2.1.1 Definitions and characteristics

Software quality is related to the ability of a software system to meet the customer's needs [40]. To achieve this, assessments should be conducted throughout the software de-

velopment process. During development, internal software quality evaluations are typically performed using static measures of intermediate products. In the testing phase, external software quality evaluations are carried out, usually by analyzing the behavior of the code during execution. Finally, assessments should also be made during software use, whether in real-world or simulated environments. This last case is known as QinU evaluation.

QinU depends on the context in which the system is used, and not strictly on the product quality of the system in itself. The QinU model serves to assess the impact experienced by users when interacting with information technology (IT) systems, which means it has the purpose to "define, measure, evaluate, and improve the quality of systems and software products and IT services" for users [21].

This model may be employed by developers, QA analysts, independent evaluators, and all those responsible for the evaluation of software quality. Moreover, QinU can also be considered during the conceptual design and development stages, influencing requirements, guiding formative and summative prototype evaluations, and informing control processes [22].

With the publication of the ISO/IEC 25000 SQuaRE series in the year of 2011, the QinU model considered five main characteristics, according to the ISO/IEC 25010:2011 standard [40] (see Figure 2.1).

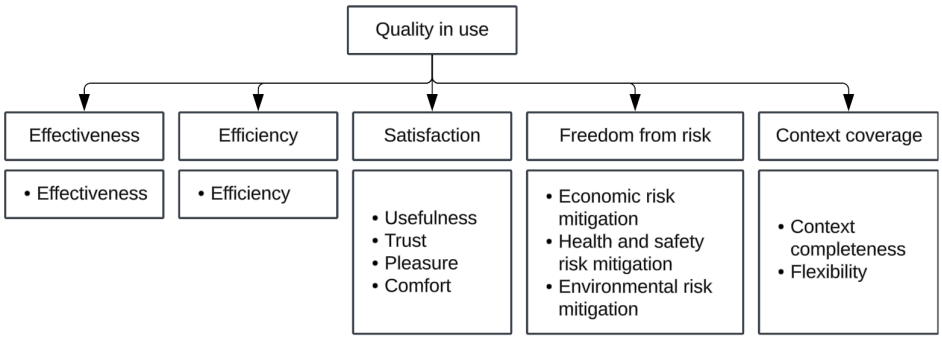


Figure 2.1: Quality in Use model in ISO/IEC 25010:2011 [40].

Ever since then, the QinU model has been revised and updated "to expand the target of QinU to not only direct users but also indirect users and other stakeholders" [21]. The

ISO/IEC 25019:2023 standard, the latest one on the topic, splits QinU into three different main characteristics, which were chosen to represent all stakeholder needs. These characteristics are then divided into sub-characteristics (see Figure 2.2), and can be defined as:

- **beneficialness** – extent of benefit that results from the use of the system;
- **freedom from risk** – extent to which the system mitigates the potential risk to economic status, financial values, human lives and health, society in general, or the environment;
- **acceptability** – extent to which a human response is favourable regarding accepting the system.

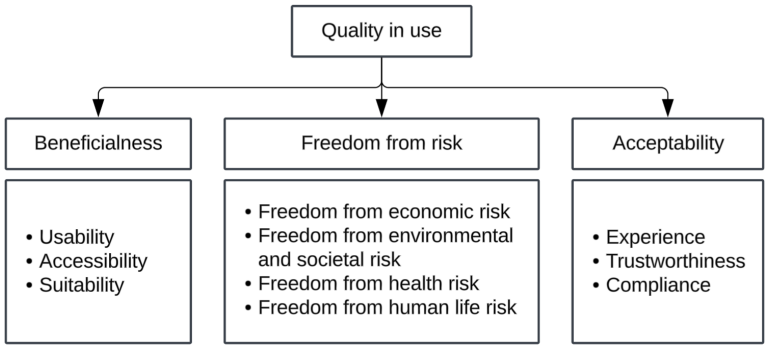


Figure 2.2: Quality in Use model in ISO/IEC 25019:2023 [21].

Some of the changes in the QinU model from the previous standard to the latest one that can be pointed out are:

- the previous main characteristics effectiveness, efficiency and satisfaction are now combined in usability, which is a sub-characteristic of beneficialness;
- freedom from risk remains as a main characteristic, but now has a wider range of sub-characteristics;
- context coverage is not a separate characteristic anymore, as this factor should be considered when evaluating all established characteristics;
- acceptability is now a main characteristic as it was judged that its sub-characteristics are important to explain the influence on organizations and society at large, and not only on direct users of the targeted system.

2.1.2 Quality in Use evaluation

For the QinU characteristics to be evaluated, the ISO/IEC standards propose the use of quality measures. The ISO/IEC 25010:2023 standard [41] suggests a set of general measures to be applied to the different kinds of characteristics. Such measures can then be specified towards the system under evaluation or adapted according to what is necessary for the measurement to take place, as the standard "does not provide any definition for a concrete evaluation method" [42]. For that, sub-sub-characteristics can be established by the user of the QinU model to have a more granular representation of the quality attributes. The ISO/IEC 25022:2016 standard [22] is entirely dedicated to the measurement of QinU and presents various proposals for measuring software quality.

In particular, the ISO/IEC 25022:2016 standard [22] proposes measures for the QinU characteristics as they had been defined in the ISO/IEC 25010:2011 standard [40] (presented in Table 6.1 in Appendix A). However, there is no version yet of any ISO/IEC standards that propose measures for QinU characteristics as defined by the most recent standard, ISO/IEC 25019:2023 [21]. To learn which data should be collected for evaluating specific QinU characteristics, it is necessary to analyze and select potential QinU measures. For this purpose, the ISO/IEC 25022:2016 standard [22] serves as a baseline; nevertheless, due to its general nature, the defined measures can either be applied directly or adapted to suit the specific software application being evaluated. Moreover, researchers can create and propose their own measures, as seen in [43].

Usability is a typical characteristic to be considered for evaluation when it comes to assessing the quality of interactive systems. In fact, the only previously recognized standard for this type of evaluation was the ISO 9241-11:2018 standard [44], solely focused on usability. Now, this characteristic has been integrated in the SQuaRE standards. According to the ISO/IEC 25019:2023 [21], usability is defined as the extent to which a system can be used by "specified users to achieve specified goals with effectiveness, efficiency, and satisfaction

in a specified context of use".

Table 2.1: Examples of QinU measures from ISO/IEC 25022:2016 [22].

ID	Name	Description	Measurement function	Method
Ef-1-G	Tasks completed	The proportion of the tasks that are completed correctly without assistance	X = A/B A = number of unique tasks completed B = total number of unique tasks attempted	Measure user performance
Ey-1-G	Task time	The time taken to successfully complete a task	Y = T T = task time	Measure user performance
SUs-1-G	Overall satisfaction	The overall satisfaction of the user	Z = $\sum$ Ai; Ai = response to a question	Questionnaire

Table 2.1 presents a set of measures that compose usability taken verbatim from the standard. Effectiveness concerns the accuracy and completeness with which the identified goals are achieved, efficiency refers to the resources used to achieve said goals, and satisfaction relates to the extent to which a stakeholder perceives the results of the use of a system and how well they meet their expectations [21]. Depending on the method, QinU can be assessed either by measuring the characteristics computed from the interaction log data (whether the product is in real or simulated use [22]) or by the user's perception.

2.1.3 Context of use influence

As mentioned previously, QinU depends on the context of use. According to the ISO/IEC 25063:2017 standard [45], information about the context of use serves as a foundation for designing a product that is usable in the intended context of use, while also supporting the maintenance of a human-centered design focus within the project.

Taking into consideration the context of use implies that, when evaluating the QinU of a system, there is the need to identify deviations from the initially assumed scenario. Therefore, comparisons of the QinU of systems are only valid when the measures are applied in

the same context of use. The changes in the context of use are monitored through quality evaluations in order to notice gaps in the initially specified scenario. These gaps are then identified and informed to the next quality improvement cycle [21].

The context of use is a combination of (i) users, (ii) goals and tasks, (iii) resources, and (iv) environment in which the system is used [23]. The usability that results from the use of the system depends on each of these components. It may also consider interdependencies and interactions between the object of interest and other systems, products or services. Hence, it is necessary to identify relevant characteristics and their variability in the actual or intended contexts of use, as different contexts of use can cause different usability aspects for the same system.

Regarding (i), the **users** within the context of use can be people who interact with different parts of an identified object of interest (i.e., the evaluated system), including: people who operate the system; people who support the system's continuing operation (e.g., maintenance workers); people who use the output of the system, even though they are not involved with the production of that output (e.g., a bank costumer that uses the bank's mobile application is a user of the output produced by the bank's system).

The users can be assigned to four different categories: direct, indirect, primary and secondary [40]. These are used to classify the ways in which the users interact with the evaluated system. Furthermore, the users may have distinct characteristics, which may include physical, sensory, psychological and social factors.

As for (ii), the **goals** within the context of use are the intended outcomes to be achieved by different tasks in the system. They can come from different sources, such as the users, other stakeholders, an organization, regulations. The **tasks** within the context of use consist of one or more activities performed to achieve a specified goal. Different activities can provide different ways of achieving the same goal, which then can lead to different levels of usability. The characteristics of activities that form the tasks, such as frequency, duration, complexity, cultural conventions, and relationships (as dependencies and interdependen-

cies) are all part of the context of use.

Concerning (iii), the **resources** within the context of use include any reusable or expendable resources that are needed to achieve specified goals. Reusable resources include equipment and applications (which include hardware, software and other physical items, such as furniture or battery chargers), information (which includes data that helps the user when completing tasks), and support (which includes services, such as assistive technology) that are used with the evaluated system while the user performs the task and that are integral for finishing it. Expendable resources include available time, human effort, financial resources, and materials.

And lastly, with respect to (iv), the **environment** within the context of use includes the technical, physical, social, cultural and organizational environments.

- the *technical* environment includes components that enable or constrain the use of the reusable and expendable resources, such as furniture, control devices, energy (e.g. electrical) and connectivity (e.g. Internet);
- the *physical* environment may include geographical features, the spatial, acoustic and visual conditions, weather circumstances;
- the *social, cultural and organizational* environment includes the roles and relationships between people, organizational structures, the language, the legislation.

Figure 2.3 summarizes the perspective of the QinU model. The four components of the context of use (user, environment, resources, goals and tasks) surround the object of interest (i.e., system, products or service in Figure 2.3), which influences the use. This impacts in the outcome of the use, which takes into consideration QinU characteristics, including usability (sub-characteristic of beneficialness that combines effectiveness, efficiency and satisfaction [21]) and other attributes, such as the characteristics defined by the ISO/IEC 25019:2023 standard.

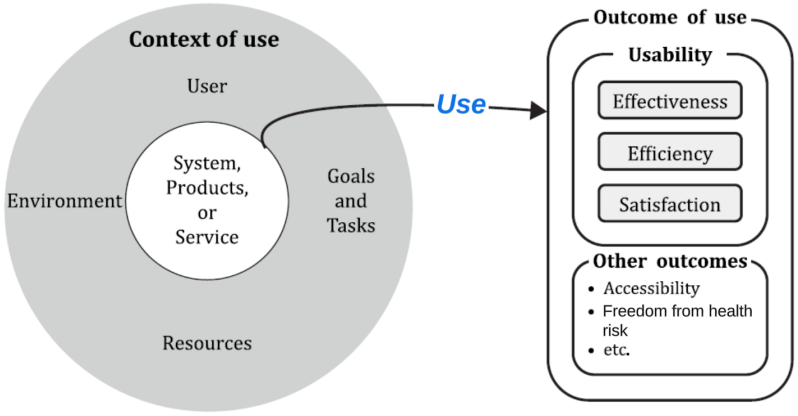


Figure 2.3: Outcomes from the use of a system in a context of use (adapted from ISO/IEC 25019:2023 [21]).

2.2 Agent-Based Modeling and Simulation

This section outlines the components of an ABMS: Section 2.2.1 provides definitions about agents; Section 2.2.2 details agent-based models, while Section 2.2.3 discusses agent-based simulations; and lastly, Section 2.2.4 presents scenarios where ABMS is applied to represent human behavior.

2.2.1 Agents

There are multiple definitions of what an agent is. For instance, Russel & Norvig [46] defined that an agent is any entity that perceives its environment through sensors and acts upon it via actuators. For example, human agents use sensory organs and limbs, robotic agents use devices like cameras and motors, and software agents process digital inputs and outputs. In all cases, agents interact with their environment through a sensing and acting interface.

Franklin & Graesser [47] defined that autonomous agents are situated in some environment. If the environment changes, an entity may no longer function as an agent. For instance, a robot equipped solely with visual sensors in an environment without any lights would no longer work as an agent. Whether a system is considered an agent depends on

the environment in which it operates.

Jennings, Sycara and Wooldridge [48] defined an agent as a computer system, situated in an environment, capable of flexible autonomous action in order to meet its goals. They established three key concepts: autonomy (the system must operate independently, without requiring direct intervention from humans or other agents); situatedness (the agents gather sensory input from its environment and are capable of performing actions that alter that environment); and flexibility (the system is responsive and proactive). Similarly, Baldwin, Sauser and Cloutier [49] have established that agents are expected to exhibit flexibility and operate autonomously from their environment, at least within the contexts under consideration. They may also possess heterogeneous characteristics and autonomy. These authors also define that an agent is an independent software entity that models actors, which may be people, animals, organizations, devices, or other systems.

The definitions provided by Jennings, Sycara and Wooldridge [48, 49] are the closest to how agents are applied in this thesis. This is because we see them specifically as autonomous software entities – regardless of what they represent – situated in an environment.

2.2.2 Agent-based modeling

A model is a construct used to represent another entity for a specific purpose. It is utilized when directly studying the actual system is impractical or infeasible – due to factors such as high cost, time constraints, potential disruption, safety concerns, or legal restrictions [50].

For an ABMS, the "modeling" step refers to designing a model that represents real-world phenomena, minimizing the gap between the model and its real-world reference [51]. This includes establishing which entities and situations will be modeled, and setting their respective variables and attributes. Serena et al. [52] established that there are two broad classes of models: *deterministic*, if the occurrences in the simulation are entirely predictable

from the initial conditions, or *stochastic*, if the occurrences depend on random choices.

Models can be categorized according to the scale or level of details in the representation. In ABM, the individuals (the agents) and their behaviors are modeled, in addition to their actions and interactions with the environment and between them. Interacting agents form a multi-agent system (MAS) and the macro-level behavior of the system results from the dynamics occurring at the micro-level [53]. For example, in a traffic simulation, the individual behaviors of drivers would be modeled at a micro(scopic) level, whereas only their average density or velocity would be modeled at a macro(scopic) level.

Through an ABM, the system can be represented at the micro-level through the collection of autonomous entities that are the agents, which can perceive their surroundings and take actions accordingly [46]. This is especially useful to model complex systems, with a growing interest from ecosystems, social systems and health domains [54]. Microscale models focus on individual environments (e.g., buildings or stadiums) and include frameworks like the Intelligent Driver Model and agent-based models. Macroscale models operate at larger scales (e.g., cities) and use approaches such as the Lighthill-Whitham-Richards and Aw-Rascle-Zhang models. Certain models can be classified as "meso", representing an intermediate scale; some others integrate multiple levels and are called "multilevel" or "multiscale" [55].

A common distinction in modeling approaches lies in the source of knowledge they rely on. Data-driven modeling builds models from data, enabling quick and cost-effective development, especially for complex tasks like language translation. Its main goal is prediction, but it often produces opaque models that are hard to validate. In contrast, knowledge-driven modeling relies on expert knowledge and theoretical components, resulting in more transparent but less scalable models [56]. For example, in financial fraud detection, a data-driven model may be trained using extensive historical data and ensemble learning algorithms, detecting fraud through pattern recognition. In contrast, knowledge-driven systems rely on handcrafted rules defined by domain experts to flag suspicious transactions [57]. Agent-

based models, too, can be implemented using either approach, depending on how agent behaviors are specified.

Validation is the process of assessing whether a model sufficiently represents the system under study for its intended purpose, serving as a critical step in establishing the simulation model's credibility with stakeholders. Multiple types of ABMS validation can be found in the literature. Some of the most applied ones include [58]: data analytics, which ensures transparent handling of input and output data while supporting input and data modeling; empirical/data-driven validation, which compares simulation outcomes with real-world data or expected values when true data are unavailable; sampling, which systematically explores the simulation space through predefined experimental designs; and visualization, which links the simulation to the real system by representing interactions, geospatial information, and other contextual elements to support feedback.

The choice of validation approach depends on both the simulation and the specific context of the study, meaning that a method suitable for one domain may be entirely unsuitable for another. For instance, sampling is well-suited for engineering applications where systematic exploration of simulation parameters aligns with controlled experimental designs [59], while visualization is particularly valuable in smart environments or healthcare simulations, where presenting interactions and geospatial context aids in interpreting complex user-system dynamics [60].

2.2.3 Agent-based simulation

Modern scientific fields employ a varied range of computational simulations to analyze the impact of different scenarios in many areas, such as aeronautics, business, chemistry, health, physics, social sciences, and more [61].

Simulation is experimentation with a model that imitates key behaviors of a real system [50, 62]. Like field testing, it enables observation and analysis, but uses a computational

model instead. Through simulation, users generate and study behaviors to understand, summarize, or generalize system dynamics [50]. When effectively applied, a simulation serves as a powerful tool for analyzing alternative actions and forecasting their impacts, in order to better allocate resources. Its versatility allows it to be applied to any system that aligns with the principles of simulation approaches [62].

From an established agent-based model, an agent-based simulation can be executed. If it follows a deterministic model, the events that happen throughout the simulation can be predicted from the conditions and parameters initially established. In contrast, if an agent-based model presents some stochasticity, the interactions between the agents will impact the global output of the simulation.

One of the strengths of ABMS is the possibility to simulate entities having heterogeneous and autonomous behaviors, which can be applied to emulate human actions in different scenarios. For instance, many studies in the ABMS field aim at predicting phenomena emerging from large crowds during emergency situations [63, 64, 65, 66, 67, 68]. Crowd behavior theories based on individual decision-making are most effectively represented using agent-based models, because each person can be modeled individually [56]. Even at a smaller scale, ABMS can provide support to analyze what can result from the interaction inside systems that include humans, such as human organizations [69, 70] or mobility choice in human daily activities [71].

ABMS can be applied for various reasons, including the difficulty of representing highly complex systems with traditional modeling tools and the availability of microdata that enables individual-based simulations [72]. Per Klügl [73], ABMS is a flexible method to describe and simulate a system made of behavioral entities, while also providing a natural framework for tuning the complexity of the agents, such as degree of rationality and rules of interactions.

Various software tools are available for developing ABMS, such as NetLogo [74] and the GAMA Platform [75], each with its own programming language. Alternatively, the

Recursive Porous Agent Simulation Toolkit (Repast) [76] supports multiple programming languages, including Java and Python. Other ABMS platforms are also based on Java, such as AnyLogic [77], Mason [78], and Swarm [79].

Standardizing ABM is crucial for advancing theory and methods, as it helps build consensus on structure, analysis, and data use. However, over-standardization can undermine ABM's key strength: their flexibility [80]. Reulier, Bonnet & L'Homme [81] performed a study comparing three ABMS platforms – NetLogo, GAMA, and Repast Simphony – to which one would be the best to execute the LASCAR model, which simulates interactions between surface runoff and agricultural landscape elements. Results show that NetLogo performs best overall for the LASCAR model, particularly in terms of execution time; however, while it is well-suited to LASCAR's discrete grid environment and visual rendering needs, Repast outperforms it when visual rendering is removed.

With the dramatic increase in the quantity of data and processes used in modeling projects, there has been a rise in the necessity of a protocol to standardize shared information about agent-based models [82]. To facilitate the replication of other researchers' work, Grimm et al. [83] designed a standard protocol to describe ABMS named Overview, Design concepts and Details (ODD), updated in Grimm et al. [84], with the purpose of providing a pattern to be followed by researchers when describing their own ABMS in a not overly technical manner. Besides easing the model replication for others, it also assists the authors to provide complete descriptions of their project, considering that offering incomplete information would go against the main requirement of science that says "materials and methods must be specified in sufficient detail to allow replication of results". Appendix B presents a classical template of the ODD protocol.

2.2.4 ABMS to study human behaviors

ABMS is well-adapted to represent phenomena that come from interaction behaviors, allowing "to take into account heterogeneity and complexity both in the individual layer and in the environment layer" [85]. Indeed, ABMS is suitable to simulate human behaviors that respond to the context as well as to the behaviors of other agents [86].

Reproducing human behavior has been "the missing element in descriptive models of the important systems that people urgently would like to model, forecast or predict, and test interventions upon" [51]. The agent-based approach offers a rather transparent representation while easing the managing and monitoring of the simulation, including non-computer scientists in the modeling process (e.g., end users), and encouraging stakeholders to think about their own role and practices in the process [87].

For these reasons, ABMS can be applied to model some humans' traits in a smart environment as well as their interaction with the software systems in their surroundings. In a literature review [87] about the simulation of human behavior in residential buildings with ABMS, the aim of most studies is to address energy related problems. Even though two of those studies represented scenarios with smart homes, they did not target specifically the human-computer interaction and did not aim at evaluating such relations. Moreover, in almost half of the observed studies, the agent behavioral model of decision-making is based on probabilities (see, for instance, [88, 89]).

ABMS are mostly employed to represent human behavior in social sciences, even though some researchers in the field are still skeptical about the use of computer-simulated models and demand more broadly accepted and standardized methodological frameworks [90]. Social science theories often lack the precision needed for formalization into coding, which makes translating them into agent behavior challenging due to logical gaps and vague descriptions of complex phenomena [38].

In order to model human behavior, the ABMS studies employ methods that belong

mainly to two different groups. Methods from the first group try to reflect the human cognition processes, from very simplified ones based on fundamental needs (hunger, tiredness, etc.) [91, 92] to more sophisticated ones, such as the Beliefs-Desires-Intentions (BDI) models [93, 94]. Methods from the second group are based on data: when the amount of data is sufficient, statistics or machine learning results about human actions can guide the model design [71]. In such cases, data mining becomes essential for extracting meaningful patterns that can guide the ABMS.

2.3 Combining ABMS and DM

This section outlines the concepts of data mining (DM), process mining (PM), and their combination with ABMS: Section 2.3.1 discusses DM, while Section 2.3.2 describes specifically PM; then, Section 2.3.3 details the merging of DM techniques with ABMS.

2.3.1 Data mining

Data mining is defined as "the process of discovering patterns in data" [95] and refers to a domain that solves problems by analyzing data gathered in databases. DM leverages both machine learning algorithms (e.g., association rule learning) and statistical techniques (e.g., regression analysis and anomaly detection) to extract meaningful patterns and insights from large datasets [95].

There are two main types of DM goals [96]: (i) descriptive, which provides a better understanding of the data with no specific target variable; and (ii) predictive, which is built to predict a particular outcome [97]. DM often applies machine learning (ML) techniques to discover patterns in data. This has become even more common after the rise of deep learning (a more specialized branch of ML [98]), which led to significant breakthroughs in engineering practice and scientific research [99]. The most used techniques for (i) are

clustering, association rules, and correlation analysis; while for (ii) are classification and regression [96].

In this thesis, we will focus specifically on process mining techniques, a branch of data mining aimed at uncovering process characteristics from event data.

2.3.2 Process mining

Process mining (PM) is a field that focuses on developing techniques to extract insights and analyze processes using event data [100, 101], and it has emerged as a combination of DM and process modeling [102].

Daily living processes exhibit considerable variability, which makes probability-based approaches a frequent choice for recognizing such patterns in smart environments [103]. These approaches are also widely applied across domains such as business, telecommunications, and healthcare, which has prompted the emergence of PM, a field that bridges process modeling and DM [104] and takes advantages of large event-based datasets [105]. In the case of healthcare, PM has been used to uncover typical procedures followed by particular groups of patients [106], as well as to create visual representations that highlight how different forms of care are interconnected [107].

The starting point for any PM procedure is an event log, which holds information about a real execution of a process that was recorded and stored by IT systems. This log can be explored for insights about how a process is executed in reality [108]. Processes are usually defined by time-stamped events that establish the start and end of instances. A process instance follows a sequence of activities "based on probabilities, logic, and rules" [105]. Hence, PM often involves exploring trajectories or detecting anomalies.

While PM is mainly regarded as a descriptive technique within DM (refer to Section 2.3.1), aimed at understanding and visualizing past events, it can also be employed for predictive purposes, such as forecasting the occurrence of recurrent events. The main PM

techniques can be divided into three categories [109]: (i) discovery, which produce a model without using any a priori information (i.e., extracting process models from an event log [102]); (ii) conformance, which compare events recorded in a log to a model and vice versa (i.e., monitoring deviations when comparing model and event log); and (iii) enhancement, which improve or extend a process model using event logs (i.e., adapting process models and displaying performance) [110, 111].

Furthermore, four perspectives of PM have been defined, establishing different possible dimensions to be explored by this technique [112]: (i) control-flow, e.g., the order of activities; (ii) organizational, e.g., people and structures involved; (iii) time, e.g., duration and frequency; and (iv) data, e.g., individual instances.

2.3.3 ABMS with DM techniques

The domains of ABMS and DM have been combined before because of different benefits. Simulations based on real-world data have been employed in data-driven modeling processes [113] to generate behavioral patterns for the agents, as well as to initialize simulations. Real-world data is also commonly used as a provider of empirical validation (by comparing it to the simulation results [114, 115]). This brings numerous advantages, as listed in [116]: a more accurate representation of real systems, as data may reproduce previously observed behavior well and with transparency; ease of the modeling process, as it is usually difficult to translate human behavior in ABMS; and the use of current data resources, enhancing the usefulness of stored dataset and grounding the ABMS in real-world patterns.

Turgut & Bozdag [116] established different cases that can link empirical data, ABMS and ML techniques. The use of PM, specifically, in combination with ABMS offers several possibilities [109]. The merge of these two areas can be classified into different categories (see Figure 2.4), as follows:

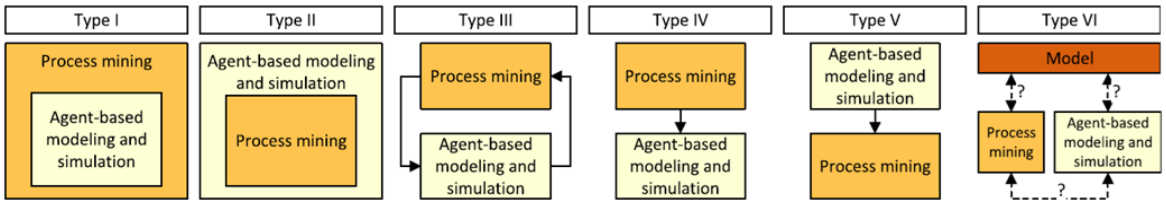


Figure 2.4: Classification of interactions between PM and ABMS [109].

- **Type I** – PM with nested ABMS-based iterations. This involves performing one or more (complete) ABMS-based iterations (like simulation runs) in all (or part) of the iterations of a PM method. The goal is to achieve a PM objective while utilizing supportive ABMS techniques;
- **Type II** – ABMS with nested PM-based iterations. The goal is to achieve an ABMS goal by embedding PM techniques;
- **Type III** – alternating PM and ABMS. This involves sequentially progressing both modules, with each iteration involving the complete or partial execution of both. The ABMS process and the PM procedure are mutually interdependent;
- **Type IV** – PM followed by ABMS. The goal is to use the output produced by PM for ABMS purposes;
- **Type V** – ABMS followed by PM. The goal is to use the output produced by ABMS for PM purposes; and,
- **Type VI** – complex interactions between PM and ABMS. This category includes ways of interaction that do not match any of the other categories.

The approach that is closest aligned with what is applied in this thesis is the use of empirical data as input for an ABM (type IV).

2.4 Smart environments

Smart environments are equipped with a network of devices that possess “computation capabilities and communication technologies and interact with each other”, and that may

be controlled by their users in a remote way [117, 118]. With the advances of the IoT, sensors and other interconnected devices are able to share data, which greatly impact such places [119]. Smart environments are composed of different elements that include sensors and computer hardware/software, all "interconnected through communication networks to exchange data and provide services" [120]. Such sensors are connected with on-site software applications to provide services adapted to both the user and the environment.

A smart environment may also be called differently to specify its kind, such as "smart home, smart office, smart building, smart factory, smart hospital, smart airport, smart village, or smart city" [121]. The term "smart home", for example, is usually applied to living spaces equipped with technological devices and software applications that help the resident conquer autonomy and better health [122].

A smart home can be described as a "residence wired with technology features that monitor the well-being and activities of their residents to improve overall quality of life, increase independence and prevent emergencies" [11]. Such spaces may be even better employed for those that have some difficulties while performing day-to-day tasks and need adapted spaces to accommodate their needs, whether they face certain challenges due to aging or having some sort of motor disability.

Technological features can mitigate the absence of a caregiver while also promoting a possible solution from not only a practical, but also economic perspective [14]. For this reason, one alternative to improve the life quality and safety of the senior population is to create "ageing-friendly living environments with the application of smart home technologies" [15]. Furthermore, getting older while using smart home technologies is believed to promote advantages for the user's well-being. Smart home technologies allow the elderly and people with disabilities to receive medical support as needed at their convenience, making it possible for them to live in their own conditions and in an independent manner.

Sensors in smart environments can range from simple binary sensors, pressure sensors, and contact switches to more advanced devices that monitor vital signs, such as heart rate

and blood pressure. They may include devices of various kinds, from motion and temperature sensors to smart televisions and surveillance cameras [119]. These possibilities highlight the critical importance of selecting the appropriate type of sensors for data collection. According to Kebir [123], this is one of the main steps when designing a smart home.

Software applications specifically designed to run in smart environments take advantage of this infrastructure by executing functionalities triggered by the integrated sensors (for instance, offering some local assistance or informing caregiver about a dangerous event [12]), in addition to the direct use of the application. Likewise, the multiple sensors placed in a smart home promote the opportunity for the end user to give an input to software applications even though they are not purposefully interacting with the system. This input is an *implicit interaction*, which is defined as the actions performed by the users when they do not necessarily intend to interact with a computing system, but which the system understands as an input and reacts anyway.

In contrast, an *explicit interaction* refers to the most common ways of interacting with a computer, which is by providing an input on purpose, whether it is by clicking a button, typing on a keyboard, or touching a screen [36]. With its integrated multiple devices, a smart home can support different daily tasks to be performed by their residents. The presence of sensors enables the acquisition of inputs provided in a natural way, e.g., by movement detection, face recognition, etc. This allows users to interact with software applications without being aware that they are engaging in the interacting process. The implicit interactions are even more useful for people that have difficulties when using a mouse, a keyboard, a tactile screen or any sort of computer interaction device.

For this reason, when evaluating the QinU of smart environment applications, one may need to consider both kinds of inputs, explicit and implicit. As a simple example, software designed to regulate the temperature in a smart environment can respond to both implicit demands (triggered by sensor information) and explicit requests (made by a user when

accessing the software application). Therefore, in such places, the user does not need to make explicit efforts to interact with the computer (use of the mouse, keyboard, etc.) but performs day-to-day tasks (moving around, using objects, etc.) that the pervasive system interprets as inputs resulting in a proactive response.

2.5 Conclusion

This chapter presented the definitions of the fundamental concepts that support the approach proposed in this thesis. Our methodology, as described in the upcoming chapters, proposes to model agents to simulate people in *smart environments* through various contexts of use to evaluate the *QinU*, while applying *DM* to ground the simulated events in real data.

Prior to initiating the design of the proposed approach, three different literature reviews were performed and are detailed in the following chapter.

Chapter 3

State of the art

The previous chapter has introduced and discussed the fundamental concepts – QinU, smart environments, ABMS and data mining – that integrate this thesis. This chapter presents literature reviews that focus on and analyze such concepts to observe if they have previously been put together in an implementation that aims at achieving the same goal as this thesis: to evaluate the QinU of software applications found in smart environments by utilizing ABMS and data mining techniques.

3.1 Overview of the performed literature reviews

To carry out the literature reviews, we applied the Rapid Review (RR) method [124]. RR is a type of knowledge synthesis in which Systematic Literature Review (SLR) [125] procedures are accelerated and methods are streamlined to complete the review in a shorter process than in typical SLRs. For example, an RR may limit search sources or have one single researcher select primary studies. The goal is to produce faster results, with lower costs and, therefore, more connected to practice than a traditional SLR, in which the required time frame may not meet the needs of some decision makers [126, 127]. An RR follows the same structure as an SLR, with its methodology having been established as a detailed and

transparent scientific method that is reproducible by other people.

We decided to use the databases Scopus and Web of Science as the sources of studies for all three literature reviews performed, as they group most of the Computer Science references including ACM, Elsevier, IEEE and Springer.

Our first insight for performing a literature review was looking for predictive evaluations of QinU, considering that we were interested in measuring the QinU before any evaluation with end users. Therefore, we set our goal according to the structure presented in Table 3.1.

Table 3.1: Structure to define the research goal for RR 1.

Analyze	QinU prediction
For the purpose of	characterizing
With respect to	agents and data mining techniques
From the point of view of	QA analysts
In the context of	smart environments

Based on this goal, we carried out a first literature review [128] presented in Section 3.2. With the low number of found papers, we decided to reorient the perspective from predicting software quality, and changed the focus to "quality evaluation". Furthermore, we decided to investigate not only the use of agents and data mining, but also how software quality has been evaluated in general. As a result, we defined the research goals that were explored in two other RR [129, 130] that are presented in Sections 3.3 and 3.4, respectively.

Table 3.2: Structure to define the research goals for RR 2 and RR 3.

RR 2		**RR 3**	
Analyze	QinU evaluation	**Analyze**	QinU evaluation
For the purpose of	characterizing	**For the purpose of**	characterizing
With respect to	agent and data mining techniques	**With respect to**	techniques, methods and approaches
From the point of view of	QA analysts	**From the point of view of**	QA analysts
In the context of	smart environments	**In the context of**	smart environments

Considering that our purpose focuses on the application of agents and data mining, we decided that the first two reviews should analyze primary studies. That way, we could identify what had already been done with such techniques. For the third review, we decided to investigate secondary studies, as we would like to have a general identification of all the techniques that had already been applied to evaluate QinU. With the goals defined, specific research questions were established for each RR, as presented in their respective sections.

All search strings were defined using the PICOC strategy [131]: ***P*opulation**, to include terms related to QinU and its characteristics from the ISO/IEC 25010:2011 standard [40][1]; ***I*ntervention**, to focus on prediction (RR 1) or evaluation (RR 2 and RR 3); ***C*omparison** was not included in any of the search strings, as we were not aware of any other reviews with similar objectives; ***O*utcome**, to include terms related to agents and data mining techniques (RR 1 and RR 2), or terms that refer to secondary studies (RR 3); ***C*ontext**, to include terms related to smart environments. Some parts of the search strings from different reviews overlapped, such as the population and context, which were identical across all three reviews, and the outcome, which was the same in the first two reviews (see Table 3.3).

Table 3.3: PICOC search string for the three RR.

	RR 1	**RR 2**	**RR 3**
Population	Quality in use		
Intervention	Prediction	Evaluation	Evaluation
Comparison	*Not applicable*		
Outcome	Agent and data mining techniques		Secondary studies on applied techniques
Context	Smart environments		

The details and analyses of each RR are presented in the following sections.

[1] All literature reviews were carried out before the publication of the latest version of the standard, therefore, the terms applied considered its previous version from 2011.

3.2 Quality in use prediction with agent-based modeling and simulation or data mining

In order to design an approach for our implementation, an RR was carried out to examine how previous studies have predicted software quality [128]. More specifically, how they implemented agents and/or data mining techniques, and in the context of applications found in smart environments. Table 3.4 presents the research protocol for the first RR. We set as *P*opulation "quality in use", its characteristics from the ISO/IEC 25010:2011 standard [40], and "usability", since this last one is usually considered synonymous of quality in use. For *I*ntervention, we included synonyms of prediction. For *O*utcome, we considered all terms related to agents and data mining techniques as seen in [132]. Finally, for *C*ontext, we included commonly used synonyms for smart environments found in the literature.

On November 8th, 2022, we executed the search string in the two search engines while automatically filtering IC1, IC2, IC3, and EC1. We got 684 publications (400 from Scopus and 284 from Web of Science), as presented in Figure 3.1. After excluding duplicated studies (applying EC2, which resulted in 468 studies), we applied IC4, IC5 and IC6 selecting studies by analyzing their titles (which resulted in 20 studies), and after that, their abstracts (which resulted in 7 studies). When applying EC3, we did not have access to five publications. Then, after reading the full publications and applying IC4, IC5 and IC6, it resulted in 2 studies for analyzing and extracting data.

3.2.1 Discussion

Even though both of the analyzed studies are related to a software system, neither had the end goal of evaluating it, and instead used it as means to an end (RQ1). One of the studies [133] aims at applying feature-learning methods to predict comfort and identify the usage context for products. To present their proposed architecture, the authors performed case

Table 3.4: Research protocol for RR 1.

Research questions	RQ1: Are there predictive QinU studies for smart environments? RQ2: Are agents/multi-agent systems used in predictive QinU evaluations? RQ3: Is data mining used in predictive QinU evaluations? RQ4: Which quality characteristics were evaluated? With what measures?
Search string	(“usability” OR “quality in use” OR “effectiveness” OR “efficiency” OR “satisfaction” OR “freedom from risk” OR “context*”) AND (“predictive” OR “prediction” OR “prognostic” OR “prognosticative” OR “anticipating”) AND ((“agent” OR “multiagent”) OR (“mining” OR “knowledge discovery” OR “KDD”)) AND (“smart*” OR “intelligent environment” OR “pervasive” OR “ubiquitous” OR “AAL” OR “ambient intelligence” OR “assisted living” OR “systems of systems” OR “internet of things” OR “Cyber-Physical Systems” OR “Industry 4” OR “fourth industrial revolution” OR “web of things” OR “Internet of Everything” OR “IoT” OR "CPS")
Strategy	Scopus and Web of Science databases
Inclusion criteria	IC1: studies published in English; IC2: studies in the Computer Science and Engineering area; IC3: studies from journals, conferences, or book chapters; IC4: studies concerning predictive quality of smart environment applications; IC5: studies applying agents or data mining; IC6: primary studies only.
Exclusion criteria	EC1: editorials, erratums or books; EC2: studies that were duplicated in the results; EC3: studies not available in our institutions and not obtained from their respective authors.
Extracted data	1. Goal 2. Were there multi-agent systems applied? If so, which were the agent abilities applied? 3. Was there data mining applied? If so, what was the data mining technique used? 4. Which quality characteristics were evaluated? 5. Which software measures took place? 6. Target audience

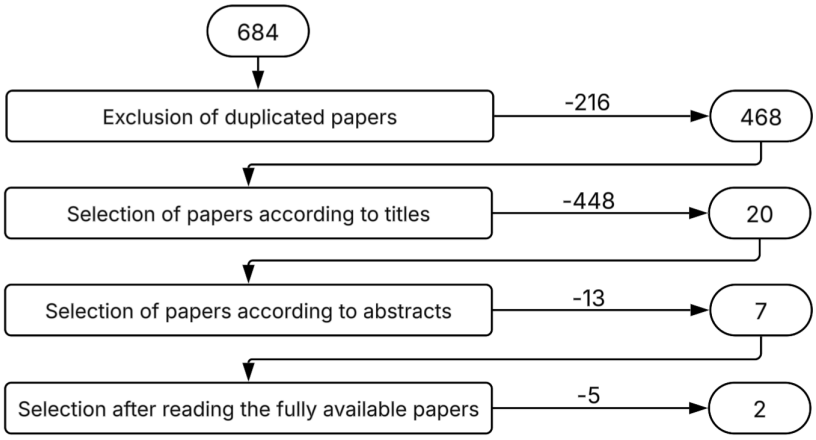


Figure 3.1: Research steps and their respective results.

studies using sensor data taken from a shoe with an integrated sensor. The other study [134] has the goal of predicting equipment component failure and equipment time to failure for both types of equipment involved: trucks and loaders. It proposes a data-driven solution to predict maintenance and improve the overall equipment effectiveness in mines.

Neither one of the studies use agents or multi-agent systems (RQ2), but both use data mining techniques to make predictions (RQ3): the two studies applied machine learning algorithms, one considered decision trees [134] and the other neural networks and support vector machines (SVM) [133].

Regarding quality characteristics (RQ4), one study [134] evaluated efficiency, effectiveness and freedom from risk (safety risk mitigation), with the goal of predicting the devices maintenance and guaranteeing that the tool would be working safely. On the other hand, the other study [133] evaluated satisfaction (more specifically comfort, which is a subcategory of this quality characteristic) and context coverage, focusing on the context completeness part of this quality aspect. In Patil et al.'s study [134], the measures concern the prediction of success and not of quality: precision, recall, and false positive rate considering the events and detected failures. In Ghosh et al.'s study [133], for comfort prediction, a group of people wore a shoe with an integrated sensor while performing different tasks and answered surveys after. None of the studies applied published standards in their studies, such as ISO/IEC 25010 or other.

3.2.2 Threats to validity

Following the classification of threats to validity proposed by Petersen and Gencel [135], the main threat to the validity of this study concerns the theoretical validity, particularly in the selection of studies. Some relevant studies may not have been captured because the search relied exclusively on two databases, which may have limited the scope. Nevertheless, Scopus and Web of Science are widely used for reviews [136]. We had initially planned to apply snowballing procedures to identify additional studies, as is commonly recommended. However, since none of the selected papers addressed our research interest, snowballing was not applied.

The remaining threats to validity are briefly outlined as follows:

- *Descriptive validity* is ensured by using structured Google Forms and Sheets for accurate data recording and management;
- *Generalisability validity* is less critical and more assured for internal generalisability, but external generalisability is limited since the findings may not apply across all contexts;
- *Interpretive validity* is supported by reduced bias through independent checks by expert researchers;
- *Repeatability validity* is supported through detailed documentation of data and analysis protocols.

3.3 Quality in use evaluation with agent-based modeling and simulation or data mining

A second RR was performed centering the search around evaluation methods [129]. Table 3.5 presents the research protocol for this RR. In order to get more studies, we decided

to include in *P*opulation the sub-characteristics of the ISO/IEC 25010:2011 standard [40], except for the ones that belong to "freedom from risk" and "context coverage", because when we executed the query with them included, it did not add any new studies. For *I*ntervention, we included synonyms of evaluation. *O*utcome and *C*ontext were kept the same as the first RR.

On November 17th, 2022, the search string was applied in both databases automatically filtering by the same criteria (IC1, IC2, IC3, and EC1). We got 1,260 publications (762 from Scopus and 498 from Web of Science). After excluding duplicated studies (EC2, resulting in 846 studies), we applied IC4, IC5 and IC6 selecting studies analyzing their titles (resulting in 62), and then by abstracts (resulting in 20). Applying EC3, we got 19 articles. Finally, by reading the full publications and considering the inclusion criteria (IC4, IC5 and IC6), we selected 10 articles for analysis and data extraction. After that, we performed a snowballing procedure on these 10 studies and got 538 publications, 305 from backward snowballing – selecting which studies referenced in the reviewed ones fit our IC and EC – and 233 from the forward snowballing – analyzing the studies that referenced the ones we reviewed.

By applying the IC and EC on these newly found studies, we obtained 5 additional studies for analysis and data extraction. That way, we had a final set of 15 articles (see Figure 3.2).

3.3.1 Discussion

By analyzing more studies than in the first RR, it was possible to extract more data to answer each research question in greater detail.

RQ1. How is the evaluation of QinU performed in smart environments? The majority of the studies has applied a methodology for evaluating the QinU (Figure 3.3) either using questionnaires (seven studies) [137, 32, 138, 139, 33, 140, 141] or simulations (seven studies) [91, 142, 25, 143, 144, 145, 24]. One study [146] presents a review on spoken

Table 3.5: Research protocol for RR 2.

Research questions	RQ1: How is the evaluation of QinU performed in smart environments? RQ2: Are agents/multi-agent systems used for QinU evaluation? RQ3: Is data mining used for QinU evaluation? RQ4: Which quality characteristics were evaluated?
Search string	(“usability” OR “quality in use” OR “effectiveness” OR “efficiency” OR “satisfaction” OR "freedom from risk" OR “context*” OR “usefulness” OR “trust” OR “pleasure” OR “comfort” OR “flexibility”) AND (“evaluation” OR “assessment” OR “quality evaluation” OR "quality assessment”) AND ((“agent” OR “multiagent”) OR (“mining” OR “knowledge discovery” OR “KDD”)) AND (“smart*” OR “intelligent environment” OR “pervasive” OR “ubiquitous” OR “AAL” OR “ambient intelligence” OR “assisted living” OR “systems of systems” OR "internet of things” OR “Cyber-Physical Systems” OR “Industry 4” OR “fourth industrial revolution” OR “web of things” OR “Internet of Everything” OR “IoT” OR “CPS”)
Strategy	Scopus and Web of Science databases combined with snowballing procedures
Inclusion criteria	IC1: studies published in English; IC2: studies in the Computer Science and Engineering area; IC3: studies from journals, conferences or book chapters; IC4: studies concerning quality evaluation of smart environment applications; IC5: studies applying agents or data mining; IC6: primary studies only.
Exclusion criteria	EC1: editorials, erratums or books; EC2: studies that were duplicated in the results; EC3: studies not available in our institutions and not obtained from authors.
Extracted data	1. Goal and evaluation procedures 2. Were there multi-agent systems applied? If so, which were the agent abilities applied? 3. Was there data mining applied? If so, what was the data mining technique used? 4. Which quality characteristics were evaluated? 5. Which software measures took place? 6. Target audience

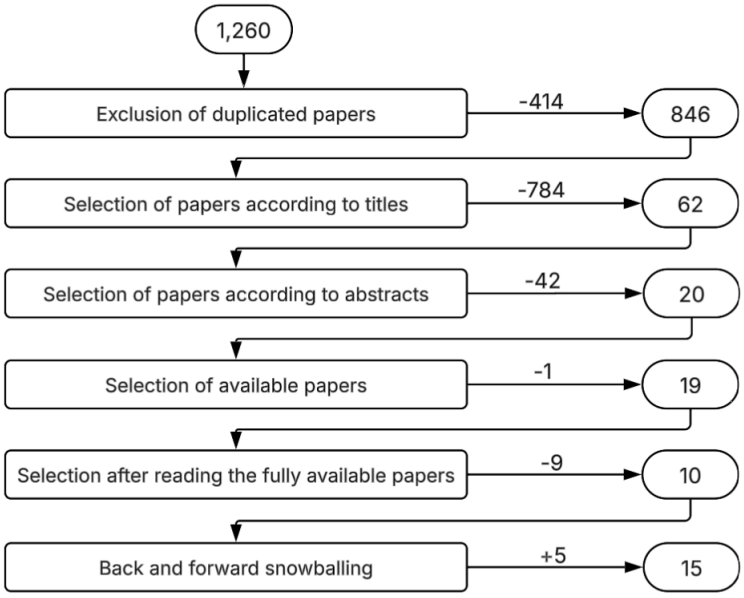


Figure 3.2: Execution overview of the performed search.

language systems.

From the seven studies that use questionnaires, two involved professionals (experts in the domain) of the area that was being evaluated. The first one [138] promoted a survey with 17 experts in the domain of integrating low-level automation functions and industrial agents, as their goal was to perceive which of the characteristics and sub-characteristics of ISO/IEC 25010 (which is the standard we will apply in our approach) are considered important for this niche area. The second study [33] presented a video narrating a use case for the participants and, after that, asked them to answer an open-ended questionnaire to collect their opinion.

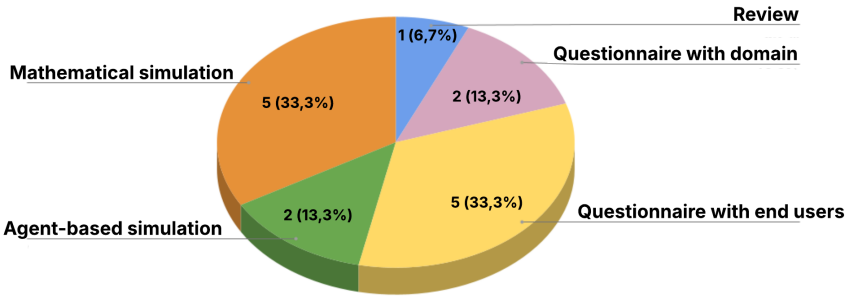


Figure 3.3: Methodologies used for QinU evaluation.

The other five studies involved end users in situations where they used the application that was being evaluated and then answered questionnaires. Two of these studies [137, 140] applied standard usability questionnaires (that is, with a pre-established pattern). Among these, O'Grady et al. [137] evaluated two different applications with embedded agents, and for that, they used the SUMI (Software Usability Measurement Inventory) [147] because it is widely recognized for measuring software quality from an end-user's perspective. The end users answered the questionnaire after using the applications themselves. Möller et al. [140] also applied SUMI as one of their evaluation methods. In this study, the authors presented approaches for evaluating AmI applying four different types of questionnaire (SUMI [147], AttrakDiff [148], System Usability Scale [149], and SASSI [150]) to be responded by end users after using three different software systems, in a randomized order.

The other three studies used questionnaires that were developed by the authors themselves. In [32], the evaluators used UXAmI Observer, an automated user experience evaluation tool for AmI environments. They rated the system on a scale from 1 to 5 in questions developed by the authors, and the measures were calculated from there, considering an average value for the evaluation result, in a similar way to [138]. Another study [139] presents an evaluation of a mobile user in the context of a smart city. For that, a preliminary survey was promoted to understand what were the users' expectations in relation to systems like the one that was going to be evaluated. After that, the end users were asked to use the evaluated system, and then to answer a more in-depth post-use survey to capture their opinion regarding functionality and utility, usability and performance, and emotional impact. In the last study [141], the end users were put in situations where they had to interact with the robots that were being evaluated. Such robots could act proactively or on demand. After that, they had to fill out a questionnaire so that their point of view could be taken into consideration for both positive and negative aspects. This is the only study in which the end users were elderly people.

Regarding the seven studies that performed some type of simulation, two of them [91,

144] presented agent-based simulations. Carbo et al. [144] created an agent-based simulation with NetLogo for an approach that can evaluate AmI, where the agents represented different people performing various tasks at an airport. Veronese et al. [91] presented a simulator capable of reproducing the behaviour of a person on their daily life in terms of Activities of Daily Living (ADLs). This activity scheduling can manage to create synthetic data from sensors deployed in a completely virtual environment. To evaluate how accurate that simulation made by agents was, the authors compared the data generated to data that was obtained through a real world data set.

The other five studies performed other types of simulations, where they defined values and applied them to mathematical formulas to validate their results. Two of them [145, 24] had the goal of assessing trust, but even though they wished to assess the same environment and to measure the same quality sub-characteristic (trust), they did not follow the definition of trust as seen in the ISO/IEC 25010 [41], therefore they might not consider the exact same aspects. Cioroaica et al. [24] presented the goal of creating a predictive simulation within the process of building trust. To achieve that, the authors executed digital twins in a simulated environment to evaluate the behavior of software smart agents, because it created safe conditions for detecting malicious behavior without having to encounter any dangerous situation that could have been met in the real world system execution. Lin et al. [145] aimed at either clarifying or augmenting the concept of trust and promoted a simulation with information from a data set of users from three social networks (Facebook, Google+ and Twitter) that granted the data. Mathematical calculations took place to validate such proposal.

The other three studies applied simulation for different goals. Du et al. [25] proposed a novel textual data-driven evaluation method after trying to measure the user satisfaction of smart and connected products. The proposal was created after defining a new frequency-oriented formula and a new position score-based formula. After that, an integrated approach was presented to obtain such attributes combined. Li et al. [143] presented a new

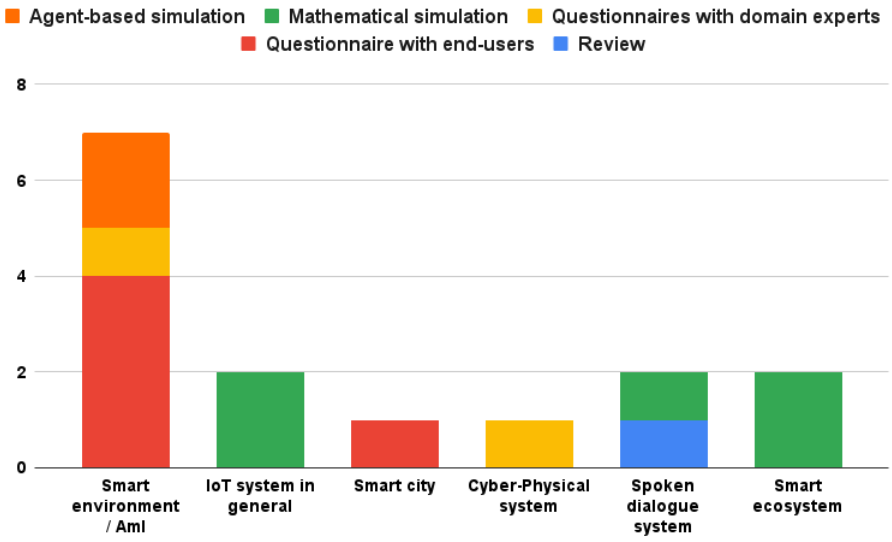


Figure 3.4: Evaluation methodologies used in different kind of system.

reliability evaluation approach of integrated energy systems based on smart agent communication. For that, a communication algorithm was integrated into a sequential Monte Carlo simulation, which was conducted on AnyLogic platform. Walker et al. [142] created a framework for evaluating spoken dialogue agents named PARADISE (PARAdigm for Dialogue System Evaluation). It provides support to compare agents performing different tasks, besides enabling the performance calculation over subdialogues and whole dialogues. To evaluate the performance, defined measures were calculated through predefined mathematical models. The last study [146] performed a literature review to analyze different methods of software evaluation specifically for spoken language dialogue systems.

Finally, analyzing the methodologies used in each one of the 15 studies regarding the kind of system they address (Figure 3.4), we note that most of the studies that evaluated smart environments/AmI applications asked participants to fill out questionnaires and calculated the mean response.

RQ2. Are agent/multi-agent systems used for QinU evaluation? Fourteen (93,33%) of the fifteen studies have involved agents somehow in their work. The agent concept is generally used to represent any kind of autonomous actor in the system, artifi-

cial (software or robot) actors as well as human actors. The use of agents can remain at this conceptual level, only describing AI agents or "smart agents" as black boxes [24], or parts of an adaptable architecture, like in [139]. When the study describes the artificial agents' behavior and/or a multi-agent architecture, the agent ability that is mainly presented is communication (see Figure 3.5), before learning and organization-related abilities. In addition, some of the examined studies make use of agents linked with simulation, without necessarily referring to agent-based simulation.

The ability to communicate is mentioned in many studies involving agents, generally with the aim of supporting the interaction with the users. For instance, in the automated UX evaluation tool for AmI in [32], artificial agents are employed to perceive the information required from the AmI environment and then translate it into the format defined by the AmI Reporter API. Communication is also the main ability used by the embodied agent in [141], controlling an assistive robot, with the role of cognitive support for elderly people, using verbal interaction to get the user's attention. The study [140] also proposed an embodied agent, but under a virtual form (a talking head), to assist the user. In [142], the communication between users and agents is at the core of the project: the aim is to provide a framework to evaluate and compare the efficiency of different spoken dialogue strategies, based on a representation of the agent's task characteristics. Another example where agents' communication ability is central is provided by the study in [145], where autonomous connected objects are presented as interacting agents, which support information and services providing to users. Finally, communication can also concern the interaction between agents and other systems, as in [138], focused on the interface between agents and the physical industrial automation systems.

Integrating both communication and organization-related abilities, the multi-agent system in [143] is able to reconfigure itself to adapt the communication network typology, with the intent to improve the reliability of energy systems. Various abilities are studied in [33], which classifies in distinct levels the functionalities that would be provided by a lighting

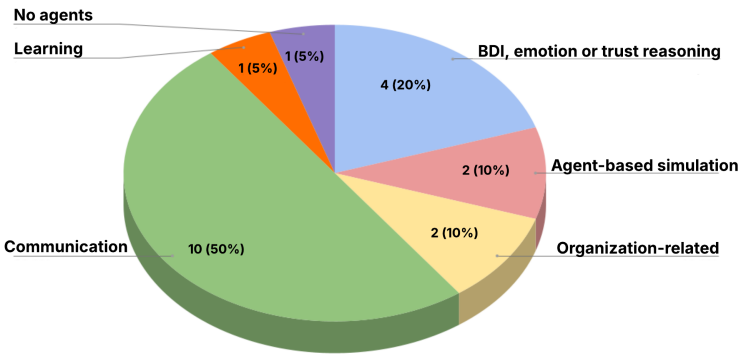


Figure 3.5: Agents abilities applied in the studies.

AmI agent, from a simple on-demand actor to an emotionally-aware and learning assistant. Reasoning about beliefs, desires and/or intentions (BDI-like models), emotions or trust is used in some studies [32, 137, 144, 145].

Two studies use agent-based simulation, close to our goal of simulating human interaction in a smart environment with the help of agents. In the first study [144], an agent-based simulation using NetLogo emulates human actions performed in an airport while going by different zones (check-in, security, etc.). Hence, the agents are not responsible for the evaluating part itself, but the authors concluded that this simulation can allow to evaluate the relevance of different AmI facilities. The second study [91] proposed two alternatives to create home automation data: an agent-based model (with deterministic behavioural pattern descriptions) and a stochastic one. The first alternative applied agent abilities to reproduce human behaviour performing daily life activities. To evaluate the accuracy of the simulation performed, the data generated by the agents was compared to the data from actual humans. Therefore, it did not use agents exactly to evaluate the system but to generate information similar to the one human activities can produce.

None of these studies use agents to evaluate the software or the system: the evaluation is performed by experts [32], surveys [138] or questionnaires [141]. However, the agents can be the subject of the evaluation. Two different agent-based applications (CRUMPET and Gulliver's Genie) are evaluated in [137] to assess their usability. The analysis highlights some observations about the agent design, but no agent ability was involved in the

evaluation part that used the questionnaire-based SUMI methodology. In [24], the authors create a predictive simulation within the process of building trust, therefore they run digital twins in a simulated environment to evaluate the behavior of software smart agents. The agents were not employed for evaluating the environment itself though.

RQ3. Is data mining used for QinU evaluation? Only one of the analyzed studies (6,66%) employed data mining techniques for evaluation [25]. In this study, a proposal was created for measuring the users' satisfaction for SCPs (Smart Connected Products) according to online commentaries made about them. For that, a systematic framework was proposed applying data collection and preprocessing using Python to mine customer reviews from websites. This study involves PLTSs (probabilistic linguistic term sets) to mine and quantify such textual reviews.

RQ4. Which quality characteristics were evaluated? All QinU characteristics considered in our review were found in the studies as shown in Figure 3.6. The QinU characteristic most measured was satisfaction (8 of the 15 studies, 53,33%), followed by efficiency (6 studies, 40%), and effectiveness and usability (5 studies each, 33,33%). We note that trust was evaluated considering different perspectives. Some of the reviewed studies [25, 138] evaluate trust as defined by ISO/IEC 25010 (even though the first does not mention the standard explicitly), that is, as a satisfaction feeling the user gets from trusting the application, while some other studies [24, 145] consider trust as an assurance that the feature agents and techniques will not fail their tasks. We also note that the studies evaluating freedom from risk characteristic were focused on reputation [141, 143] (a criteria of economic risk in freedom from risk) by evaluating the respect, admiration or how liked the system is by the end user.

Most of the reviewed publications follow some measurement pattern to establish a result for their evaluation [25, 32, 91, 138, 139, 141, 142, 143, 145], even though they do not apply measures from any standard. However, there are studies that do not specify any measurement for their evaluation [24, 33, 146].

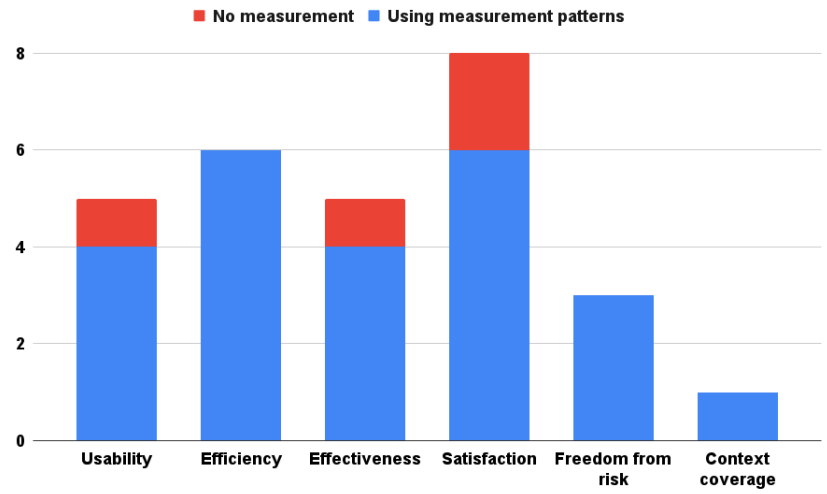


Figure 3.6: Quality in use characteristic considered in the evaluations.

Figure 3.7 shows which QinU characteristics have been exploited in the different types of systems analyzed. We noticed that usability, effectiveness and efficiency were each evaluated in three of the seven studies regarding smart environments/AmI, followed by satisfaction (two). This is due to the fact that most of them use standard usability questionnaires. Moreover, satisfaction was evaluated in all kinds of applications.

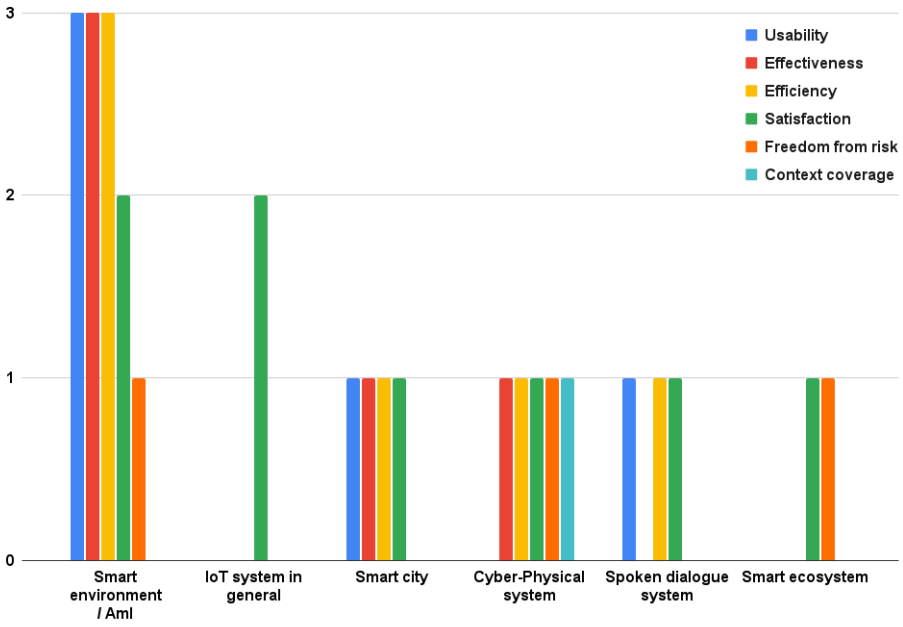


Figure 3.7: Quality in use characteristic evaluated in each kind of application.

3.3.2 Threats to validity

The analysis of threats to validity in this study follows the framework proposed by Petersen & Gencel [135], in line with the previous study:

- *Descriptive validity* is about the data collection regarding recording, storing, and reviewing. To mitigate this threat, a Google Form was created for extracting data according to the established RQs, and that information was then accessed in a spreadsheet on Google Sheets;
- *Theoretical validity* concerns (i) the study identification/sampling and (ii) the data extraction. For (i), we are aware that some studies may have been left out because we performed the search only on two databases, which may promote incomplete results. However, studies have shown that combining Scopus and Web of Science with snowballing procedures can mitigate the lack of search engines and provide a representative set of studies to a characterization work [151, 136]. For (ii), we followed the recommendations of Kitchenham [152] to mitigate the risk of bias interpretation, by having one researcher to extract the data and another to review the extraction;
- *Generalizability validity* is related to the degree to which we could generalize the obtained results internally (within one group or one company) and externally (between multiple groups or different organizations). As an RR methodology, internal generalizability is not a major threat. Despite that, we do not guarantee external generalizability, as the "awareness of interpretation" in the data analysis is important in this context;
- *Interpretive validity* regards whether the reached conclusions from the analyzed data are reasonable and not biased by the researchers during the analysis. To mitigate this threat, both the selection and the extraction have been double checked by different researchers with significant expertise across the domains involved;
- *Repeatability validity* requires that data and analysis methods are defined and repro-

ducible, as done in the protocol documented in Table 3.5.

3.4 Quality in use evaluation of smart environment applications

The third RR aimed at investigating which approaches have been applied the most for quality evaluation in smart environment applications. As previously mentioned, a tertiary study [153] by RR was carried out [130]. *P*opulation, *I*ntervention and *C*ontext were kept the same as RR2. For *O*utcome, we included synonyms of "secondary study" without precising the key terms for the tools and techniques applied to not restrict the string, risking to miss some studies[2]. Table 3.6 presents the planning for the third RR performed.

On July 19th, 2023, the search string was executed across both selected databases, and thanks to their engine tools, the outcome was filtered applying IC1, IC2, IC3 and EC1. This resulted in 338 studies from Scopus and 861 from Web of Science (Figure 3.8). Combining the results from both databases (1,199 papers), we applied EC2 to exclude duplicated studies before the manual review, which resulted in a total of 1,028 studies. Then, all the titles were reviewed to include only secondary studies that analyzed QinU assessment of smart environment applications (IC4 and IC5), which resulted in 100 studies. Then, all the abstracts were analyzed with the same purpose, which resulted in 35 studies. Two studies were not available (EC3), which resulted in a total of 33. After reading the full publications, we concluded that 12 studies concerned mobile applications and were not effectively applied in smart environments (IC4), which was not clear from the abstract, so they were also excluded. Finally, 21 studies remained for data extraction.

[2]The detailed process and replication packages are available on [154, 155].

Table 3.6: Research protocol for RR 3.

Research questions	RQ1: What are the most common evaluation approaches for QinU in smart environments? RQ2: What are the most evaluated types of systems regarding smart environments? RQ3: Which quality characteristics were the most evaluated in smart environments?
Search string	(“usability” OR “quality in use” OR “effectiveness” OR “efficiency” OR “satisfaction” OR “freedom from risk” OR “context*” OR “usefulness” OR “trust” OR “pleasure” OR “comfort” OR “flexibility”) AND (“evaluation” OR “assessment” OR “quality evaluation” OR “quality assessment”) AND (“*systematic literature review” OR “systematic* review*” OR “mapping study” OR “systematic mapping” OR “structured review” OR “secondary study” OR “literature survey” OR “review of survey*” OR “state of research” OR “state of art” OR “rapid review” OR “SLR” OR “scoping review”) AND (“smart*” OR “intelligent environment” OR “pervasive” OR “ubiquitous” OR “AAL” OR “ambient intelligence” OR “assisted living” OR “systems of systems” OR “internet of things” OR “Cyber-Physical Systems” OR “Industry 4” OR “fourth industrial revolution” OR “web of things” OR “Internet of Everything” OR “IoT” OR “CPS”)
Strategy	Scopus and Web of Science databases
Inclusion criteria	IC1: studies published in English; IC2: studies in the Computer Science and Engineering area; IC3: studies from journals, conferences, or book chapters; IC4: studies concerning quality evaluation of smart environment applications; IC5: secondary studies only.
Exclusion criteria	EC1: editorials, erratums or books; EC2: studies that were duplicated in the results; EC3: studies not available in our institutions and not obtained from their respective authors.
Extracted data	1. Goal 2. Number of papers reviewed 3. Approaches used for evaluation 4. Type of the system that was evaluated 5. Quality characteristics that were evaluated 6. Type of secondary study 7. Type of publication 8. Year of publication

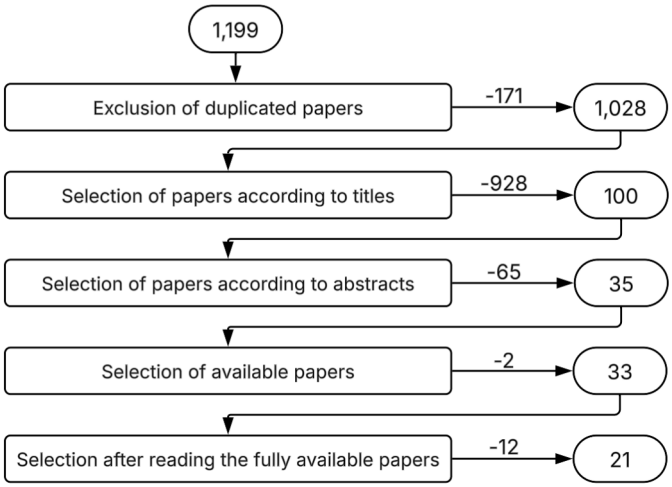


Figure 3.8: Research steps and their results.

3.4.1 Discussion

We did not establish any date limits for the results. We noticed that the found studies are recent, ranging from 2018 to 2023 (see Figure 3.9), with at least two being published each year. The peak was in 2021 with seven publications found on the matter, which shows it is a relatively new field of research, and also that this is a topic which has been actively studied.

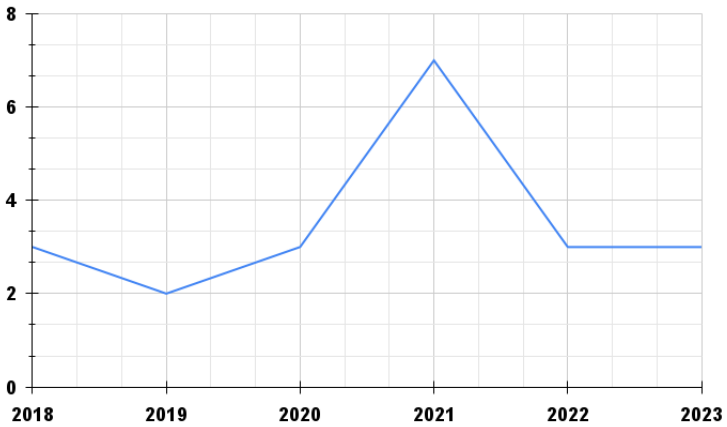


Figure 3.9: Number of studies published per year.

The studies selected for our review can be categorized based on the methodology they applied to choose the studies for their analyses. In total, there were three different cate-

gories of secondary studies: *Systematic Literature Review* [27, 28, 30, 120, 156, 157, 158, 159, 160], *Scoping Review* [26, 29, 31, 161, 162, 163, 164, 165, 166], and *Systematic Mapping* [167, 168, 169].

Considering all the secondary studies we analyzed, seven out of 21 (33,33%) were published in international conferences and 14 (66,66%) were published in journals. Not all the secondary studies listed the papers (primary studies) they analyzed.

RQ1. What are the most common evaluation approaches for QinU? Different types of evaluation were found by the secondary studies (Table 3.7), with the most common evaluation approach being the questionnaire, whether the authors created it themselves for their study [120, 27, 28, 29, 169, 160, 157, 162, 161, 167, 166, 156] or applied an already published questionnaire [26, 28, 29, 160, 157, 158, 168, 161, 167, 165, 166], followed by interviews [26, 27, 28, 29, 169, 160, 157, 162, 164, 161, 167, 165, 166, 156], focus groups [26, 27, 28, 29, 160, 162, 167, 166], observation [26, 27, 28, 30, 161, 167, 166, 156], log data [163, 26, 160, 167, 166], the think aloud protocol [28, 29, 161, 167], surveys [30, 31, 166], user feedback [28, 162], heuristic evaluations [28, 29], and others. Table 3.7 presents the reference for each secondary study, the number of primary studies they analyzed (#), and the methods and techniques identified (e.g., [156] analyzed 24 primary studies).

More than half of the secondary studies analyzed (12) reported the use of questionnaires that were created by the authors with the sole purpose of answering their research questions and without any plan for repetition [27, 28, 29, 120, 160, 157, 162, 161, 167, 166, 156, 169], while other authors used established questionnaire forms to reach their goals. Between those, the secondary studies indicated a total of 44 different questionnaires, including well-known and standardized ones, according to [34] and [158], that are presented in Table 3.8. This table lists the standardized questionnaires, indicating how many secondary studies identified them and how many primary studies applied each questionnaire within those secondary studies. An *X* is used when the secondary studies did not specify the number of primary studies that applied the questionnaire in question.

Table 3.7: Methods and techniques found in the primary studies.

Ref.	#	Ad-hoc quest.	Estab. quest.	Interview	Focus group	Observation	Log data	Think aloud	Survey	User feedback	Simulation	Heuristics	Automated test	Real testbed	Prototype	Exp. protocol	Others
[26]	111	-	9	4	2	1	101	-	-	-	-	-	-	-	-	-	-
[156]	24	13	-	2	-	7	-	-	-	-	-	-	-	-	-	-	-
[157]	44	16	19	13	-	-	-	-	-	-	-	-	-	-	-	-	-
[161]	35	12	26	6	-	4	-	2	-	-	-	-	-	-	-	-	-
[162]	21	8	-	8	3	-	-	-	-	2	-	-	-	-	-	-	1
[163]	63	-	-	-	-	-	7	-	-	-	-	-	-	-	-	-	56
[158]	553	-	553	-	-	-	-	-	-	-	-	-	-	-	-	-	-
[120]	21	21	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
[167]	65	29	41	8	1	2	17	8	-	-	-	-	-	-	-	-	-
[30]	15	-	-	-	-	10	-	-	9	-	-	-	-	-	-	-	-
[159]	146	-	-	-	-	-	-	-	-	-	76	-	-	37	11	-	22
[27]	29	1	-	6	4	1	-	-	-	-	-	-	-	-	-	-	17
[28]	22	2	2	2	2	1	-	1	-	2	-	11	5	-	-	-	-
[29]	133	39	66	37	13	-	-	45	-	-	18	-	-	-	-	-	57
[31]	9	-	-	-	-	-	-	-	6	-	-	-	-	-	-	3	1
[160]	34	2	2	1	1	-	9	-	-	-	-	-	-	-	-	19	-
[164]	8	-	-	4	-	-	-	-	-	-	-	-	-	-	-	-	4
[165]	51	-	29	24	-	-	-	-	-	-	-	-	-	-	-	-	-
[168]	12	-	10	-	-	-	-	-	-	-	-	-	-	-	-	2	-
[169]	10	7	-	5	-	-	-	-	-	-	-	-	-	-	-	-	-
[166]	31	14	17	16	2	4	3	-	3	-	-	-	-	-	-	-	-

Table 3.8: Standardized questionnaires found in the secondary studies.

Questionnaire	**No. of sec. studies**	**No. of primary studies per sec. studies**
System Usability Scale (SUS)	9	1 [156], 16 [157], 16 [161], 1 [120], 11 [167], 2 [28], *X* [165], 44 [29], 3 [166]
System Usability Scale (SUS) adapted	3	2 [161], 1 [167], 1 [166]
Technology Acceptance Model (TAM)	4	4 [161], 5 [29], *X* [165], 2 [169]
Post-Study System Usability Questionnaire (PSSUQ)	3	1 [157], 1 [161], 12 [29]
Attrakkdiff	2	341 [158], 2 [167]
Modular evaluation of key Components of User Experience (meCUE)	1	12 [158]
Usefulness, Satisfaction and Ease of use questionnaire (USE)	2	1 [157], 1 [167]
User Experience Questionnaire (UEQ)	2	200 [158], 1 [167]
Computer System Usability Questionnaire (CSUQ)	3	1 [156], 2 [157], 1 [161]
Software Usability Measurement Inventory (SUMI)	1	1 [166]
Usability Metric for User Experience (UMUX)	1	1 [167]
Unified Theory of Acceptance and Use of Technology (UTAUT)	1	2 [161]

The most common used questionnaire was System Usability Scale (SUS), which was found by 9 secondary studies applied in its original form [156, 157, 161, 120, 167, 29, 165, 166, 28] and was also in an adapted way in three secondary studies [167, 166, 161]: one of them adapted the questionnaire according to the context [167] and the others [166, 161] customized it by putting it together with other established questionnaires. Between the latter, one of them [166] found it adapted with the NASA Task Load Index (NASA-TLX), while another secondary study [161] spotted SUS adapted in two different primary studies: one combined it with PSSUQ (Post-Study System Usability Questionnaire) and another combined it with CSUQ (Computer System Usability Questionnaire). Three different secondary studies [156, 167, 165] also found the application of the NASA-TLX in its original form, being the non-standardized questionnaire to be applied the most.

One study [158] focused particularly on three different standardized questionnaires: AttrakDiff, UEQ (User Experience Questionnaire) and meCUE (Modular Evaluation of key Components of User Experience), with the purpose of determining how they were applied in the past for AmI and ubiquitous computing. Apart from the standardized questionnaires,

one published non-standardized questionnaire was found applied by primary studies three times [29], seven were found applied twice in the primary studies [156, 167, 169, 166], and eighteen were found applied only once in the primary studies [156, 161, 167, 166, 26].

Plenty of secondary studies encountered methods and techniques that involved the researchers having direct contact with end user for evaluating the QinU. Some of them performed direct interviews to collect the information they wanted to report on, asking predefined questions regarding the software they wanted to evaluate [26, 27, 28, 29, 169, 160, 157, 162, 164, 161, 167, 165, 166, 156], while others took into consideration people's opinion through user feedback [162, 28], listening to the users' comments and thoughts after they interacted with the applications in a free manner, and some applied the think aloud protocol [28, 29, 161, 167], where the people involved use the software that is being evaluated while describing, out loud, their actions (and expectations regarding what will come from it).

A number of secondary studies found authors that used focus groups to evaluate the software quality [26, 27, 28, 29, 160, 162, 167, 166], while some also indicated the observation method [26, 27, 28, 30, 161, 167, 166, 156].

Four secondary studies found the use of log data in primary studies [26, 160, 167, 166], and in such cases, they were used for calculating metrics, even though they were evaluating the QinU in different systems. Two of these studies focused on the assessment of the QinU of smart environments for the elderly [166, 160], one [26] aimed to evaluate QinU characteristics in wearable devices, such as smartwatches, and another [167] had the goal of analyzing multi-touch systems in general. In all these cases, the log data came from the interaction of the end users themselves with the application that was being evaluated.

We noticed that only one study [159] pointed out the use of simulations applied in different platforms for different contexts of use. Most of them provided mathematical simulations, simulating "energy cost, hardware frequency rate, and computation time" of the scheme to be validated, but some other examples were of simulating "a resource description framework for heterogeneous IoT devices". There was no mention of any primary studies

performing agent-based simulations for replacing human-computer interaction.

Some secondary studies pointed out evaluation methods and techniques based on different testing performances, which included real testbeds and prototypes [159], automated tests [28], and experimental protocols [160, 31, 168].

Finally, we defined a category "others" in Table 3.7 to represent some specific findings. First, to represent some methods and techniques for evaluation found only once in a secondary study: card sort [162], which is a method to help the researcher perceive how others categorize information (guaranteeing a data architecture that matches the users' expectations), and randomized controlled trials [164], which are techniques that balance the characteristics of the participants between the trial groups to allow differences in the intervention results. Similar to this last technique, four studies are identified in [31] as applying randomized clinical trials. Another study [159] found the use of formal techniques, employed to model complex systems as mathematical entities, and used what the authors called "design techniques" for evaluation, in which the primary studies aim to provide new approaches or frameworks. Task completion was also quoted [29] as another method for evaluation that considers whether or not the users were able to perform (with adequate standards) certain tasks within a defined period of time. Finally, several other approaches were quoted in the studies, not really precising which technique or method was applied [159, 163, 27]. For instance, [163] mentioned that the identified approaches in the primary studies worked on visual data anonymization methods to try "to retain all the informative richness of visual data acquired with RGB cameras" and on security based issues for user authentication and data encryption. Another one [27] mentioned the use of experimentation field tests in general to collect data.

It is noticeable that for most of the secondary studies there was a higher number of methods and techniques than for primary studies, which is caused by many of them combining different ways of evaluating the QinU of the application under analysis. One example is when a secondary study [28] pointed out that two primary studies applied focus groups

and interviews together.

We cannot quantify all the information from the primary studies found in the secondary ones, as some of them did not provide all the references and it is likely that there is an overlap of information. Besides, even though some secondary studies went into detail about their findings, some of them did not disclose specific numbers for the data taken from primary studies (for example, how many of them applied the same method or technique for assessing software quality or how many measured a specific QinU characteristic).

RQ2: What are the most evaluated types of systems? The most evaluated type of systems was smart environments/AmI [157, 161, 162, 163, 158, 30, 160, 164, 169, 120], with ten studies (47,61%), as seen in Fig. 3.10, followed by IoT [159, 27, 31, 168, 165, 26, 166] with seven studies (33,33%). The other software categories were evaluated by only one study each, those being cyber-physical systems [156], multi-touch systems [167], eGovernment [28], and eHealth [29].

Most of the secondary studies about smart environments/AmI focused on AAL [157, 161, 162, 163, 158, 160, 164, 120]. However, one study investigated specifically ubiquitous healthcare [169], while another looked into smart learning environments [30]. Among the seven studies that included QinU evaluations for IoT, two of them dealt with navigation apps: one of them [27] involved mobile safety alarms with GPS, RFID tags and readers, product design assessment for safe navigation and more; the other study [165] focused on indoor navigation apps for people with mobility disabilities. Three other studies about IoT investigated wearable devices (such as smartwatches, wristbands or neckwear) for different reasons: tracing people and objects regarding hospital-like scenarios involving medical teams and patients [159], improving medication adherence [31], or analyzing physical activities [26]. One study focused on assistive technology for older adults [166] and another on IoT in general [168].

RQ3: Which quality characteristics were the most evaluated? All QinU characteristics, as well as usability, were found to be evaluated in the secondary studies, as presented

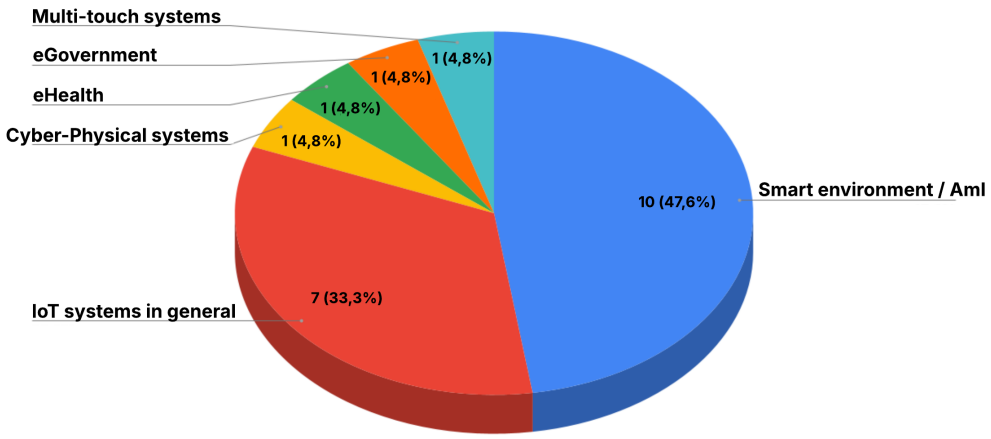


Figure 3.10: Types of systems evaluated in the secondary studies.

in Table 3.9. Similar to Table 3.8, an *X* is displayed for secondary studies that did not specify the number of primary studies that measured said quality characteristic.

It is usual for a study to evaluate more than one QinU characteristic at once. Usability was evaluated the most, being found in sixteen secondary studies (76,19%) [163, 26, 27, 28, 29, 120, 169, 157, 162, 158, 30, 168, 161, 167, 165, 156]. This result is expected, considering that usability is the focus of questionnaires, which were the most commonly used method for evaluation. Effectiveness was the second most mentioned, with nine studies [26, 120, 160, 164, 168, 167, 31, 166, 156], followed by satisfaction, with eight studies [163, 26, 120, 30, 168, 167, 31, 156]. Then, tied with six studies each, efficiency [120, 159, 168, 167, 31, 156] and freedom from risk [163, 120, 159, 168, 31, 156]. Lastly, context coverage was found to be assessed by only one of the secondary studies [120].

It is noticeable that many studies apply questionnaires for evaluating usability (and, as a consequence, effectiveness, efficiency and satisfaction as defined by ISO 9241-11[23]) since the Human-Computer Interaction domain has been investigating usability issues for a long time.

Most of the studies aimed at evaluating the main QinU characteristics, but some had the goal of evaluating ones that were considered as sub-characteristics, according to the ISO/IEC 25010 standard [41], such as usefulness, acceptability [30, 166, 163], which fall un-

Table 3.9: QinU characteristics found in the primary studies (#).

Ref.	#	Usability	Effectiveness	Efficiency	Satisfaction	Freedom from risk	Context coverage
[156]	24	12	5	3	3	1	-
[157]	44	44	-	-	-	-	-
[161]	35	35	-	-	-	-	-
[162]	21	21	-	-	-	-	-
[163]	63	40	-	-	6	17	-
[158]	553	553	-	-	-	-	-
[120]	21	15	17	17	17	14	13
[167]	65	65	13	11	2	-	-
[30]	15	7	-	-	3	-	-
[159]	146	-	-	*X*	-	*X*	-
[27]	29	29	-	-	-	-	-
[28]	22	22	-	-	-	-	-
[29]	133	133	-	-	-	-	-
[31]	9	-	6	2	1	3	-
[160]	34	-	34	-	-	-	-
[26]	111	16	107	-	20	-	-
[164]	8	-	8	-	-	-	-
[165]	51	51	-	-	-	-	-
[168]	12	4	2	2	2	5	-
[169]	10	10	-	-	-	-	-
[166]	31	-	31	-	-	-	-

der the satisfaction umbrella, and reliability [168], safety [163], security [163], which are considered a sub-characteristic of freedom from risk. It is important to follow a pattern to guarantee that studies have the goal of evaluating the same characteristics, as "it is evident that the concepts of UX and usability are not the same among the authors" [167]. However, few of the secondary studies extracted information about alignment with quality standards. One of them [120] pointed out that five primary studies followed ISO/IEC 25010 [41], four followed ISO/IEC 9126 [170], two followed ISO/IEC 25012 [171], and lastly, one followed ISO/IEC 14598 [172]. Another [168] classified the selected primary studies according to the characteristics and sub-characteristics they evaluated, splitting them between system/software product quality and QinU, according to ISO/IEC 25000 [173]. Moreover, none of the analyzed studies followed a standard such as ISO/IEC 25022 [22], which brings measures for the said characteristics to be assessed. One of the secondary studies [28] suggested that future researches should take ISO/IEC 25010:2011 [40] into consideration, as it defines usability in a context of use.

3.4.2 Threats to validity

The analysis of threats to validity in this study is based on Petersen & Gencel [135] and largely mirrors the threats identified in the previous review (Section 3.3.2). One exception concerns *theoretical validity*: while the searches were also conducted in Scopus and Web of Science (databases widely employed in reviews [136]), no snowballing procedures were performed in this study. This decision was motivated by our focus on identifying secondary studies that analyze how the QinU of software applications has been evaluated. More broadly, our aim was to gain an overall understanding of how software application evaluations have been carried out, seeking a general view of the methods employed.

3.5 Conclusion

We can summarize our findings obtained in the literature reviews as follows:

- from RR 1, we noticed that the term "prediction" is not usually applied to refer to quality assessments. This conclusion made us adapt and refine our search, changing the perspective in the reviews that followed;
- from RR 2, it was possible to notice that (i) all QinU characteristics have been applied to evaluate smart environments applications, even though not always by employing established quality measures; (ii) agent-based simulation can be used to emulate human interactions; and (iii) none of the analyzed studies applied agents and the data mining for the evaluation of the quality of the system;
- from RR 3, we point out that most of the assessment approaches to evaluate the QinU of smart environment applications use classic methods that require that potential users interact with the evaluated application to provide their personal opinion after. That means the user should have a live experience in these environments before answering questionnaires or attending interviews to offer their feedback. We argue

that carrying out these experiences with elderly or people with disabilities may not be adequate.

Those three literature reviews presented show that an automated approach of QinU evaluation providing supporting material for QA analysts has not been done yet. Moreover, they also indicate that the use of software agents and data mining was not applied for means of quality evaluation. On top of the results of these reviews, we propose, in the next chapter of this thesis, a new approach to address and design a solution for the found problem.

Chapter 4

QUESTS: a methodology for quality in use evaluations with agent-based modeling and simulation

As indicated in the previous chapter, there is a gap in the literature regarding procedures for evaluating the software quality of applications found in smart environments without having to ask potential end users to participate in such assessment.

Smart environment applications are dependent on the integration of multiple sensors, which work together to enable context-aware interactions [123]. These applications are shaped not only by the types of sensors in place but also by how they are combined and deployed within specific environments. In this sense, we argue that it is possible to simulate such environments by systematically varying the elements that define the context of use. Through simulation, a wide range of scenarios can be generated, reflecting the variability that occurs in real-world smart environments while enabling controlled and repeatable experimentation for an initial evaluation of the QinU.

This chapter begins by laying out the foundational concepts upon which our approach

is built. It then introduces a proposal for supporting QinU evaluations using software agents. The proposal outlines its key components aimed at addressing the problem, including ABMS, PM, and the assessment of QinU measures.

4.1 Fundamentals of the approach

As previously announced in Section 1.2, the core of QUESTS is the application of simulation to reproduce the use of the software application under evaluation in multiple contexts of use. To deal with such variability, it is necessary to analyze the four elements that compose these contexts: users, resources, tasks, and environment (see Section 2.1.3). Different contexts of use are created by different combinations of these four elements, which could be different users in the same environment, the same user in different environments or with different tasks, etc.

The goal is to provide supporting material for Quality Assurance (QA) evaluations. Therefore, it is necessary to consider the various possible combinations that make up the context of use for an accurate evaluation. For instance, diverse traits of the user, potential resources that could impact the software application, different tasks to be performed, and changes in the environment according to where the user is. Thus, the goal is to create a methodology that uses simulation to provide supporting material for the QA analyst, tester, or any other professional in charge of evaluating the QinU of a software application, to reach a final decision on their analyses of certain quality characteristics. The QA analyst will be, therefore, the main user of the approach.

To propose a possible solution for the found problem, different assumptions were made. In this section, the working assumptions are introduced to support the fundamentals of the proposed approach.

Assumption 1. *It is possible to measure QinU from simulations.*

As defined in Section 2.1.2, QinU is usually obtained either (i) by measuring the characteristics computed from interaction log data or (ii) the users' perception. For case (i), we argue that we can simulate events to generate such data. For the measurement to occur, it is necessary to establish what QinU characteristics will be targeted since the beginning of the implementation of the methodology, as it is required to know what data needs to be extracted from the outcome of the simulation to provide enough material for the QinU measurement.

The approach allows to collect data from the log files of both the simulation and the application under evaluation, and merge them together according to established measures for the evaluated QinU characteristics (Figure 4.1). It does not require real people to play the role of end users, and instead, replaces them using simulation.

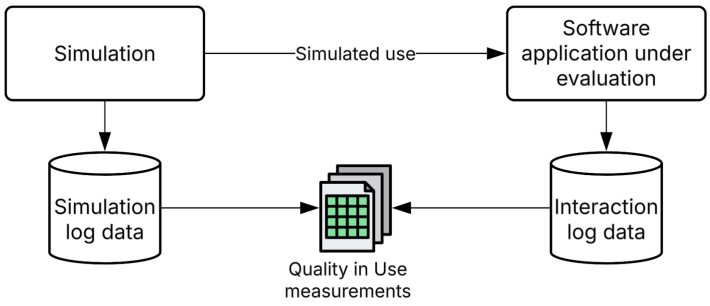


Figure 4.1: Targeted outcome for the employment of the approach.

To be able to extract sufficient information from the log data of both the simulation and the investigated application, it is necessary to model all the entities that compose the context of use so that all aspects needed are considered during the execution of the simulation.

Assumption 2. *The user and the devices of the smart environment can be modeled as agents.*

We argue that the agent-based model and simulation approach is appropriate for the implementation of the proposed methodology, as it enables to set up the model at an individual and micro level. This is a relevant feature, considering that the contexts of use are characterized by multiple entities that can interact with the application under evaluation. Moreover, agent-based simulations have been widely applied to represent human behavior

in groups (see Section 2.2.4) – even though its use is more rare to represent individuals [174].

A smart environment provides the possibility of two different kinds of interactions for their systems: explicit and implicit. While the former simulates moments where the user performs actions exclusively to get a response from an application (e.g., pressing buttons on a device), the latter refer to actions that will be caught by sensors that are placed in the surroundings. Regardless if the user intended to provide an input to the application, they can get an output resulting from their performed activities noticed by the sensors in place. Since both users and smart devices can potentially interact with the application that is under evaluation, we consider both of them as agents in the model.

To simulate the dynamics between the user and the software application under evaluation, we assume that it is necessary to model a *user agent* that plays the role of the resident of the smart environment (i.e., the end user), as well as *device agents*, to represent the smart devices found in place and/or any resources needed to access the targeted application. These can vary between smartphones, tablets, smartwatches, personal computers (as represented in Figure 4.2), and more. Furthermore, it needs to include the sensors, that are commonly found in smart environments working as input devices, and mostly, differentiate common homes from smart homes. Sensors can be of multiple kinds, e.g., motion, temperature, light, proximity, touch, and others.

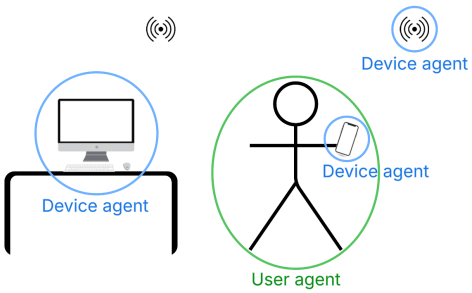


Figure 4.2: Examples of agents to be modeled in the ABMS.

Considering the need of modeling the agents, an additional actor (besides the QA analyst) is required to implement the methodology: ideally, the agent-based model (ABM) is supposed to be produced by an ABM designer, a professional responsible for creating such

systems. It is their task to understand how to enact traits and attributes that may impact in the usage of the software application under evaluation upon the demands of the QA analyst. For that, they need to interact and reach an agreement about the needs and requisites for this procedure. Figure 4.3 represents the interchange between the two actors: once their roles have been established, the first step is to clarify any questions they may have about the process and reach an agreement about the needs and the requisites for the targeted software application. Then, the ABM designer proceeds to create the model. Once the model is defined, the QA analyst can therefore run the simulation and, afterwards, analyze the outcomes.

It is vital that both actors are aware of the functionalities and purposes of the software application under evaluation, and it is necessary that both properly execute their parts. It is the responsibility of the ABM designer to prepare a model that supports different contexts of use, whereas it is the job of the QA analyst to run the simulation and accurately examine the multiple outcomes from the different scenarios.

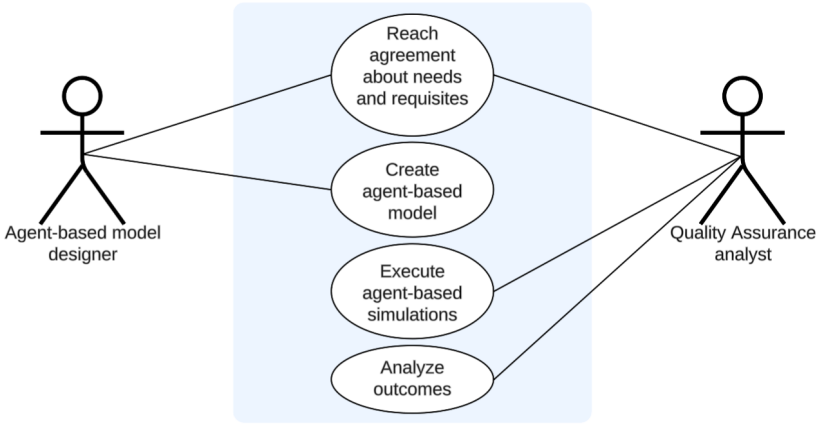


Figure 4.3: Actors and their roles in the execution of the methodology.

Assumption 3. *The ABMS can interact with the software application under evaluation.*

To avoid having to involve real people to interact with the software application under evaluation, we assume that it is possible to transmit events that happen in the ABMS to the investigated application. For that, we need a key software component that we named *ABMS observer*, which combines the calls to the ABMS and to the application. It is responsible for

analyzing what occurs in the simulation, as well as interacting with the application under evaluation accordingly, generating log data from both.

The ABMS observer needs to have access to the ABMS (to read any relevant occurrences in the simulation) and the software application (to communicate the simulated events). For that, it is required to consider the technical environment of both. The tools to be employed may vary according to the ABMS environment, as well as the platform that accesses the evaluated system.

From the interaction between the ABM designer and the QA analyst, it must be established what is significant for the observer to perceive and what should trigger interactions in the application under evaluation. Similarly, it is necessary that the observer knows how to engage with the application and reproduce end-user behavior according to the parameters of the user agent in the context of use planned by the QA analyst.

The context of use can be greatly impacted by the changes in the environment, as the surroundings of the user agent affect how situations may occur. For example, regarding the use of an application that outputs audio, the system needs to increase or decrease its volume according to where the user is at that moment, especially if it is someone with hearing issues. Another example is the case of a user that may only be able to provide an input to an application when they are in certain parts of their home, e.g., a spoken command may not be heard when the user is far from the system's microphone. All the resources that may impact the use of the application under evaluation, as well as all the possible inputs and outputs from such, need to be included in the model. It is essential that all the important components are modeled as clear variables in the ABM to facilitate the extraction of data for the QinU measurement.

Figure 4.4 presents the components and their respective flow from one to the other: (1) the *ABMS observer* initiates an observation of the *ABMS*, which in turn (2) provides (and can also receive) relevant events. The ABMS observer then (3) forwards the detected occurrences to the *software application under evaluation* in the form of interactions with

the system.

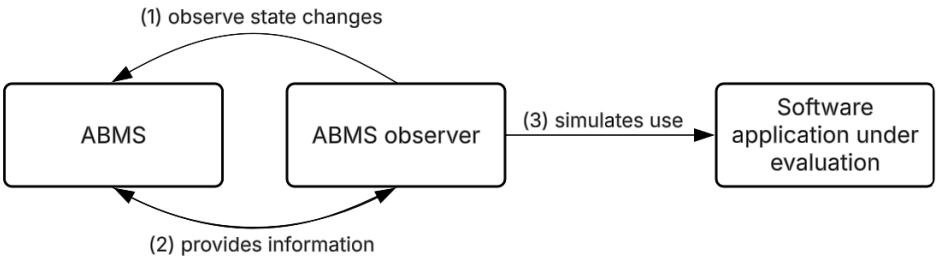


Figure 4.4: Components and their flows in the approach.

In the following sections, we will dive into details about the three main pillars of the approach: agent-based model, agent-based simulation and the calculation of QinU measures.

4.2 Agent-based modeling

To create the agent-based model following *Assumption 2* established in Section 4.1, it is necessary to consider different details during its design. The modeling phase addresses the typical elements found in agent-based modeling methods, including the two main points, that are the identification of the agents and their behaviors [84].

The targeted QinU characteristics also need to be established in the beginning of the process, as it impacts the modeling of the resources (what are the devices that need to capture information?), tasks (what are the explicit and implicit interactions that need to happen?), environment (where are the user and the devices supposed to be during the use of the application?).

Furthermore, the ABM must be able of reading and interpreting the input behavior patterns that will guide the agent-based simulation (ABS). These patterns may vary in structure, ranging from predefined agent needs to BDI models (see Section 2.2.4), or they may follow techniques grounded in real-world data. As we work towards simulating the dynamics between a real person living in a smart environment and interacting with its devices, we defend that the best approach to establish the agents' behavior patterns is through the

latter option, based on real-world data. We assume that grounding the approach on datasets made of real-life events has the potential to enhance the model's realism. For this reason, we propose to integrate a data mining step in the ABM phase, with the aim of grounding the agents' behaviors in real-life data.

In summary, three elements should be treated as inputs in the ABM phase: the variables of the context of use, the QinU characteristics that are going to be evaluated, and the outcome of the data mining step (see Figure 4.5). These three elements and their impact on the model are discussed in Sections 4.2.1, 4.2.2 and 4.2.3.

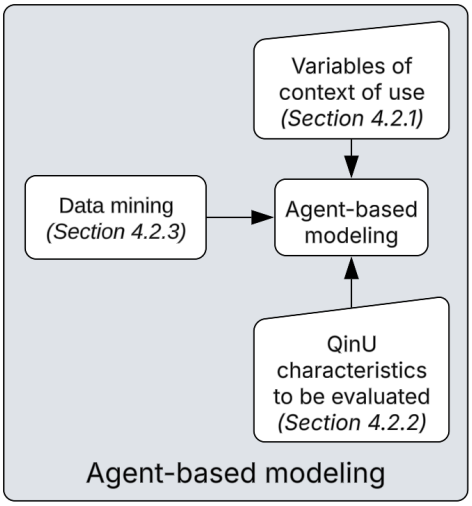


Figure 4.5: Agent-based modeling phase and its inputs.

4.2.1 Modeling the context of use

In this approach, each component of the context of use is represented as follows:

- ***users*** – the user of the software application under evaluation, i.e., the resident of the smart environment;
- ***resources*** – devices in the smart environment, including the technology necessary to run the targeted software application, such as tablets, smartwatches, and so on;
- ***tasks*** – actions the users perform to reach certain ***goals*** within the software application;

- ***environment*** – the surroundings in which the agents are situated, i.e., the smart environment.

Every element that shows some autonomy and has potential effects on the context of use or on the application itself should be represented as agents. This is the case of the user and the smart devices (*Assumption 2* from Section 4.1). Every agent has their respective attributes and responses to their surroundings – e.g., the user agent (resident of the smart environment) moves through the place and interacts with the software application, whereas the device agents (sensors and other technological tools) sense state changes and behave accordingly.

The actions performed by the user agent to reach a goal within the software application are the tasks that need to be included in the model. The ABM is not supposed to recreate the entire targeted application, especially to prevent errors in replicating its functioning. Instead, it must include the activities that the user can carry out with the system; these will be sent to the real application under evaluation, considering both explicit and implicit interactions. For that to be possible, different resources can be placed in the environment to capture inputs provided by the user. Figure 4.6 presents devices that can read explicit and implicit interactions performed by the user.

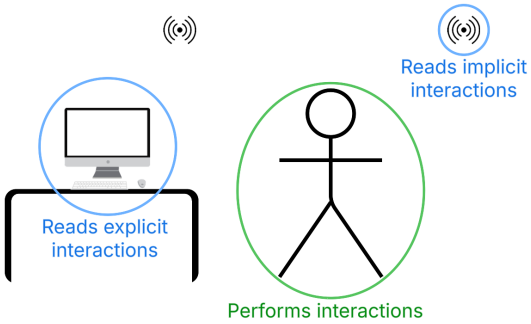


Figure 4.6: Agent examples and their roles regarding explicit and implicit interactions.

The entities and variables included in the ABM phase will define whatever comes from the simulation, and it is important to know what to extract from each element. On that account, it is required that the ABM designer and the QA analyst are in agreement as men-

tioned in the previous section.

4.2.1.1 Modeling users

The profile of the **user** allows to set the characteristics of the agent that represents the end user in the smart environment. For the variability of such traits, it is necessary to model them as variables (adjustable components that impact the simulation), allowing for such impactful characteristics to be changed for one user agent to the other in each run of the simulation, therefore representing people with different needs and abilities. This will influence the manner in which the user agents engage with the device agents. Identifying the details to be translated into variables requires an understanding of the factors that may influence how users interact with the software application under evaluation. One possibility is to observe the target audience and notice what are their traits that most affect the usage of applications, and how it varies. For instance, it could be the movements speed at which the user navigates through a mobile app, or the pattern of usage. For the ABM designer to be able to model relevant features, they can define the answers for the following questions: Who will be the end users of the software application under evaluation? Who is the target audience for this evaluated system? What are their traits that could impact the usage of application? How frequent is the application used?

In summary, the model should incorporate user agent traits that are potentially relevant to the software application under evaluation – for example, their activity performance speed, behavioral tendencies, and other related characteristics. As an illustration, to evaluate an application that alerts the user when it is time to take their medication according to how far they are from the assigned place for that, it is necessary to consider the position of the user agent in the house, as well as their slowness of movement to know how long they would take to arrive there. The user's profile characteristics are, therefore, variables that will be set in the modeling phase.

4.2.1.2 Modeling resources

The resources refer to the sensors and other devices (such as cellphones, tablets, smartwatches, and computers) that are essential for achieving specified goals within the software application. Enabled by IoT capabilities, these devices can exchange information (Figure 4.7). In this context, sensors are considered part of the device set. All these interactions between the devices, as well as their communication with the targeted software application under evaluation, need to be represented in the ABM.

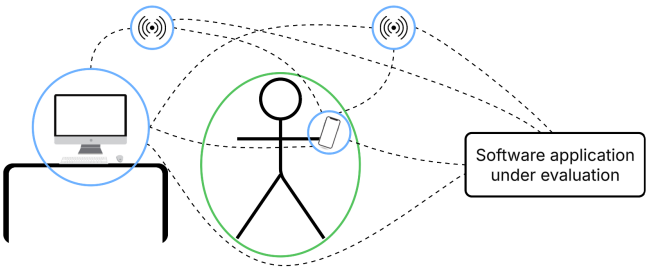


Figure 4.7: Device agents are connected and share information.

Resources that can impact the use of the application need to be represented in the ABM as agents. In this approach, they are called device agents and can be of different kinds, with their own state variables and behaviors. The state variables depend on the data the device agents are sensitive to: light, heat, collision, etc. The behaviors depend on how the device agents act and on the subsequent changes in the environment: window shutters react to the lighting, a moving vacuum cleaner robot detects obstacles, and so on. Hence, it is necessary to know what they perceive and how they act to know which variables they should have (if any).

To model the resources, the ABM designer can establish responses for the following inquiries: What are the smart devices and gadgets that the users will interact with to receive a response from the system? Which resources are necessary for the users to interact with the targeted software application (such as Internet connection, mobile devices, etc.)?

4.2.1.3 Modeling tasks

The **tasks** to be modeled are actions performed to reach a specified goal (regardless if they are performed towards the software application or not), which may require explicit or implicit user interactions. Therefore, the tasks are the actions to be perceived by the devices in place, i.e., the resources described in the previous subsection. The explicit interactions are carried out by the user agent when performing some activity on a device, while the implicit ones are captured by the sensors.

For instance, a software feature to close the window shutters can be carried out implicitly by the light sensors when it gets dark or explicitly by the user whenever they decide to close them. Hence, the inputs and outputs resulting from the implicit or explicit user interaction should also be modeled in this phase. The ABM should recreate not only the human behavior related to the explicit use of the application, but also the behavior that can trigger the implicit use of the application through the devices in the smart environment.

The interactions between the users and the smart devices will vary according to the former's characteristics and behavior, while the latter are supposed to provide a response as soon as these interactions are received. The variables of the context of use need to serve as inputs in the ABM, as seen in Figure 4.5. Moreover, the QinU characteristics to be evaluated need to be considered from the beginning of the procedures, so that the ABM designer is aware of which tasks are necessary to be modeled (hence, which inputs should be provided to the application under evaluation).

As discussed in Section 2.4, systems designed for smart environments can read implicit and explicit interactions. Therefore, it is crucial to model the activities recognized by the applications along with their associated inputs. Implicit inputs originate from the user's actions within the household environment, detected by sensors, whereas explicit inputs result from deliberate user actions, such as pressing a button or issuing a clear voice command to the system.

Modeling these inputs is crucial, as they directly influence the interaction outcomes, which are fundamental for measuring QinU characteristics. For this reason, it is necessary to determine which application functionalities will be targeted and which QinU characteristics will be evaluated. The selected functionalities define the tasks the user agent must perform to interact with the application, while the chosen characteristics establish the expected outcomes that should be identifiable in the log data to enable measurements.

4.2.1.4 Modeling environments

In the context of quality assessment, the term environment encompasses "the technical, physical, social, cultural and organizational environments that influence the usage of the software system" [21]. According to the agent domain terminology, the agents are situated in a local context, hence, for the design of the ABM, it is necessary to establish which of those dimensions should be included in this local context to be represented.

In a smart environment, the technical and physical dimensions are extremely important, as the end user is spatially situated in a place (*physical environment*) while surrounded by diverse smart devices, which are resources that can interact with different software applications (*technical environment*). In these spaces, connection distances and sensing areas matter. Furthermore, the simulated environment has its own internal clock, which may be an advantage (as it can generate data faster than it would according to a real-world clock) or an issue (as in the evaluation of a software application it may need to consider the real time for its usage).

To be able to model the environment, the ABM designer needs to answer the following questions: Which spaces should be modeled? Where is the user located when interacting with the software application under evaluation? How does it influence the usage? Where should the device agents be placed? Should the real-world clock cause some impact in the simulated events?

The environment impacts greatly how and when the software application may be used. For such variability to take place, it is necessary to model the environment's own characteristics in order to impact the user and its interaction with the software application under evaluation. Such characteristics can make events more or less probable to occur, as well as allow or avoid interactions with systems to happen.

For example, in the event of the user forgetting the stove on, sensors placed in the kitchen would be the ones to notice the occurrence. In the event of the user falling down because they slipped, it is more likely that such episode happened in a room with a slippery floor. In the event of the user deciding to interact with a software application that belongs to a hardware that is fixed somewhere (as opposed to a mobile device that they can carry), the user has to take their time to go towards it. Consequently, it is essential to take into consideration the surroundings of the agents in the simulation, which change according to the data input from the process mining solution (discussed in the next subsection).

4.2.2 Modeling data to evaluate Quality in Use characteristics

Considering *Assumption 1* of Section 4.1, it is required to define which are the QinU characteristics that should be evaluated. To that end, the QA analyst should engage with the clients (people responsible for requesting the development of the software application under evaluation) to identify the essential non-functional requirements regarding quality that must be met.

Then, the QA analyst should discuss with the ABM designer to inform them which QinU characteristics need to be measured in alignment with the client's requirements, as well as to determine the data required for this evaluation. From this data, the ABM designer will decide what they need to model in order to represent the user of the application, the application itself, and the devices necessary for its utilization.

In order to determine the data required for measuring specific QinU characteristics, it

is essential to examine established QinU measures, such as the ones defined in the ISO/IEC 25022:2016 standard (see Section 2.1.2). In this thesis, we consider the measures proposed by the ISO/IEC standards, since they were established to be applied to any kind of system. The QinU characteristics to be evaluated must be taken into account during the development of the ABM, as the inputs to the software application under evaluation need to be modeled. This is also the case for the output of the simulation: its content must be adapted taking into consideration the data that needs to be extracted in order to compute the QinU measurements.

As previously presented in Section 2.1, QinU is divided into three categories: beneficialness, freedom from risk and acceptability [21]. The following subsection analyzes each category in detail, aiming to determine whether these characteristics can be measured through simulation.

In these analyses, we specify what data is necessary to measure each of them, and consequently, what needs to be modeled in the ABM. The necessary data was either drawn from the measures proposed by the ISO/IEC 25022:2016 standard [22] (see Appendix A) or from the definition of the characteristics by ISO/IEC 25019:2023 standard [21]. From there, we identified the elements that need to be modeled in the ABM to enable the extraction of relevant data from both the simulation and the software application under evaluation.

4.2.2.1 Beneficialness

For beneficialness, Table 4.1 outlines its characteristics (usability, accessibility, and suitability).

As previously detailed in Section 2.1.2, usability is composed of effectiveness, efficiency, and satisfaction. For this reason, it has been divided into its three sub-characteristics in Table 4.1. These sub-characteristics are each linked to a measure proposed by the ISO/IEC 25022:2016 standard.

Table 4.1: QinU characteristics from the Beneficialness category [21].

Characteristic	**Measure description**	**What is the necessary data?**	**What is necessary to model?**
Usability	Effectiveness (tasks completed): accuracy and completeness with which identified goals are achieved [22].	The amount of tasks completed and the amount of tasks attempted.	The interactions the user performs (or tries to) in the application.
	Efficiency (task time): resources used for the results achieved [22].	The time when a task started to be performed and the time it finished.	The range of the time period in which the user performs actions towards the application.
	Satisfaction (overall satisfaction): extent to which a user perceives the results of use of a system to meet their needs [22].	Responses to questions.	This is not possible to model, as it is based on the users' perception, which is usually collected through questionnaires or interviews.
Accessibility (adapted from [21])	Accessibility range: extent to which systems can be used by people with a wider range of user needs.	The total amount of features in the application and the amount of features that are successfully utilized by users with special needs.	Interactions towards the application performed by a group of users with special needs.
Suitability (adapted from [21])	Suitability supported by simulation: extent to which behaviours or outcomes (or both) of a system meet specified quality requirements that do not depending on the users' opinion.	Identify all data from QinU characteristics that was set as requirements and that do not depend on the users' opinion. Consider different kinds of user profiles.	All the necessary data modeled for the QinU characteristics that can be simulated (effectiveness, efficiency, accessibility and freedom from risk).

To evaluate effectiveness according to Table 4.1, it is required to collect both the total number of tasks successfully completed within the application and the total number of user attempts. Consequently, it is essential to model not only the successful interactions of the user agent with the application but also all failed attempts (whatever the reason for the failure may be in that scenario). In this case, the total amount of attempts should be gathered from the simulation log data (as all occurrences should be observed and registered in the ABMS), and the total amount of completed tasks should be obtained from the interaction log data (from the software application under evaluation). In the case of efficiency, the time spent to execute the tasks is needed. Therefore, it is necessary to model the time period range that the user agents will take to perform each evaluated task within the software application. In contrast, note that satisfaction cannot be evaluated through simulation,

since this measure is based on the users' opinions, which we consider are not possible to model through agents.

As mentioned in Section 2.1.1, there are no ISO/IEC standards that propose measures for the QinU characteristics recently defined in the ISO/IEC 25019:2023 standard [21], such as **accessibility** and **suitability**. Nevertheless, for accessibility, we argue that the "wider range" of user needs can be represented by different kinds of disabilities or difficulties, such as motor impairment, eye conditions, hearing issues, and more. Based on this, it is possible to estimate the necessary components for the measurement: user agents with different traits (including special needs) and their implicit and explicit interactions towards the application under evaluation. Analyzing established measures from the standards, we suggest that a possible manner for evaluating this characteristic would be comparing the total amount of features from the evaluated application to the amount of features that are prepared to be utilized by users with special needs. One possible approach to carry out this measure would be for the QA analyst to execute simulations with user agents representing multiple people with different special needs, and according to their expertise, judge if the software application is adequate for them or not. At this point, they would have the necessary data to evaluate such characteristic. Then, the measurement for this characteristic should be further adapted considering the specific software measure for that end.

Suitability, in turn, refers to the extent to which outcomes of a system meet specified quality requirements. This means it should consider beneficialness, acceptability and freedom from risk [21]. In our case, we are considering only the QinU evaluation specifically when it does not depend on the users' opinion. Since acceptability and satisfaction (from usability) are, by definition, subjective quality requirements, we decided to adapt this measure to consider only effectiveness, efficiency, freedom from risk and accessibility. Data required for the measurement of such characteristics are described in their own categories (Table 4.1 and 4.2, respectively). However, as stated in the ISO/IEC 25019:2023 standard, suitability is "important because it affects the gain or loss of opportunities for a more di-

Table 4.2: QinU characteristics from the Freedom from risk category [21].

Characteristic	Measure description	What is the necessary data?	What is necessary to model?
Freedom from economic risk	Return on investment (ROI): extent to which a system mitigates the potential risk to financial status or other economic aspects [22].	The total amount of money involved and its returns.	Financial threats that may impact investments and business.
Freedom from environmental and societal risk	Environmental impact: extent to which a system mitigates the potential risk to the environment and society at large [22].	The characteristics of the area at risk to be endangered before and after a disaster.	Possible threats to nature and society as a whole, such as a wildfire.
Freedom from health risk	User safety risk: extent to which a system mitigates the potential risk to people's health [22].	The total amount of implicit interactions that occurred and the amount that was registered by the software application.	Possible issues the user may face while using the application, and scenarios where its flaws may cause harm.
Freedom from human life risk (adapted from [21])	Safety of people affected by the use of the system: extent to which a system mitigates the potential risk to people's lives.	The total amount of users and the amount of users put at hazard.	Scenarios where flaws from the application may put the user's life in danger.

verse group of stakeholders to obtain valuable benefits of use". Ideally, the ABM should include a variety of users (by modeling multiple different traits to represent various people) and of tasks (by modeling different activities to be performed within the application under evaluation).

4.2.2.2 Freedom from risk

Table 4.2 presents the characteristics from the category of freedom from risk. The first one, **freedom from economic risk**, can be evaluated with measures proposed by the ISO/IEC 25022:2016 standard. It is usually applied to software related to finances and commercial business, thus, one of its proposed measure requires information about investments and its returns. The simulation should include possible financial threats that could impact capital and feed the updated information to the software application under evaluation accordingly.

Freedom from environmental and societal risk aims at evaluating possible dam-

ages to the environment, and consequently, society as a whole. For that, the simulation needs to represent the possibly endangered area, such as a forest in the case of a wildfire, and report to the software application (in this case, likely a disaster management system, for predictability and forecasting) the details and pace of the calamity.

Freedom from health risk and its proposed measures focus on estimating the likeability of the users jeopardizing their health during the use of the application. The ISO/IEC 25022:2016 standard suggests three measures for this characteristic: user health reporting frequency, user health and safety impact, and safety of people affected by use of the system. We adapted the first one to be applied through the use of simulation and named it "user safety risk". In this case, the act of reporting (originally expected to be carried out by end users) was replaced by the observation of events that did or did not occur, as recorded in the log data. To that end, we adapted the measure to treat each unregistered implicit interaction that would potentially affect the user's health (detected by device agents in the simulation) as a flaw. This contrasts with what was suggested by the standard [22], which proposed counting user-submitted reports.

Regarding **freedom from human life risk**, this characteristic has been introduced only in the most recent QinU standard [21]. As the previous characteristic, it deals with the possibility of people being harmed due to some flaw of the software application but considering the risk of human life. Hence, we based ourselves in a measure proposed for freedom from health risk, more specifically the one named "safety of people affected by the use of the system". This measure is described as the incidence of hazard on people caused by use of the system. We argue that the hazard mentioned here could be seen as life threatening, and for that reason, adapted it for this characteristic.

4.2.2.3 Acceptability

Table 4.3 concerns the category of acceptability: **experience**, **trustworthiness**, and **compliance**. None of them have had measures proposed by the ISO/IEC 25022:2016 standard

Table 4.3: QinU characteristics from the Acceptability category [21].

Characteristic	**Measure description**	**What is the necessary data?**	**What is necessary to model?**
Experience (adapted from [21])	Overall experience: extent to which users accumulate knowledge acquired over time, especially gained in a particular profession.	Responses to questions.	This is not possible to model, as it is based on the users' perception, which is usually collected through questionnaires or interviews.
Trustworthiness (adapted from [21])	Overall trustworthiness: extent to which users have confidence that their expectations are met in a verifiable way.	Responses to questions.	This is not possible to model, as it is based on the users' perception, which is usually collected through questionnaires or interviews.
Compliance (adapted from [21])	Overall compliance: extent to which a user has confidence that a system meets requirements, as required by rules or laws.	Responses to questions.	This is not possible to model, as it is based on the users' perception, which is usually collected through questionnaires or interviews.

as they are newly defined. However, according to their definitions, we note that all of them take into consideration users' opinions and perceptions towards the system. We assume that to gather such information, a questionnaire would be needed, as in the case of evaluating satisfaction. For this reason, we believe that they cannot be modeled within an ABM.

4.2.2.4 Synthesis of findings

The previous analyses have been made considering the ISO/IEC 25022:2016 standard [22], as it proposes measures for QinU evaluation. However, these measures should be refined for each case, depending on the selected measure and the application under evaluation. In the case of specific measures to be defined or selected from the literature, a detailed examination should be performed in order to adapt them as necessary. Nonetheless, even though we tailor the measures to the specific case under study, the methodology should always follow the same steps that we described.

From this discussion, we observe that the QinU characteristics that take into consideration the users' subjective feelings and opinions cannot be measured by simulation. This is because we believe it is not possible to model "how the user feels regarding the use of the application" (satisfaction) and all measures regarding acceptability, because it regards the users' acceptance. In addition, we suppose it is difficult to model "knowledge and skills" (experience) the user acquires throughout the use of the application, as this would vary according to the cognitive abilities of each user. Hence, this methodology cannot be applied to evaluate all QinU characteristics, which reinforces the need of determining the targeted ones at the beginning of the procedure.

As established in *Assumption 1* of Section 4.1, we defend that it is possible to simulate outcomes to measure QinU characteristics that do not depend on the users' perceptions. Table 4.4 summarizes the QinU characteristics discussed in this section. The second column indicates whether or not, in our opinion, they can be measured through simulations. This depends on the necessary elements and what we envision that can, in fact, be modeled in the ABM. According to our analysis, eight out of the twelve QinU characteristics (66,67%) can be evaluated with the application of the methodology proposed in this thesis.

Table 4.4: Evaluation of QinU characteristics from ISO/IEC 25019:2023 [21].

Characteristic	**Possible by simulation?**
Effectiveness	✓
Efficiency	✓
Satisfaction	✗
Accessibility	✓
Suitability	✓
Freedom from economic risk	✓
Freedom from environmental and societal risk	✓
Freedom from health risk	✓
Freedom from human life risk	✓
Experience	✗
Trustworthiness	✗
Compliance	✗

4.2.3 Data mining

Assumption 2 of Section 4.1 introduced the idea of reproducing the dynamics of a person in a smart environment. Given the inherent difficulty in validating a model designed to reflect human cognitive processes, the approach of this thesis is positioned in accordance with research based on real-world evidence (see Section 2.3.3), with "inputs for ABMS that are drawn from empirical data" [116].

Assuming that an existing dataset can be used as-is (without any modification) to evaluate the QinU of a new application is unrealistic. Consider, for example, a scenario in which one aims to replicate the functionality of an application that reminds users to take their medication. The user is expected to turn off an alarm once the task has been completed. Even if a suitable dataset is available, where the recorded individual is indeed required to take medication daily (e.g., at 8 a.m.), it is highly improbable that the person would go to the location of the device where the app is installed immediately after. Consequently, a user agent that strictly follows this person's recorded trajectory cannot reasonably be expected to press (within the simulation phase) the turn off button on the app. Therefore, it becomes necessary to find a way to stay grounded in real-world data while being flexible enough to support the QA analyst's objectives for the software application under evaluation.

We argue that the best way to achieve this is by applying data mining techniques to extract patterns from real-world data and subsequently generate artificial data that is tailored to the QA analyst's needs, while ensuring a high similarity with the data that was used for its generation. In particular, in this thesis we employ methods derived from the field of process mining. A central premise of process mining is the use of event logs, where processes are typically represented as sequences of time-stamped events that mark the start and end of instances. This aligns well with the structure of the data we aim to analyze, as it is stored in a comparable format. From that, a model is built without using any information other than the source material [102].

As previously stated, finding a dataset that completely matches our problem is nearly impossible, taking into account that the QA analyst is interested in exploring as many combinations as possible for the parameters in the simulation. Therefore, the preliminary step is to *select a dataset* that represents as much as possible the practices that the model must include. Next, potential patterns (i.e., repeated chronologies of events) are extracted with the purpose of guiding the modeling of the behavior of the users and the devices placed around them. Note that the ABM needs to be coded in a way that allows reading and interpreting these patterns.

As the objective of the data mining phase is to supply the ABMS with realistic patterns typical of smart environments, these patterns may reflect either human movements – captured through mobile device geolocation or motion sensors that activate based on individuals' locations – and reproduced by user agents, or device operations – such as thermostat adjustments or the opening and closing of a refrigerator door – which are simulated by device agents.

The outcome of this phase will be provided to the ABM in the form of probabilistic models, thus allowing the user agent to follow, at each execution, different combinations of behavior patterns.

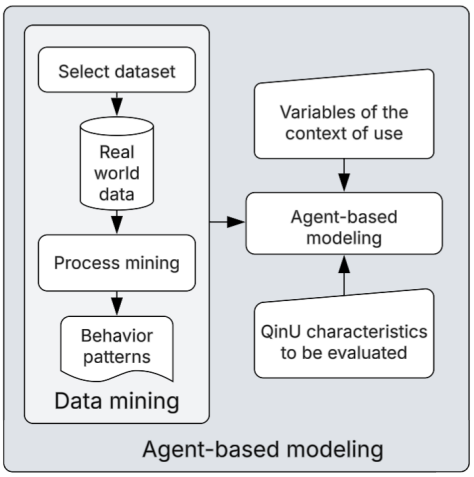


Figure 4.8: Detailed data mining phase during the agent-based modeling.

To conclude, Figure 4.8 provides a detailed version of the data mining phase. First, it

involves the identification of potential patterns from the previously selected dataset (*process mining*). As it often happens when working with real-world data, pre-processing needs to be performed to check possible inconsistencies and to get rid of outliers before establishing a procedure to get the probabilistic model, for which, a third professional (besides the QA analyst and the ABM designer) may be required. Then, the next step consists of generating *behavior patterns* to be provided as an input in the agent-based model (ABM).

4.3 Agent-based simulation

The agent-based simulation consists in, essentially, putting the agent-based model into action. This means that all the dynamics of the modeled entities and their behaviors will take part in how the agents act and react, and all events that transpire. The goal is to reproduce a person living in a smart environment surrounded by devices with some autonomous behaviors which receive and analyze information of what goes around them. The benefits of using the approach appear at this stage of the process: variations of contexts can be simulated using different combinations of parameters related to the user's and the environment's states. The simulation should put in motion the elements that were modeled, allowing the agents that represent people to emulate the behavior expected from them while engaging with the systems found along the way in the simulated smart environment.

The simulation can provide different results at every execution, thanks to the stochasticity of the methodology (as introduced in Section 2.2.3). As the environment affects the user agent performing tasks (and consequently, what the resources in place read), the events occur accordingly. Hence, even if a simulation is ran with the same user agent, it would produce a variability in the outcome, similarly to how it goes in real life, where a resident of a smart environment does not perform the exact same activities in the same way every day.

Prior to the execution of the simulation, the QA analyst is supposed to set up the pa-

rameters (defining values to the variables established in the ABM phase). These values will vary according to the contexts of use that the QA analyst would like to test and analyze. Once the simulation is running, synthetic data will be generated and the user agent performs their actions as planned by the ABM designer, provoking explicit and implicit interactions read by the device agents present in place. The relevant events that occur in the simulation are saved in the simulation log data, while every transmitted interaction to the software application should be saved in its respective log data. This is possible because of the implementation of a "bridge" between applications, the ABMS observer (introduced in *Assumption 3* of Section 4.1), described in the upcoming subsection.

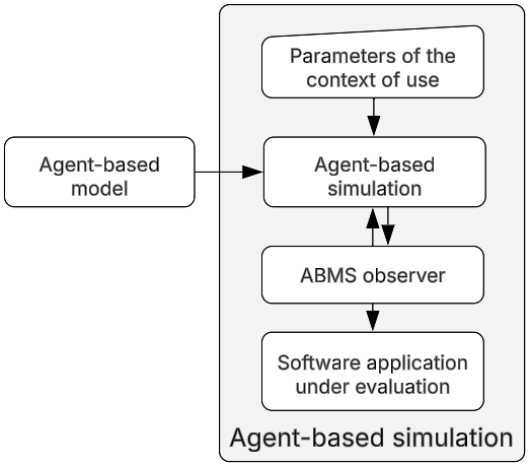


Figure 4.9: Agent-based simulation phase and components.

Figure 4.9 represents the execution of the ABS, with the parameters of the context of use serving as input. Then, the ABS has a two-way connection with the ABMS observer, that transmits relevant events to the software application under evaluation. In the next subsection, we will discuss in detail the observer.

4.3.1 Agent-based model and simulation observer

The simulation generates events of the application use that must be transmitted as inputs to the real application. For that, the *ABMS observer* (introduced in Section 4.1) makes the connection between the simulation and the real application, as established in *Assumption 3*.

This key element combines the calls to the simulator and to the application, according to the functionalities under evaluation and the simulated context. These calls generate log data – both from the simulation and the software application.

The observer development is impacted by the technical environment of both the ABMS and the software application under evaluation. For it to accurately work, it is necessary to adapt it to the ABMS it is observing, considering the platform and coding language employed for such, so that it is possible to read what happens in the simulation and transmit information to the software application.

Similarly, it should adapt to whatever kind of application is under evaluation. Depending on the type of application (whether it is an online application or a software locally installed on the machine) distinct tools and procedures must be employed. Python, for instance, has various libraries to interact with different kinds of application, such as Selenium for online applications and AutoIt for local software.

The observer sets up and starts the ABMS and the software application to be evaluated so that it can read and engage with them. The events occurring in the simulation cause state changes impacting the device agents (e.g., the cooktop switched off) and/or the user agent (e.g., the user fell down). When the events represent the use of the software application, either in the form of automatic actions from device agents or by manual actions from the user agent, the observer transmits the related actions towards the application (*Assumption 3* in Section 4.1). The device agents' actions simulate implicit interactions, while the manual actions from the user agent simulate explicit interactions with the software application.

It is necessary for the observer to know what to look for, and which events in the simulation are significant and should trigger interactions in the software application. Likewise, it is crucial for the observer to be able to engage with the application under evaluation. For the events that may trigger interactions with the software application, the observer needs to detect relevant changes in the simulation components states. This can be represented in different forms, such as the outcome of a procedure that generates a number, the change of

a shape, a certain time in the simulation clock, a specific location reached by the agent, for example.

The interaction events are reproduced on the real application's interface to create a contextualized interaction log, which stores information regarding usual log data (session identifier, IP address, etc.) and the timestamps for when the application is accessed and actions are performed. In addition, the simulation log records data about contextual information, related to the states of the user agent, the environment and the simulated event that caused the interaction to occur. The exact content of the output data must be adapted to the purpose of the simulation and to what is necessary for the calculation of the quality measures in the next phase.

Based on *Assumption 1* of Section 4.1, it is required to analyze and extract information from the *simulation* and the *interaction log data* originated from the execution of the methodology to evaluate the QinU characteristics by measurements. Figure 4.10 presents two log data and examples of information that may be found in them. The *simulation log data* will present its information according to what has been modeled in the ABM, whereas the *interaction log data* varies in accordance with the software application under evaluation.



Figure 4.10: Simulation and interaction log data and their contents.

The simulation log data needs to contain information about the user agent for that execution of the simulation, which may be the age, gender, or any user's traits included in the ABM. It might as well record the user agent's movement through various locations, what sensors (regardless of their type) detected their presence, which devices received some sort of interaction (implicit or explicit). These events need to be communicated to the software application under evaluation. The ABMS observer monitors what happened in

the simulation and interacts with the application, just as an end user would do. These interactions are registered and gathered in the interaction log data (recording events of multiple kinds with their timestamps) depending on the inputs that need to be provided and on the features found in the application. Such events can be clicks, scrolls, typing, accessing different parts of the system and so on. This will depend on the QinU characteristics defined as non-functional requirements and chosen to be evaluated (see Section 4.2.2).

4.4 Quality in Use evaluation through measures

The necessary data to calculate the measures are extracted from both log data. Merging the records produces the data for the QinU report of the evaluated software application further used by the QA analyst (*Assumption 1* of Section 4.1).

Considering the proposed measures and the possible outcomes from the simulation, eight QinU characteristics can be measured (see Section 4.2.2) using the log data generated (either the simulation log data originated from the ABMS, or the interaction log data originated from the software application). Table 4.5 suggests data sources for each measure. The information from both log data should be matched and merged according to pre-established identifiers. It is important to note that this represents one possible implementation of the methodology, rather than a mandatory approach.

For the results of these calculations to be relevant during the QinU evaluation, the QA analyst is expected to analyze them based on their expertise and in accordance with established standards and the client's requirements. Once the report is under review by the QA analyst, relevant charts can be generated for a better visualization and comparison of the outcomes (such as bar charts, pie charts, and more).

A possible issue may come from the fact that a simulation has its own internal clock, which is different from a real-world one. This can create difficulties if the specific time for the occurrence of events is necessary to calculate some QinU characteristic. However, the

Table 4.5: QinU characteristics from ISO/IEC 25019:2023 [21].

Characteristic	**Measure**	**Data from simulation log data**	**Data from interaction log data**
Effectiveness	Tasks completed	Total amount of tasks completed and its the attempts	Amount of tasks completed in the application
Efficiency	Task time	Range of time period for the performance of tasks	Initial and final time for the completion of a task
Accessibility	Accessibility range	Different user profiles with special needs interacting with the application	The total amount of features in the application and the amount of features that support special needs
Suitability	Suitability supported by simulation	A variety of user profiles and tasks to be performed	Registration of the interactions performed by the different user profiles, e.g., the measures from beneficialness or freedom from risk
Freedom from economic risk	Return on investment (ROI)	The total invested amount of capital	The additional benefits obtained
Freedom from environmental and societal risk	Environmental impact	The target environmental impact	The actual environmental impact
Freedom from health risk	User safety risk	Total amount of implicit interactions that occurred	Amount of implicit interactions registered by the application
Freedom from human life risk	Safety of people affected by the use of the system	Total amount of users	Amount of users put at hazard

fact that the simulation and the real-world clocks are different also has its benefits, as this is how the QA analyst can get results in far less time than they would if actual people were involved in creating log data for the QinU evaluation. To avoid such difficulties, a further study may consider a digital twin instead of a simple simulation, to allow real-time interaction. Figure 4.11 provides a visual representation for the last phase of the methodology.

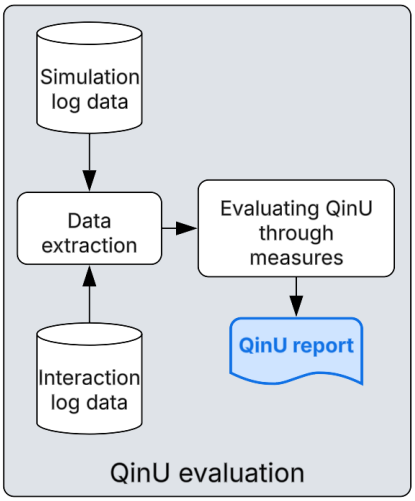


Figure 4.11: QinU evaluation and its components.

4.5 Potentially executable contexts of use

In an in-vivo evaluation of a software application, it would be necessary to let people use the application for a sufficiently long period of time in order for events to occur, such as issues related to the users themselves or technical problems (e.g., Internet connection). Since the QUESTS methodology connects directly to the real application, these events can be recreated, and as many simulations as needed can be executed to collect a sufficient amount of data for evaluation.

The application is being tested using the QUESTS methodology in a black box way [175], since we only know the application's input and output, not its internal coding. Simulation in QUESTS allows several test scenarios to be reproduced by considering the variability of the context of use. To estimate the number of possible scenarios that can be simulated using

this methodology, we apply the concept used by black box techniques, considering how the different values of the context of use variables (users, resources, tasks and environments) can be combined. This process corresponds to a Cartesian product of these elements. The result is the complete set of possible scenarios, representing all combinations that may occur within the simulated contexts of use. In practice, however, it is important to determine which part of the application is of interest to evaluate, since this decision determines which values of those variables and combinations are most relevant. The total number of possible scenarios that can be generated through the QUESTS methodology can be expressed as:

$$X = U \times R \times T \times E$$

- U: number of distinct user profiles, each defined by a set of traits;
- R: number of distinct resources configurations that influence the tasks required to use the evaluated features of the application;
- T: number of tasks related to the evaluated features of the application;
- E: number of temporal or spatial configurations in the environment.

To determine the maximum number of user profiles (U), all possible combinations of user traits are considered. Therefore, to calculate the total number of user profiles, we need to consider the Cartesian product of all traits values. For example, consider two traits with the following values: blindness (total, partial or none) and biological sex (male or female). The total number of possible user profiles U = 6 (3×2), representing all combinations of the two traits values. It is important to note that in some specific domains, not all combinations are valid and the QA analyst should not consider those combinations in the evaluation. The same idea should be used to establish the maximum number of R. For each resource considered in the evaluation, we should identify the different values (states) to be tested and generate the Cartesian product. For instance, consider a case with two motion sensors, each having the values “on” and “off”, and one humidity sensor with the values “very

high", "high", "medium", "low", "very low", then R = 20 ($2 \times 2 \times 5$). This represents all possible combinations of sensor values. The QA analyst should decide which combinations are pertinent to the task they want to evaluate and to the environment characteristics.

To establish the maximum T, we are considering only the number of tasks to be executed and not all possible ways of executing each task, since we consider the QUESTS methodology will be used after the classical test procedure where black box and white box techniques are applied (that is unit test, integration test, and system test). However, the quality analysis can also decide to test some specific configurations of the task from the point of view of the user.

Finally, for E, we consider each environment or variations of the states of these environments. For example, in a household setting, we may only be interested in certain rooms (bedroom, bathroom or kitchen), so E = 3. In another situation, we might be interested in considering a characteristic of these environments, for example lighting ("light" or "dark") during different moments of the day. In this case, if we want the same characteristics for all rooms, E = 8 ($2 \times 2 \times 2$). Once again, the QA analyst should decide what is important to vary for the task being evaluated.

In summary, QUESTS allows us to evaluate how the application is executed in all possible combinations of user, resource, task and environment values (context of use).

4.6 Conclusion

This chapter focused on each of the important parts that compose the proposed methodology, starting from process mining, a data mining technique commonly used to extract probabilistic models from sources that have events registered with timestamps. This is employed in the approach to generate plausible variations in the behaviors of the entities represented as agents in the model. The agent-based model is then built considering the QinU characteristics to be evaluated and a set of variables that can impact the context of use.

To reproduce the actions from the modeled elements, the agent-based simulation is supposed to recreate action sequences that happen in the smart environment, including explicit and implicit interactions between the user agent and the environment around them; these occurrences are then sent to the targeted software evaluation. This allows for multiple changes in the context of use, which is considerably easier than requiring actual participants to be placed in the environment to provide inputs for the application under evaluation. Lastly, the QinU characteristics are measured and analyzed according the ISO/IEC standards that establish patterns to be followed in the domain.

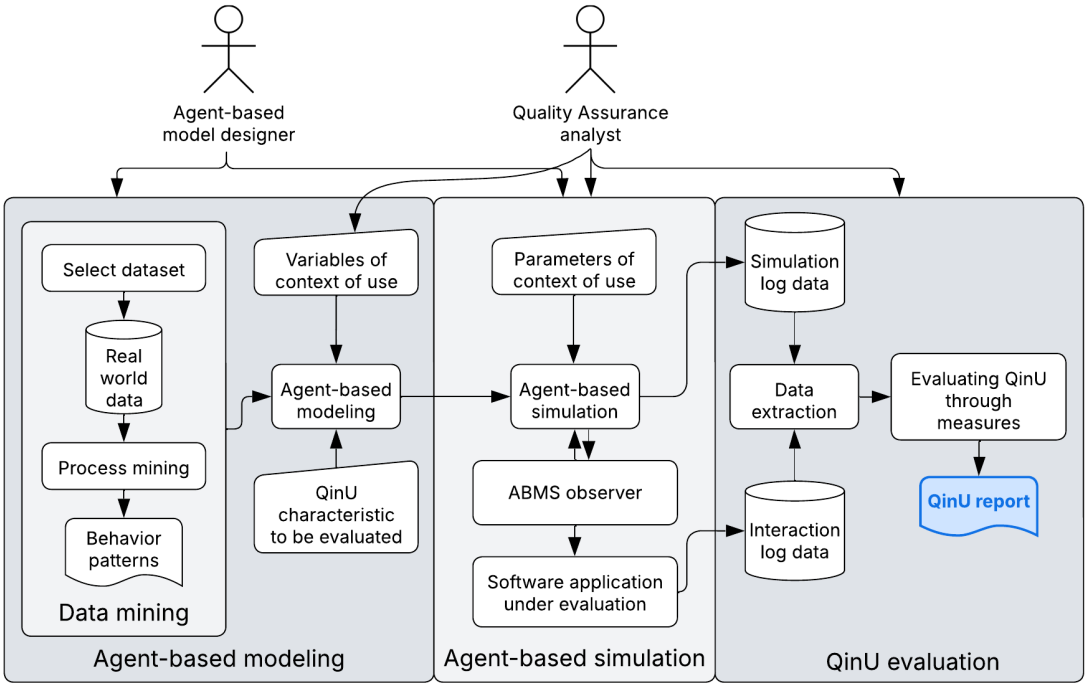


Figure 4.12: QUESTS: a three-phase methodology for the evaluation of the QinU of software systems.

Figure 4.12 synthesizes the QUESTS methodology proposed in this thesis for the QinU evaluation of smart environment applications. The *agent-based modeling* phase includes a *data mining* part that consists in selecting a real-world dataset based on the quality requirements, and mining it to obtain a probabilistic model that provides behavior patterns. These patterns need to be considered to guide the agents behaviors of the *agent-based modeling* phase. The context of use is parametized for the *agent-based simulation*, where the

interactions with the software application under evaluation are simulated. The data generated by these interactions are then exploited in the third and last phase (*QinU evaluation*) to evaluate the targeted application, producing a QinU report that serves as a supporting material for the QA analyst to decide whether it is ready for end-user evaluation or should be improved.

The following chapter will integrate various tools in a practical implementation of the methodology, demonstrating its applicability and feasibility.

Chapter 5

The application of the QUESTS methodology

The previous chapter presented the QUESTS methodology, proposed for evaluating the QinU of smart environment applications. This chapter presents three experimental studies, providing technical solutions to assess its feasibility and demonstrate how the proposal can be applied. Furthermore, it also provides the threats to the validity of the experimental studies and a step-by-step guide for the implementation of the QUESTS methodology.

5.1 Introducing the experimental studies

In this thesis, the methodological choice was to employ real data whenever possible. Our focus is on vulnerable individuals living by themselves in smart environments. For this reason, data from users of ParkinsonCom[1] – a project conducted in our laboratory [176] – was employed across all three experimental studies to model the user agents, ensuring both diversity of representation and grounding in real-world information.

As outlined in the previous chapter, three assumptions were established as foundational

[1] https://parkinsoncom.eu/projet/parkinson/

to this work. The immediate goal was to assess the feasibility of these assumptions, which then would determine whether the proposed thesis could be pursued. To that end, we planned a feasibility validation strategy and conducted the experimental studies.

In the first experimental study, we sought to verify all three assumptions while applying the QUESTS methodology without the DM step (described in Section 4.2.3), and instead prepared the user agents to be guided by basic needs (introduced in Section 2.2.4) pre-established by us. The aim of this experimental study was (i) to determine whether QinU could be measured via simulation (*assumption 1*); (ii) to assess whether both users and smart environment devices could be effectively modeled as agents (*assumption 2*); and (iii) to evaluate the capability of the ABMS to interact with the software application under evaluation (*assumption 3*). To support this, we developed a prototype capable of receiving both implicit and explicit input (presented in Section 5.2.1). In essence, this first experimental study (further detailed in Section 5.4) served as a proof of concept to determine whether the QUESTS methodology could be operationalized.

From there, the next goal was to implement the full QUESTS approach as described in the previous chapter. To this end, we expanded the scope of the first experimental study by incorporating the DM techniques. This led to a second experimental study where we used a publicly available dataset to apply PM techniques, while continuing to use the prototype developed earlier. This experimental study is further detailed in Section 5.5.

Finally, we aimed at applying the QUESTS methodology to a third-party application, that is, a software that has not been developed with the specific purpose of taking part in our experimental studies. For that, we selected the ParkinsonCom application as the evaluation target. The design and results of this third experimental study are presented in Section 5.6.

Before detailing each experimental study, we first introduce the software applications that were the target of the QinU evaluations. The following sections will provide an in-depth account of each experimental study.

5.2 The evaluated software applications

This section presents both software applications that were the target of the QinU evaluation: the first one is an online application developed for a proof of concept (Section 5.2.1), while the second one is a third-party application (Section 5.2.2).

5.2.1 The Help Me! application

For a first instance of the application of this methodology, and to assess the feasibility of the approach, a simple software application was implemented. Aiming at evaluating explicit and implicit interactions of an application, a webpage was designed to be part of the proof of concept to test the assumptions presented in the previous chapter.

With that purpose in mind, the application was developed in .NET and C# to allow the end user to perform explicit and implicit interactions. The webpage is entitled "Help Me!" (Figure 5.1) and its purpose is to send out a help request made by the end user. It presents a text field to allow for a message to be included and a button for the help request to be sent. All occurrences are saved in a MySQL database set up in a Raspberry Pi. Its source code is publicly available on GitHub[2], within the "webpage" directory.

The explicit interaction refers to when an action is performed on purpose by the user to provide an input to the application. In this prototype, the user has the option to simply press a button, or to also include a message to accompany the help request. Once the button is clicked, an email is sent to the registered contact.

The implicit interaction occurs when a sensor notices a fall and inform the application. In this case, it results in the same outcome as the explicit interaction: if the user falls down to the ground, an email is sent with a default help request. This is possible because the application has a default value assigned to the gyroscope sensor, which remains in standby

[2] `https://github.com/mpangeloni/thesis/`

mode throughout its operation. When this sensor of the mobile device detects a fall (as it keeps detecting the angles of the device during the use of the application and it notices a sudden change), this parameter changes its value and triggers a help request.

This software has been designed for both real-world usage (when an actual person interacts with the system on mobile devices) and simulated usage (by external devices), such as when applying this methodology. For the latter, in this proof of concept, it is necessary for the ABMS to be prepared to interact with the targeted software application, and for the ABMS observer to know what to examine, as detailed in the upcoming sections.

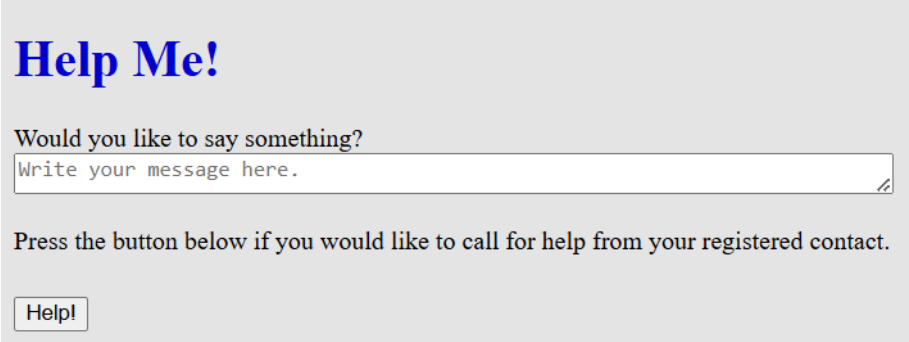


Figure 5.1: Screenshot of the webpage created for the proof of concept.

5.2.2 The ParkinsonCom application

To apply the QUESTS methodology in a third-party application (i.e., a software that has not been developed by us with the purpose of being used in a proof of concept), we applied the methodology on ParkinsonCom. This is an Android application created for people with Parkinson's disease and designed in our laboratory [177].

The ParkinsonCom project was developed with the support of the European regional development fund (Interreg V France-Wallonie-Vlaanderen) and the Agency for quality life AVIQ (l'Agence pour une Vie de Qualité) from Wallonia, Belgium. The application is available on the project's website[3] for free download and has accumulated over 1,852 downloads. The project involved people affected by Parkinson's disease from France and Belgium, and

[3] https://www.parkinsoncom.eu/ressources/application/

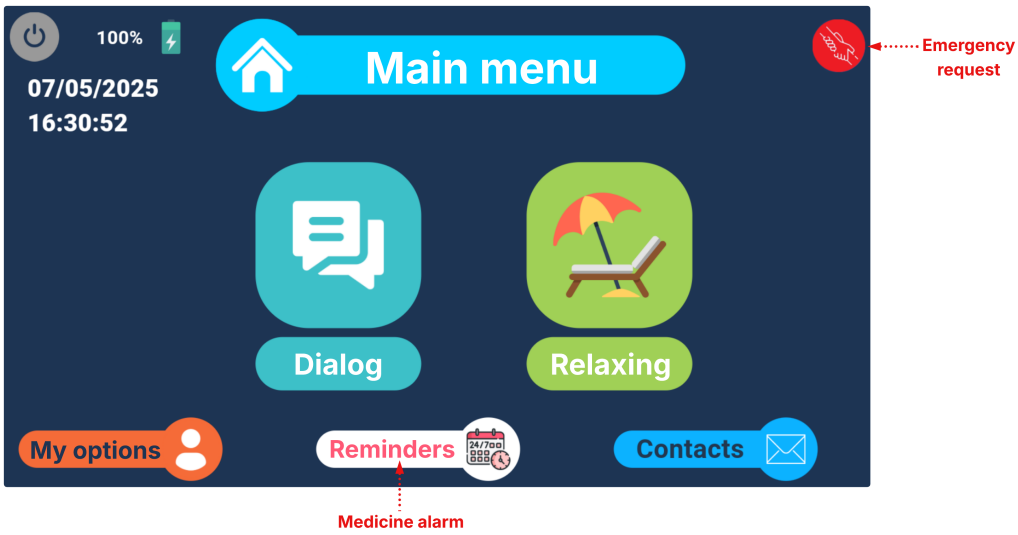


Figure 5.2: Screenshot of the ParkinsonCom application.

was motivated because the Parkinson's disease is the world's second most common neurodegenerative disorder, which impacts quality of life and social participation through both motor and non-motor symptoms [176].

To address these challenges, the project uses a participatory, user-centered design approach to create a communication support tool with people with Parkinson's and their caregivers. Interviews showed that participants struggle with speech, writing, and memory, leading to social strain and negative emotions. Although they rely mostly on paramedical support, some use technology (mainly messaging apps, social networks, or video calls) to stay connected, access information, or for entertainment, though motor difficulties often limit use. For that reason, the project's goal was to develop and validate a communication tool that builds on users' experiences and challenges.

Even though the main objective of the project was to ease the communication of people with Parkinson's, the application also provides other features, such as listening to the radio, watching YouTube videos, sending out emergency requests, setting up medicine alarms, and more. However, user evaluations have focused only on the communication features. Hence, the third experimental study (Section 5.6) will evaluate the *medicine alarm* feature (accessed by *Reminders* in Figure 5.2), as well as the *emergency request* feature. This request can be

triggered in two different ways: explicitly and implicitly. The explicit interaction requires that the user clicks on the red icon in the top right corner of the screen (see Figure 5.2), while the implicit one receives information from the accelerometer sensor of the mobile device. In the case of a sudden change in the accelerometer values, the application detects that the user fell and requests the help sending an email to the registered contacts (triggering the same outcome as when the user requests help on purpose). Once help is requested, the system initiates a loop that sequentially calls the registered contacts (up to four) until one comes, provides assistance, and confirms their arrival through the application. For the third experimental study (see Section 5.6), three contacts were registered. Due to the stochasticity of the simulation (see Section 5.6.1), any of these registered contacts could appear to answer the emergency request.

5.3 User profiles with Parkinson's disease

In all three experimental studies, the **user** is represented by one agent that plays the role of a person in the smart environment, moving around in the place and interacting with the software application under evaluation.

Regarding the questions suggested for the ABM designer to ask themself when building the ABM (in Section 4.2.1.1), we provide the following answers related to this implementation: **"Who will be the end users of the software application under evaluation?"** People with motor difficulties possibly caused by Parkinson's disease. **"What are the users' traits that could impact the usage of application?"** Parkinson's disease related symptoms. **"How frequent is the application used?"** The implicit interaction is more or less probable to happen according to the falling frequency parameter set up for the user agent, while the explicit interaction is impacted by the application use frequency parameter. More details about these settings are provided in the upcoming Section 5.5.5.

To define diverse user's profiles, we relied on interviews conducted for the Parkinson-

Com [178] project with twenty-six different individuals. To consider issues related to the Parkinson's disease, we selected some of the symptoms described in the Parkinson's Well-Being Map[4] [179], which are listed below, and modeled them as a set of variables to modulate the agent's behavior as listed below. Each parameter was defined based on a ParkinsonCom participant's response, thus reflecting an actual personal trait, and was set on a scale from 1 to 5 prior to the execution of each ABS (as detailed in the forthcoming Section 5.5.5).

- "I have slowness of movement (Bradykinesia)" – determines the slowness of movement with which the user agent should move around the smart home and perform actions, which will have an impact on the navigation speed when they are interacting with the software application;
- "I have shaking (tremor)" – sets a frequency of the tremors that the user may have: the stronger they are, the more likely it is for the user to not be able to properly use the application;
- "I fall over" – establishes a frequency with which the user agent may fall down.

Table 5.1 presents the parameters attributed to each user agent, which have been based in the responses provided by the real-world participants of the ParkinsonCom project. It includes their ID, the parameter for each one of the variables listed above and their nationality.

5.4 First experimental study: validating the assumptions for QUESTS

This first experimental study has been published as a proof of concept [180]. As it does not present the whole QUESTS methodology, we will dive into a more detailed description

[4] A guide co-designed by doctors and patients to facilitate describing difficulties and improving care.

Table 5.1: Parameters set up for each user agent (from P1 to P26).

ID	*slowness-of-movement*	*tremor-frequency*	*falling-frequency*	Nationality
P1	2	1	1	French
P2	3	2	3	French
P3	4	2	1	French
P4	4	5	1	French
P5	5	2	5	French
P6	3	1	4	French
P7	4	1	2	French
P8	4	3	3	French
P9	3	2	1	French
P10	5	3	5	French
P11	3	1	1	French
P12	4	1	2	Belgian
P13	4	5	3	Belgian
P14	1	4	1	Belgian
P15	3	4	2	Belgian
P16	5	4	4	Belgian
P17	3	4	2	Belgian
P18	5	1	3	Belgian
P19	5	3	5	Belgian
P20	5	1	4	Belgian
P21	4	1	3	Belgian
P22	4	2	3	Belgian
P23	3	2	2	Belgian
P24	3	1	2	Belgian
P25	4	5	3	Belgian
P26	4	1	3	Belgian

in the following section that describes the second experimental study, while presenting an overview of this first approach in this section.

The purpose of the ABMS is to simulate a person living alone in a smart home and using the Help Me! application presented in Section 5.2.1. In this first experimental study, we worked with a subset of the data from the ParkinsonCom project, using information from the eleven French end users to model the user agents.

For its implementation, we chose the widely used software NetLogo [74] that allows the user to model and simulate natural and social phenomena, and to easily explore the impacts of diverse configurations [181]. Furthermore, it effectively meets the visual rendering requirements that we consider essential for our model (see Section 2.2.3). NetLogo presents a graphical user interface (GUI) where it is possible to set up variables for the model and their parameters before every simulation, which is useful for the QinU analyst that uses

this methodology.

Figure 5.3 presents an overview of the implementation for this first experimental study with its three main phases.

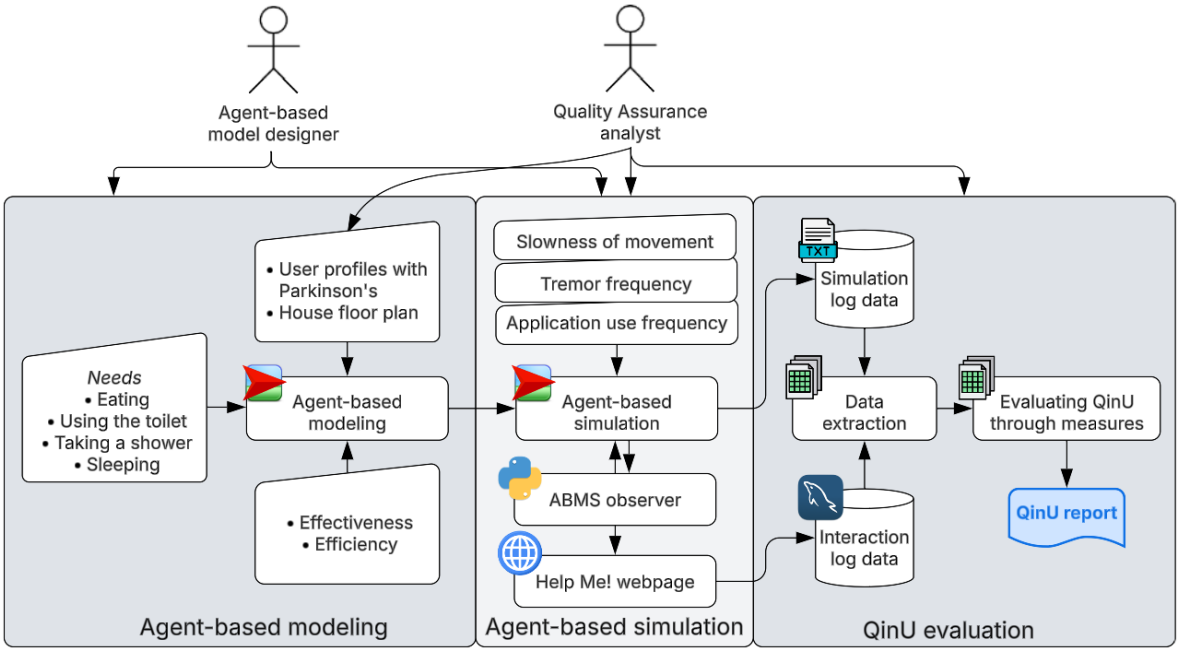


Figure 5.3: Implementation of the first experimental study.

5.4.1 Agent-based modeling

From the establishment of the target audience of the evaluated application, the ABM designer should set up variables for the traits of the user agents. These variables can represent what they may have in common, and need to be modeled in a way that impacts the application usage.

Regarding the *modeling of the users* in this first experimental study, the behavior of the user agents was guided by a group of fundamental needs that determine the tasks to be performed, which includes eating, going to the toilet, taking a shower, and going to sleep. Once the agent has fulfilled its most urgent need, it moves forward to accomplish the next task depending on its other needs. To that end, during the *modeling of the resources*, four sensors were positioned in strategic locations. The user would then move toward one of

them (kitchen, shower, toilet, or bed) depending on the activity to be performed. There are idle periods between the activities.

The parameters applied to the users' needs, e.g., the time duration for a meal, are present as constant values in the model and need to be established by the ABM designer. Apart from that, the variables presented in Table 5.2 are included in NetLogo in this first experimental study (and later expanded in the others, as presented in the forthcoming Section 5.5).

Table 5.2: Parameters of the simulation in the first experimental study.

Parameter	**Interpretation**	**Value**
slowness-of-movement	How slowly the user agent moves around: number of patches the agent can cross per tick.	[1..5]
tremor-frequency	Tremor frequency of the user agent: the higher the value, the more frequent the tremor.	[1..5]
application-use-frequency	Frequency of the application use, expressed as the number of hours between each use: the lower it is, the more often the application can be used.	[1..24]

Concerning the *modeling of the environment*, Figure 5.4 illustrates the smart home layout, designed using the house floor plan from a CASAS dataset [182]. Furthermore, regarding the *modeling of the tasks*, the user agent's surroundings at the moment they use the application can cause an impact: if they are in the bedroom or in the office and decide to request help, they are more likely to include message 2 ("I need help with my medicine"). Table 5.3 presents the probabilities of each message being added to the help request, varying according to where the user agent is. These probabilities are not grounded on real data and have been arbitrarily defined.

Table 5.3: Probability of each message being sent according to the user's location.

Message text	**Bedroom or office**	**Other places**
No message	0%	25%
"I have a headache"	25%	25%
"I need help with my medicine"	50%	25%
"I broke something"	25%	25%

The tools we used to develop the mediation between the ABMS and the "Help Me!"

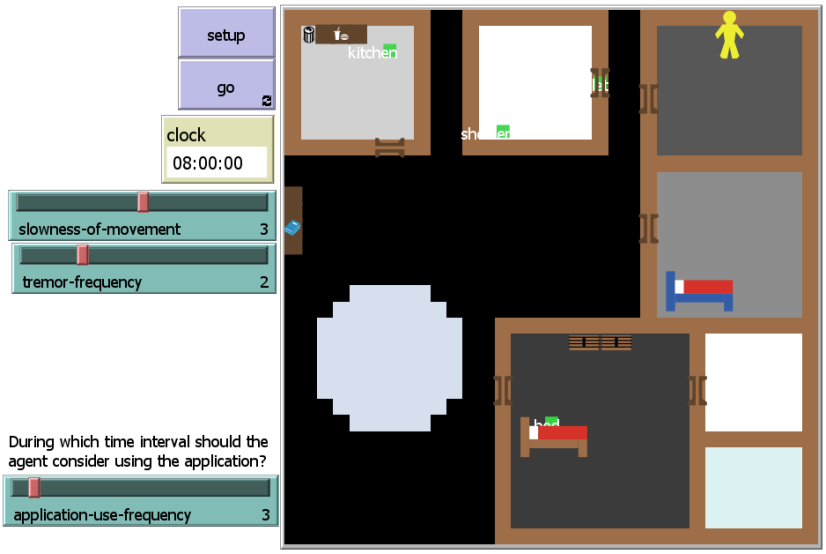


Figure 5.4: Implementation of the first experimental study in NetLogo.

application are based on Python. As presented in Figure 5.5, the observer sets up and starts the NetLogo model, and also launches the software application under evaluation. The simulation and the application become thus both active. The events occurring in the simulation cause changes in the variable states that are read by the observer, such as the agent's coordinates and the agent's actions (in this case, if they have decided to interact with the software application). Based on this information, the observer can launch potential related actions towards the application. For reasons of simplicity, we assumed the sensors placed in the simulated environment do not present errors, as this is not part of the scope of our study.

As mentioned in Section 4.2.2, usability is a popular measure to be applied when evaluating QinU. In this first experimental study, we selected the two QinU characteristics from usability for which we propose that simulation is feasible: effectiveness and efficiency. Since the focus of this first experimental study is merely to confirm whether it is possible to measure QinU characteristics using simulation, the details of how those measures were computed will be presented in the next study.

Table 5.4 presents the measures analyzed in this implementation. For effectiveness and efficiency, we specialized the generic measures (Ef-1-G and Ey-1-G) by considering the ex-

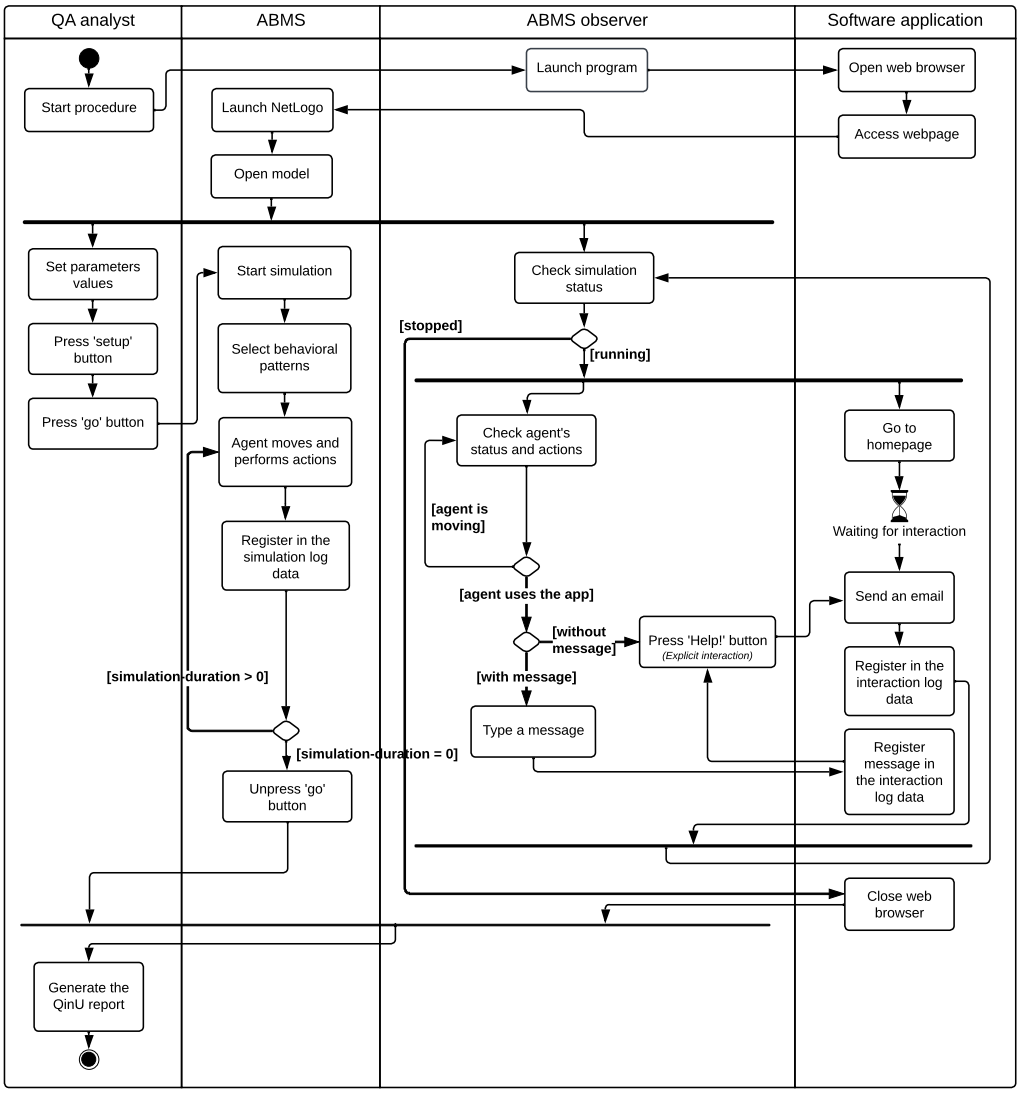


Figure 5.5: Activity diagram of the first experimental study.

Table 5.4: QinU measures applied in this study (based on the ISO/IEC 25022:2016 standard [22]).

ID	Name	Description	Measurement function
Ef-1-S	Tasks completed	The proportion of the explicit interactions to call for help that are completed correctly without assistance	$X = (A/B)$; A = number of unique tasks completed; B = total number of unique tasks attempted
Ey-1-S	Task time	The time taken to successfully complete the task to call for help	$Y = T$; T = task time

ecution of the task through explicit interactions only (that is when the user agent decided to call for help using the software application, taking into consideration if they were successful and how fast they could do it), renaming them Ef-1-S and Ey-1-S, respectively.

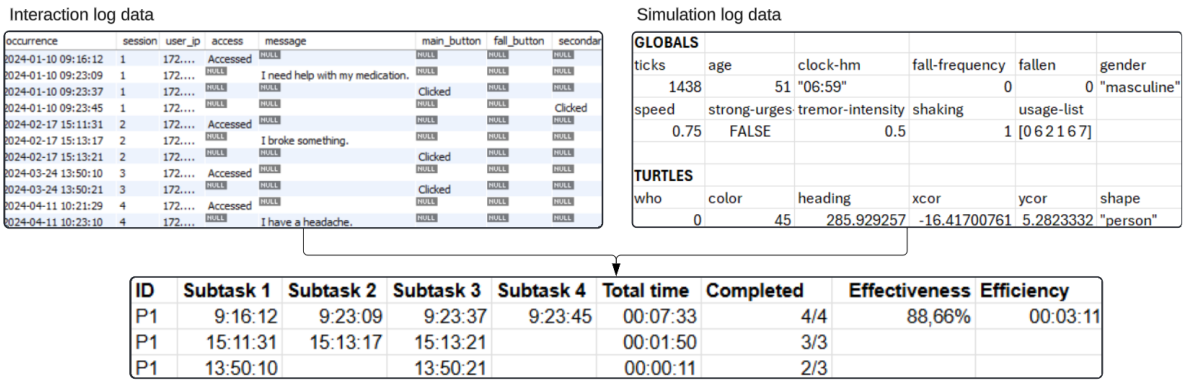


Figure 5.6: Interaction (top left) and simulation (top right) log data and the spreadsheet (bottom center) calculating the QinU measure with data that comes from them.

Based on the data collected (see Figure 5.6), the QinU report can be visualized with charts and explanations. Figure 5.7 shows an example of chart generated in the QinU report for the *explicit interaction*. The agents are represented by their identifiers, P1, P2, and so on. They are ordered (X axis) according to their tremor frequency, which can be seen below their identifiers. By analyzing this figure, it is possible to confirm that:

- the effectiveness is higher for people that do not present tremors. Consequently, the QA analyst can learn that the application is not very well adapted for people who suffer with this;
- regarding efficiency, the measures confirm that the slower the agent is in their move-

ments, the longer it takes for them to perform each subtask while using the software application (see, for instance, P7 and P8's orange bar); and,

- P8 and P9 have the same effectiveness, but since P8 has slower movements, they got a lower efficiency. Moreover, even though these two agents present a slight difference in their tremor frequency, both were able to achieve the same measure for effectiveness in the application, just as different people might conquer the same tasks with different degrees of disabilities.

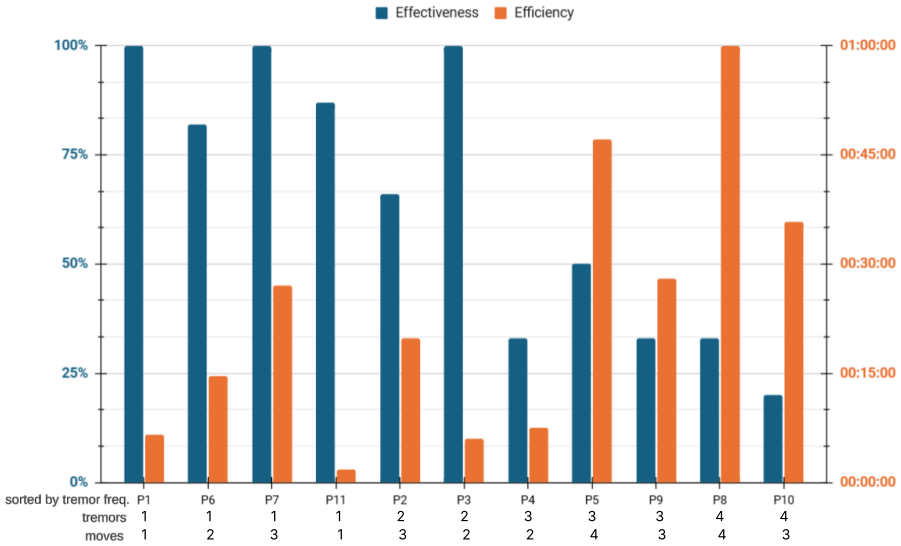


Figure 5.7: Measured effectiveness and efficiency of the application for each user.

This first experimental study confirmed, therefore, that it is possible to apply ABMS to evaluate certain QinU characteristics.

5.5 Second experimental study: implementing the whole QUESTS approach

The first experimental study served as a proof of concept and did not implement the full QUESTS methodology. This second study applies the complete methodology, as described

in the previous chapter, incorporating data mining techniques to provide agents with behavioral patterns grounded in real-world data.

Figure 5.8 presents the specifications and tools implemented in this experimental study. It presents the following key phases: *agent-based modeling* in NetLogo, with a *data mining* step that applies a semi-Markov model implemented in Python; *agent-based simulation* in NetLogo, with the ABMS observer component (also implemented in Python) that examines its executions and the targeted software application; and the *QinU evaluation* through the application of measures from the extraction of information from both the simulation and the interaction log data (from a text file and a MySQL database, respectively). Finally, a QinU report is provided for the QA analyst to assess software quality according to their expertise.

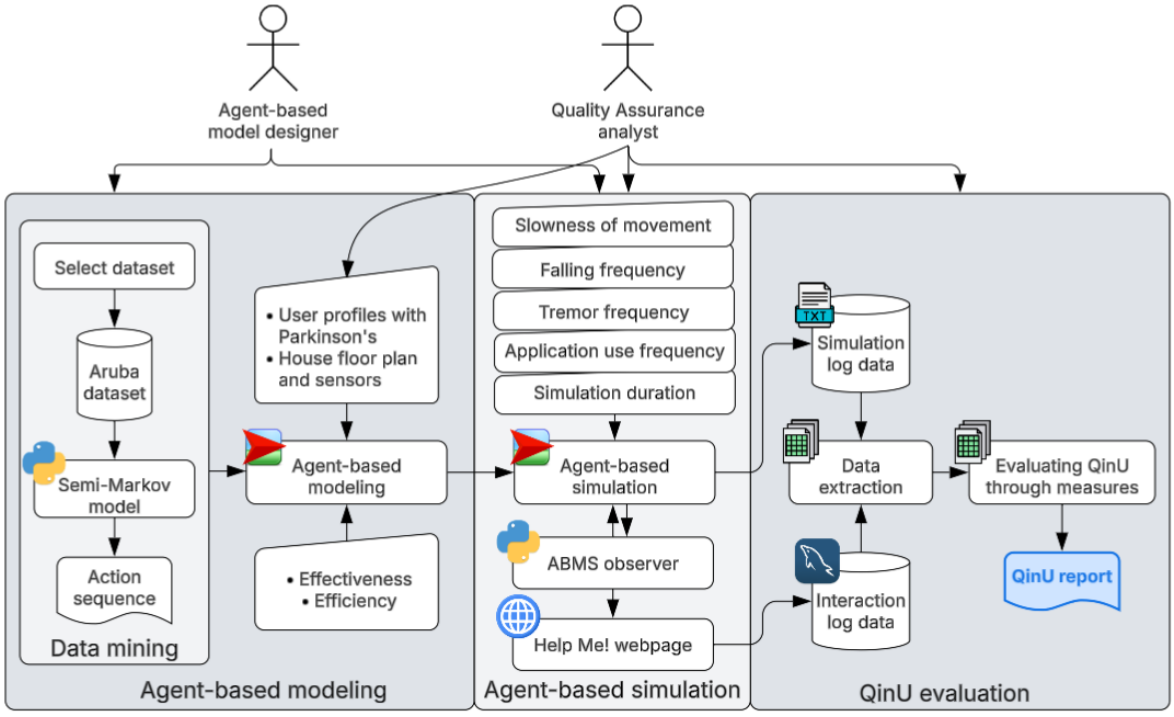


Figure 5.8: Overview of the second experimental study implementing the QUESTS methodology.

5.5.1 Agent-based modeling

Figure 5.9 presents the elements that are inputs in the implementation of our ABM. The variables of the context of use (from Figure 4.5 of Section 4.2) are represented here by the *user profiles with Parkinson's disease* (variables established according to possible traits of the users), and the *house floor plan* (the spatial environment of the simulation) with its distributed *sensors*. The selected *QinU characteristics to be evaluated* are further discussed in Section 5.5.4, and the implementation of *data mining* is detailed in Section 5.5.2.

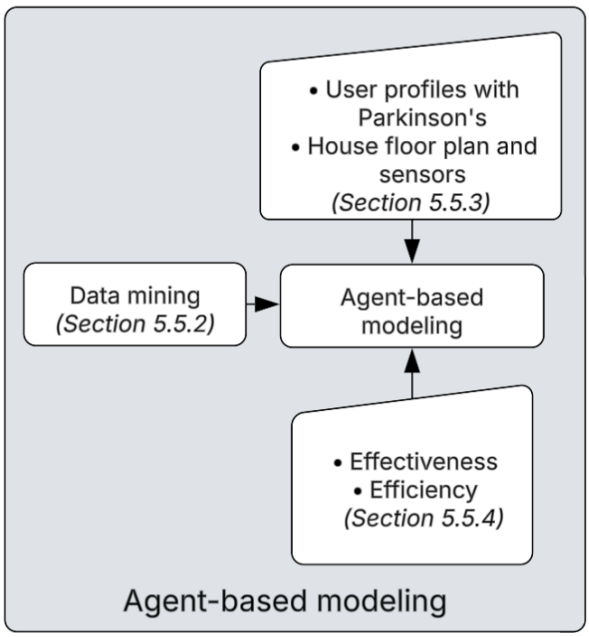


Figure 5.9: ABM phase and its inputs in this implementation.

5.5.2 Data mining

As outlined earlier in Section 4.2.3, we chose to use real-world data as input to the ABM. The output of the modeling phase must closely approximate real-world behavior, as the primary objective of grounding the ABM input in empirical data is to ensure realism. Following the selection of a suitable dataset, a standard data mining methodology was applied, comprising three key phases: **data preparation** (which includes both cleaning and understanding), **modeling** (involving the construction of a probabilistic model based on the original data

and the generation of artificial data) and **evaluation** (focused on validating the results obtained from the modeling phase).

The selection of the real-world dataset to ground the user agent behavior is based on the characteristics of individuals identified as potential users of the software application under evaluation. Therefore, we chose a publicly available dataset named Aruba from the CASAS repository [182] that contains data gathered from sensors located at the home of a volunteer older woman over the course of seven months (from November, 2010 to June, 2011). This dataset was selected because it contains data from a senior citizen living alone, which aligns with the scenario we aim to simulate.

The dataset has a total of 1,719,558 rows of data. The information comes from three different types of devices: motion sensors, which have their state set as either "ON" or "OFF" (according to whether or not there is movement within one meter from the device); door sensors, that detect when they are opened and closed; and lastly, temperature sensors, which capture the temperature in the room, and records a new input in the dataset every time it increases or decreases. The following section provides a detailed explanation of the data preparation process for the modeling phase.

5.5.2.1 Data preparation

As part of the **data cleaning** process, we conducted both outlier detection and sanity checks to ensure the consistency, reliability, and overall quality of the dataset prior to the modeling phase. The issues identified could be grouped into three main categories: errors in the "sensor state" of motion sensors; errors in the "sensor state" or in the "description" of temperature sensors; and lastly, errors in the "sensor ID". These faults are presented in Appendix C. To address them, we performed a scan and resolved the issue either by replacing the incorrect entry with the value we judged most plausible or by removing nonsensical data. These steps ensured that gaps or inconsistencies in the dataset did not compromise the robustness of the subsequent analyses.

With regard to **data understanding**, our efforts focused on two main directions: analyzing potential differences between sensor activations during daytime and nighttime, and identifying the underlying distributions of sensor activation durations.

Previous studies have emphasized the importance of validating simulation outcomes by accounting for distinctions between daytime and nighttime periods [105], as users' movement and activity patterns typically vary across these time frames. The generation of artificial data may target both day and night behaviors or focus solely on one period, based on the specific behavioral patterns relevant to the simulation and the objectives of the software application under evaluation. In this study, we opted to generate artificial data exclusively for the daytime interval (8:00 a.m. to 11:59 p.m.), as the software application under evaluation in this proof of concept relies on the user agent actively navigating the environment and interacting with the system – behavior that is less likely to occur during nighttime hours.

To identify the underlying probability distribution of sensor activation durations, a histogram was generated for each of the 31 motion sensors (as shown in Figure 6.1 in the Appendix). Subsequently, we applied statistical tests to evaluate conformity with several common distributions: Gaussian, chi-square, exponential, Weibull, and gamma (the latter two being generalizations of the exponential distribution). To assess normality, the Shapiro-Wilk test was conducted, which revealed that the data did not follow a Gaussian distribution. Following this, the Kolmogorov-Smirnov (KS) test was employed to examine the fit of the data to alternative distributions. In addition, Pearson's chi-squared test was used to further characterize the distribution properties. The results for each sensor and distribution type are summarized in Table 6.6 in the Appendix, which presents the corresponding p-values. With the sole exception of sensor M011 under the Weibull distribution, the results suggest rejecting the null hypothesis that the data samples were drawn from any of the tested distributions.

It is worth noting that the authors of [105] proposed modeling activity durations using

a mixture of gamma distributions. However, this approach appears unsuitable for modeling sensor activation durations of this dataset, as the histograms typically exhibit a single peak, with sensor M007 being a potential exception.

5.5.2.2 Data modeling phase

As previously stated, the objective of the data modeling phase is to generate realistic artificial data to serve as input for the ABM (see Figure 5.10). We focus exclusively on motion sensors, as the aim is to identify movement patterns. In Section 4.2.3, it was indicated that the output of this phase would be integrated into the ABM in the form of probabilistic models, thereby enabling the user agent to follow different combinations of behavioral patterns in each simulation run. This section begins by justifying the selection of a **semi-Markov model** as the chosen probabilistic framework. Then, we argue that once constructed, the model can be utilized in two ways: to generate **action sequences** (i.e., lists indicating the order in which motion sensors are activated) or to produce data dynamically ("on the fly") during the simulation phase, guiding the next action of the user agent in real time. The benefits and limitations of both approaches will be discussed later in this section.

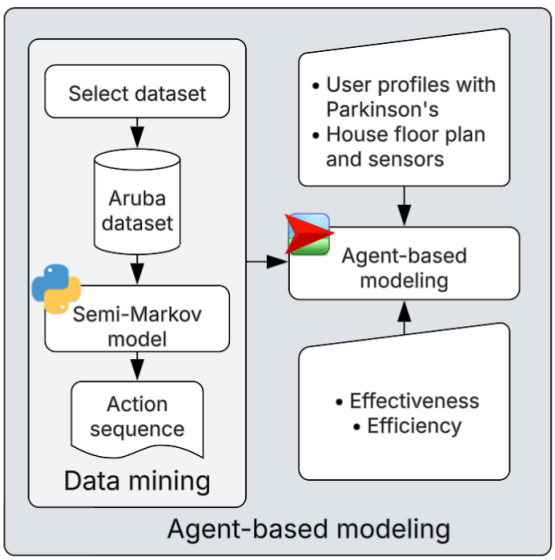


Figure 5.10: Detailed data mining phase during the ABM phase.

A Markov process is a stochastic process that satisfies the Markov property: the next

state in a sequence depends exclusively on the current state, with no influence from previous states [183]. Therefore, predictions on future outcomes can be made based only on its present state. In process mining, Markov models (and their adaptations) have been extensively used as mathematical framework for the movements of individuals through processes, where they enter and leave a given state (an activity of daily living, for example) at a particular time [105]. In the smart home's literature, many studies used Markov models [104, 184], and only a few considered semi-Markov models [185]. The difference between them consists of the underlying distribution of activity durations. Markov models assume that all activity durations are explicitly exponentially distributed (which may not be true in practice), while semi-Markov models can represent any kind of distribution.

Since the activation and deactivation of sensors can be recorded as time-stamped log events, Markov chains offer an effective manner for modeling state transitions probabilities [103]. Essentially, in our approach, we built a Markov chain to compute the probability of transitioning to the next state based on the previous state only, as in [186]. More precisely, each state is associated with a specific motion sensor. The transition probability from state A to state B is computed by normalizing the frequency of the (A, B) pair within the sequence of sensor activations, relative to all possible transitions originating from state A. Similarly, the initial state probability for a given state A is determined by calculating the proportion of sequences in which sensor A appears as the initial activation. Figure 5.11 presents a sample result focused on starting at Sensor M009 for the time interval between 8:00 a.m. and 8:05 a.m., aggregated across all available days in the dataset. The figure illustrates the data in two formats: (i) the total frequency of transitions between motion sensors, and (ii) the corresponding probabilities of each transition occurring between the selected sensors.

It is important to note that a standard Markov chain does not allow for modeling the duration each motion sensor remains active. To incorporate duration information, a Markov model can be employed if the underlying distribution of sensor durations follows an exponential pattern; otherwise, a semi-Markov model is required to accommodate more general

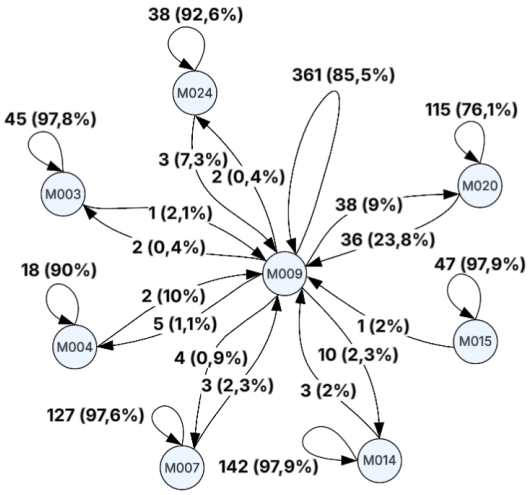


Figure 5.11: Markov chain modeling consecutive sensor activations starting at sensor M009 from the real dataset (#transitions and %).

distributions.

As discussed in the preceding subsection, we were unable to precisely identify the underlying distribution of sensor activation durations; however, we confirmed that it does not follow an exponential distribution. Consequently, our approach involves discretizing the set of possible duration values for each sensor (applying the ceiling function[5] to eliminate zero-duration values) and subsequently selecting one of these values at random using a uniform distribution whenever a duration needs to be generated for that sensor. This method ensures that durations occurring more frequently in the original data are more likely to be selected than those that were less common.

We conclude the description of the phase dedicated to constructing the probabilistic model with a final remark. Although the dataset includes instances where multiple motion sensors are simultaneously active, only the sequence of sensor activations was utilized for model construction. As a result, concurrent activations have no impact on the modeling process. By design, the semi-Markov model assumes that only one sensor is active at any given moment. However, this constraint does not limit the number of sensors that can be active during the simulation, as multiple sensors may be triggered simultaneously when a

[5]This function returns the smallest integer greater than or equal to x.

user agent is present within their respective detection ranges.

In our approach, a sequence of motion sensor activations is generated in advance, with transitions determined by the model's probabilities. Each activation is assigned a duration based on the probabilistic method described earlier, and a fixed time interval is maintained between successive activations. The resulting action sequence is then stored in a file for later use.

5.5.2.3 Evaluation of the outcome of the data modeling phase

The synthetic data generated should closely resemble the original dataset. To validate this, we conducted three types of evaluations.

First, we compared the transition matrix derived from the original dataset with that generated from an action sequence produced by our solution. Table 6.7 in the Appendix displays the matrix of absolute differences between the two (for a better readability, only non-zero values are listed). The observed differences were few and generally minor. Although this comparison was repeated multiple times using different action sequences, the results were consistent; hence, only one representative example is presented.

Second, we assessed the similarity of transition patterns for each sensor using Pearson's chi-squared test. The results yielded values equal to one across all sensors, indicating a high degree of similarity. Additionally, pairwise Kullback-Leibler divergence values were close to zero, further supporting this conclusion.

Third, we compared the average activation durations of each sensor between the original and the artificially generated datasets, observing similarly consistent results.

Finally, we conducted a visual inspection to verify the coherence of the probabilistic model. The coordinates of all motion sensors were mapped onto the floor plan image provided with the Aruba dataset [182] (see Figure 5.12), and a GIF was generated to visually represent the sequence of sensor activations. This visualization allowed us to examine the

spatial continuity of activations. While some sequences exhibited discontinuities in sensor activation paths, this phenomenon was also present in the original dataset. Nevertheless, during the simulation phase, the user agent is configured to activate all motion sensors it encounters along its path, thereby ensuring continuity in sensor activation.

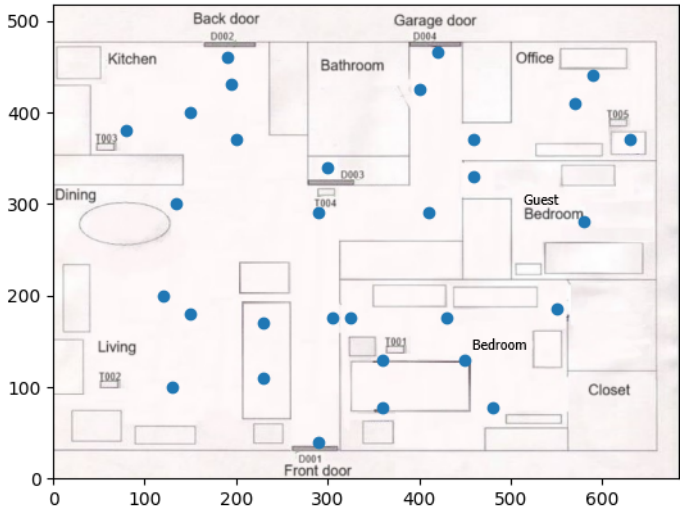


Figure 5.12: House floor plan with blue spots to represent all the motion sensors.

5.5.3 Modeling the context of use

As previously introduced in Section 4.2.1, the ABM includes different details regarding the context of use. All elements that may cause an impact in the application usage should be modeled. In this approach, each component of the context of use is represented by the following:

- ***users*** – the resident of the smart environment. It is limited to one person, as the purpose is to represent a person affected by symptoms of Parkinson's disease living alone;
- ***tasks*** – requesting help, whether because they willingly called for it (explicit interaction) or because they fell down (implicit interaction);
- ***environment*** – a one-level smart home;
- ***resources*** – motion sensors in the smart home and a mobile device to access the

application.

To promote a variability in the context of use, the variables of the ABM can be combined in different ways. The first three rows of Table 5.5 have been previously presented in Section 5.4, as they were already part of the first experimental study. The variables *slowness-of-movement* and *tremor-frequency* serve to modulate the user agents' behavior according to Parkinson symptoms, as well as the newly added *falling-frequency*. The *application-use-frequency* variable allows the QA analyst to set the frequency with which the user agent should consider using the application. In other words, it establishes a time in the simulation for the user agent to decide to interact or not with the application. The *simulation-duration* variable allows the QA analyst to set how many hours will be simulated in the ABMS.

Table 5.5: Parameters of the simulation.

Parameter	**Interpretation**	**Value**
slowness-of-movement	How slowly the user agent moves around: number of patches the agent can cross per tick.	[1..5]
tremor-frequency	Tremor frequency of the user agent: the higher the value, the more frequent the tremor.	[1..5]
application-use-frequency	Frequency of the application use, expressed as the number of hours between each use: the lower it is, the more often the application can be used.	[1..24]
falling-frequency	Falling frequency of the user agent: the higher the value, the more frequent the user falls down.	[1..5]
simulation-duration	Simulated duration in hours.	[1..24]

The following subsections will discuss each the modeling of each element of the context of use and how the model's variables impact them.

5.5.3.1 Modeling users

As previously introduced in Section 5.3, the modeling of the users has been based on data gathered during the ParkinsonCom project. This second experimental study executed simulations representing the eleven French participants (see Section 5.3).

5.5.3.2 Modeling tasks

The **tasks** represent the actions that the user performs to provide inputs to the software application under evaluation. In this proof of concept, this action is requesting help, and it can be performed in two different manners: explicitly and implicitly.

The explicit interaction occurs when the user presses a button to request help with the possibility of typing a message prior to it. Then, the system sends out an email to the registered contact. We can therefore say that this task is composed of the following subtasks: accessing the application (subtask 1); writing a message (subtask 2), which is not mandatory; and pressing a button to request the help of the registered contact (subtask 3). The possible outcomes are:

1. the user agent does not to use the application (hence, does not request help);
2. the user agent uses the application to request help and does not include a message;
3. the user agent uses the application to request help and includes a message. In this case, the content of such message comes as a list of three different possibilities ("I have a headache", "I need help with my medicine" and "I broke something"). The message is influenced by the location of the user at the moment (see Table 5.3 in the previous section); and,
4. the user agent tries to use the application to request someone's help but does not succeed. This last event is more likely to occur the higher the *tremor-frequency* parameter is, because users that present tremors in their hands may face difficulties to click on buttons as they wish.

The decision of using or not the application and how it is used is impacted by the *application-use-frequency*, the *tremor-frequency*, and the current **spatial situation** of the user agent. The *application-use-frequency* parameter establishes the frequency with which the user agent considers using the application within the hours of the simulation. The higher the *tremor-frequency* parameter is, the less likely the user agent is to succeed in us-

ing the application, as tremors cause users to have difficulties when pressing buttons (or trying to).

Moreover, when the decision is made to use the application, the interaction may or may not include typing a message, chosen from a fixed list ("I have a headache", "I need help with my medicine" and "I broke something"). If the user is in the bedroom or in the office when they use the application, then the "I need help with my medicine" message is twice more likely to be sent than the other two possible messages (see Table 5.3 in Section 5.4.1). This has been implemented to promote variability in the context of use and further emphasize that the environment can influence in the outcome of the interactions.

The implicit interaction occurs when the gyroscope sensor of the mobile device detects that the user has fallen down, so an automatic email is sent to the registered contact. To model this occurrence in the ABM, the shape of the user agent (normally in its default state) is changed to its "fallen" state. The change must be perceived by the sensors in the simulation and communicated to the application, causing a change in the parameter assigned to the gyroscope feature. This, then, triggers the sending of the email.

The probability of a fall happening or not in the simulation is impacted by two factors: how high is the *falling-frequency* parameter set for the user agent, and their spatial situation, i.e., where they are at the moment (further detailed in Table 5.6 from the following subsection).

As the scenario in the ABS represents a person in a possibly dangerous situation (considering they are requesting for help), and to represent IoT features, the room where the user agent is in at the moment that the help request is sent is included in the email. That is possible due to the motion sensors in place that notice their presence.

5.5.3.3 Modeling environment

The **environment**, in this proof of context, is defined by the technical and physical environments [23]. The technical is composed by the resources described in the upcoming Section 5.5.3.4, while the physical is represented by the structure of the simulated smart environment, and refers to the spatial situation of the user agent. This environment is modeled with walls, doors, and objects in it, such as beds and drawers (see Figure 5.13).

The layout of the smart environment in the ABM was configured based on the house floor plan accompanying the dataset used in the data mining phase (detailed in Section 5.5.2), specifically the Aruba dataset obtained from the CASAS website [182]. It reproduces the dimensions of the real house, as well as the placement of the sensors found in it.

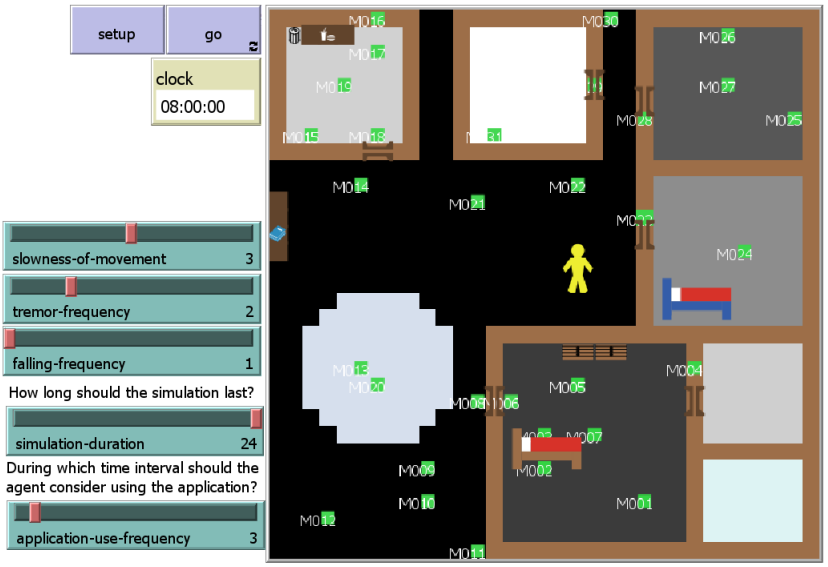


Figure 5.13: Smart environment modeled in the ABM.

We did not consider social, cultural and organizational environments [23], since they represent roles and relationships between people, organizational structures, and so on. As we are simulating a person living alone in a smart home, we considered that these elements would not apply to the reality we aim to reproduce. To establish if such environments could be represented within an ABM, another study should be conducted to focus on these factors, which are out of the scope for this thesis.

Each room is modeled to cause an impact in the events that will occur in the simulation. Other than the bedroom and the office impacting in which message is more probable to be sent (see Table 5.3 in Section 5.4.1), the bathroom and kitchen were designated as slippery areas, affecting the likelihood of the user agent falling. In this situation, the *falling-frequency* parameter is a factor just as important for such event to happen or not. Table 5.6 presents the chance of a fall happening within the simulation. These probabilities are not grounded on real data and have been arbitrarily defined (similarly as described in Section 5.4.1).

Table 5.6: Probability of falling according to *falling-frequency* and the user's location.

falling-frequency	**Bathroom or kitchen**	**Other places**
1	2%	1%
2	5%	2.5%
3	10%	5%
4	20%	10%
5	50%	25%

The fact that the targeted application is accessed through mobile devices implies that it can be utilized by the user agent anywhere in the smart environment. The different rooms promote changes in the context of use, as the environment is modified according to the user's surroundings. If the evaluated application was accessed through a device fixed in one place, such as a desktop computer, a thermostat or a smart refrigerator application, the user would always interact with the software in the same environment, which would not promote as many scenarios.

5.5.3.4 Modeling resources

To model the resources, the ABM designer can establish responses for the following inquiries: **"What are the smart devices and gadgets that the users will interact with to receive a response from the system?"** With a mobile device to access the application and with the motion sensors in place. **"Which resources are necessary for the users to interact with the targeted software application (such as Internet connection, mo-**

bile devices, etc.)?" A mobile device to access the application to explicitly interact with it, motion sensors to detect and inform the application for implicit interactions, and Internet connection for both cases.

The **resources**, in our implementation, include a mobile device that is supposed to grant the user agent access to the application under evaluation regardless of where they are at their home, and 31 motion sensors installed in the smart environment. The motion sensors were distributed throughout the simulated home according to the house floor plan provided with the Aruba dataset [182]. The motion sensors were modeled as agents, able to sense the occupant's position in the house. Whenever the user agent gets close to one of these sensors, they are switched "ON", which means their surroundings become red (to represent a light that goes on when something is detected) and such event is registered in the simulation log data, e.g. "sensor M001 ON" (a sample is provided in the upcoming Section 5.5.2). Once the user agent moves farther, the represented light goes off and an event is registered as well, e.g., "sensor M001 OFF". All sensors are assigned to a room, and for that reason, it is possible to know where the user agent is located at the moment (such information is then forwarded in the email that is sent to call for help).

Moreover, as we are evaluating an online application, the quality of the Internet connection can impact the use of the application under evaluation. Connection issues may spoil the application of the methodology and influence the final QinU report. This may occur because if the interactions are not transmitted to the application under evaluation, there may be information missing about the user agent's performance. This matter needs to be considered when running the experimental studies: given that the simulation transmits events to the real application via the network, we assume that as the duration of the simulation increases, the probability of encountering connection issues rises concomitantly.

The Internet connection may or may not be modeled as a resource inside the ABM. The investigators should decide the best approach for their implementation. In our approach, we opted to account for the real-world connection without modeling it within the ABM

itself, as we consider this approach brings a realistic aspect to the simulation. This choice is based on the understanding that residents in a smart environment are likely to experience connectivity problems at some point. Considering that the interaction takes place directly through the software application, and that a stable Internet connection is required for a help request to be successfully sent, we conducted a continuous 24-hour simulation to attempt to replicate such situations.

5.5.4 Modeling data to evaluate quality in use characteristics

The selection of the QinU characteristics that will be measured happens according to what the QA analyst plans to evaluate on the targeted software application. That will facilitate the identification of potential issues within the application, and to simulate multiple contexts of use without having to assemble individuals with the mentioned difficulties for testing purposes only.

In this second experimental study, we continue to focus on evaluating effectiveness and efficiency. Both characteristics are related to the use of the application itself and evaluate the performance of the explicit interactions.

5.5.5 Agent-based simulation

The ABS will put into action the dynamics from the entities modeled in the previous phase (Section 5.5.1). This takes into consideration the modeled variables, that are the parameters for the simulation (see Table 5.5). The ABM's source code is publicly available online on GitHub[6], within the "abms" directory.

Figure 5.14 shows the elements involved in the execution of the ABS, including the five established parameters discussed in this section. It also presents the two-way connection with the ABMS observer (forthcoming Section 5.5.6), that sends the relevant simulated

[6] `https://github.com/mpangeloni/thesis/`

occurrences to the application under evaluation (the “Help Me!” webpage, previously described in Section 5.2.1).

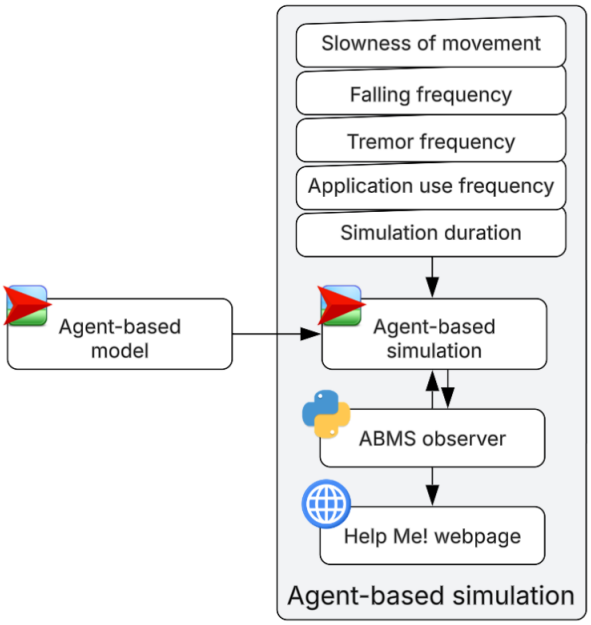


Figure 5.14: ABS phase and its inputs in this implementation.

Moreover, the following subsection presents the detailed implementation using a protocol to present the designed ABMS. The adoption of a protocol to describe an agent-based model is recommended to ensure that the information is provided in a clear manner, and to overcome possible issues found by other researchers in what has been called a “replication crisis” [84].

5.5.6 Agent-based model and simulation observer

The ABMS observer is a component that serves as a mediator between the ABMS and the application under evaluation. In this implementation, we developed it using tools based on Python, which is “often used as a ‘glue’ language, meaning that it connects pieces of software written in different languages together” [187].

To create the link between the ABMS (in NetLogo) and the targeted application (the

webpage), we used the pyNetLogo[7] library to enable the control of NetLogo from a Python script. Hence, pyNetLogo is used to transmit commands from the ABMS observer to NetLogo (set up the parameters and execute the simulation). From there, the observer analyzes the occurrences and checks the agent's state in the simulation. To access and interact with the webpage, we used the Selenium[8] library, which allows for web browser navigation through Python scripts. Moreover, the ABMS observer should inform the data mining solution about the total time duration selected for the simulation, in order that it generates the coordinates and the time duration for the motion sensors to be on. This will guide the user agent throughout the events. The ABMS observer's source code is publicly available on GitHub[9], in the "abms-observer" directory.

The activity of the ABMS observer and the interactions between the different components and actors of the simulation are represented in the activity diagram seen in Figure 5.15. The QA analyst is responsible for starting the whole process through the launch of the ABMS observer. The following points listed below occur during this implementation:

- the ABMS observer opens the web browser and accesses the webpage. Then, it opens the ABMS in NetLogo. The QA analyst is supposed to parametize the variables before executing the simulation. Once this is done, a parallel process begins as any relevant pre-established occurrence in the ABS should be reported to the webpage;
- from that point on, the user agent moves around following the behavior patterns from the data mining implementation. This action sequence (provided from the ABMS observer to the ABMS) is generated according to the *simulation-duration* parameter, which calls the process mining procedure, to create a text file on the spot. The motion sensors (the device agents in the ABMS) switch "ON" and "OFF" accordingly; The events are registered in the simulation log data;
- in the meantime, as the ABMS observer checks the user agent's state and actions

[7] https://pynetlogo.readthedocs.io
[8] https://selenium-python.readthedocs.io
[9] https://github.com/mpangeloni/thesis/

(established by the generating number procedure) – i.e., if they have fallen (implicit interaction) or if they use the application (explicit interaction) – it engages with the webpage that awaits any interaction. When it happens, an email is sent (with or without an added message) and these events are saved in the interaction log data;

- if the time has not run out, the ABMS observer continues to check the simulation occurrences, inspecting the ABMS for any possible interaction with the webpage again (as it has gone back to its homepage for any new event). However, once the established time is over, the simulation is stopped and the web browser is closed, bringing the process to an end;
- from there, both the simulation and the interaction log data are ready to be merged and examined by the QA analyst.

5.5.7 Executing the agent-based simulation

As previously mentioned in Section 5.3, we relied on interviews conducted for the ParkinsonCom [178] project to represent different and real people. We adapted their responses to set up parameters for the modeled variables, as detailed in Table 5.1, and therefore, set up eleven different user agents to represent these potential end users. The simulation started at 8 a.m. (according to the simulation clock). A text file with coordinates is generated on the fly, based on the *simulation-duration* parameter, by executing the process mining function developed for this purpose.

The relevant events from the simulation are saved in the *simulation log data* with their simulation timestamp. The simulation log data for the eleven executions are available publicly on GitHub[10], in the "simulation-log-data" directory.

Table 5.7 presents an extract of the simulation log data for P1 with such occurrences. The first column always presents the timestamp for the simulation clock, while the second

[10] `https://github.com/mpangeloni/thesis/`

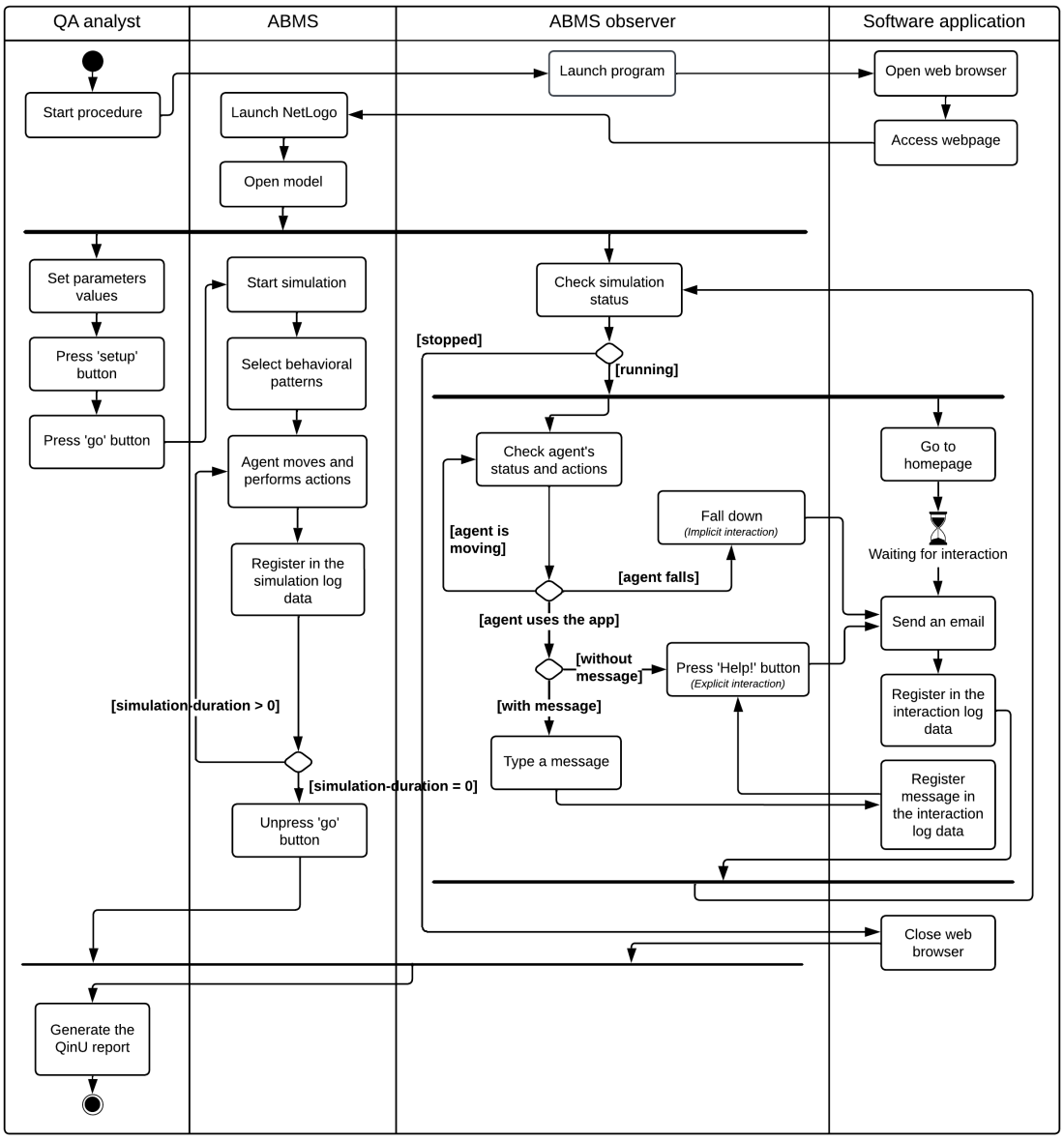


Figure 5.15: Activity diagram for the second experimental study.

Table 5.7: Sample of simulation log data from NetLogo for user agent P1.

time in the simulation	*event*
08:00:01	sensor M006 ON
08:00:46	sensor M008 ON
08:00:47	room_changing bedroom corridor
08:01:13	sensor M006 OFF
08:01:16	sensor M020 ON
08:01:17	sensor M008 OFF
...	...
08:31:01	I fell in the bathroom.
...	...
11:00:00	random_number: 7 [0..9] [message: I broke something.]
...	...
11:32:01	I fell in the living room.
...	...
20:00:00	random_number: 5 [0..9] [message: I have a headache.]
...	...
05:00:00	random_number: 9 [0..9] [could not use the application.]
...	...
17:00:00	random_number: 6 [0..9] [message: I need help with my medicine.]
...	...
08:00:00	random_number: 6 [0..9] [message: I need help with my medicine.]
...	...
14:00:00	random_number: 5 [0..9] [message: I have a headache.]
...	...

column shows the relevant events we chose to be printed, such as a sensor being switched on or off, going from one room to the other, generating a random number to trigger (or not) an explicit interaction within the software application, falling down to the ground, and where the occurrence happened.

User agent representing person one (P1) fell down twice and executed six times the help request feature by an explicit request. We point out that:

- the two first rows registered two motion sensors switched on at first, which means that the user agent approached them;
- then (third row), *room_changing* notifies that the user agent left the bedroom and

moved to the corridor;

- in the next line, a motion sensor that was previously switched on (M006) is now switched off as the user agent moves farther from it;
- at time 08:31:01, the user agent fell down on the bathroom floor (first red line) – this event must be read by the ABMS observer, which then should trigger an action in the software application for someone to be called for help (representing an implicit interaction);
- at time 11:00:00, it shows an explicit interaction (first blue line) displaying the random number generated (number 7), its range [0..9] and the message included ("I broke something"). The value of this random number determines whether the agent uses the application or not, and the message that is sent, if any. Note that at 05:00:00, the selected random number (number 9) establishes that the agent was not able to use the application.

The data generated by the interaction with the application include the real time when the "Help Me!" application is being used, which is different from the time inside the simulation. This data has been recorded in a MySQL database using a Raspberry Pi. The *interaction log data* for the eleven executions are publicly available on GitHub[11], in the "interaction-log-data" directory.

Table 5.8 contains an extract of the data generated by the interaction with the software application for the same user agent (P1) corresponding to one implicit interaction and two out of six explicit interactions. Column *real time* presents the real-world timestamps from when the events occur; *session_id* is the session identifier, which changes every time the homepage is accessed; *user_ip* provides the IP of the user. Each interaction has its own specific row in the database, which means that columns *access*, *message*, *button*, and *fell*, only register information in the database about the event related to that specific interaction. Column *access* refers to the moment the user enters the webpage; *message* presents the input

[11] `https://github.com/mpangeloni/thesis/`

from the text box; *button* for when the button on the homepage is clicked sending a help request to the registered contact; and *fell* when the user falls down, in other words, when the gyroscope sensor of the device notices a sudden change.

Table 5.8: Sample of interaction log data recorded in the Raspberry for user agent P1.

real time	*session_id*	*user_ip*	*access*	*message*	*button*	*fell*
...	...	...	...	...	...	...
2024-11-0115:13:32	2	10.4.1...	null	null	null	Fell
2024-11-0116:09:12	3	10.4.1...	Accessed	null	null	null
2024-11-0116:12:24	3	10.4.1...	null	I have a headache	null	null
2024-11-0116:12:29	3	10.4.1...	null	null	Clicked	null
2024-11-0118:43:01	4	10.4.1...	Accessed	null	null	null
...	...	...	...	...	...	...

Table 5.8 presents only an excerpt of the interaction log data for P1 (see the GitHub repository for all log data). We point out the following:

- the first implicit interaction presented in Table 5.7 does not appear in the interaction log data. That means the software application did not register such event;
- the second implicit interaction in Table 5.7 (also presented in red in Table 5.8) is the only implicit interaction properly registered in the interaction log data for P1 (at 15:13:32);
- the rows in blue present an explicit interaction and its sequence of three subtasks starting at 16:09:12: (1) the user agent accessed the webpage, (2) included a message, and (3) called for the registered contact. This interaction was triggered by the simulation event in the second blue row of Table 5.7;
- the last row in this sample presents another explicit interaction (triggered by the third blue row of Table 5.7) where, although the application was accessed, the user agent could not use the application.

As previously mentioned, we decided to evaluate the QinU for the "Help Me!" application considering the effectiveness and efficiency quality characteristics. To that end, the effectiveness and efficiency measures presented in Table 2.1 (Ef-1-G and Ey-1-G) were spe-

cialized considering the task of the software application to be evaluated, which is requesting help.

Considering the measures for effectiveness and efficiency presented in the previous experimental study (see Table 5.4), the data from the simulation (Table 5.7) and from the interaction with the real application (Table 5.8) are merged to generate a spreadsheet file. This file includes the data necessary to calculate the measures, as exemplified in Table 5.9, with an excerpt from P1's data. Each line of the spreadsheet corresponds to one execution of the "Help Me!" application: either when the user agent explicitly requested help by clicking on the button or falling down (implicit interaction). In the case of an explicit request, columns 2 (Subtask 1) to 6 (Ef-1-S) will be filled in with the data from previous tables and the other columns will be empty. For the implicit interaction (the falling event), columns 2 to 6 will be empty and the two last columns (Fell) should be filled in.

Table 5.9: Outcome from the QinU measures for P1.

ID	Subtask 1	Subtask 2	Subtask 3	Ey-1-S	Ef-1-S	Fell (sim.)	Fell (app.)
P1						1	
P1	15:11:31	15:13:17	15:13:21	00:01:50	1		
P1						1	1
P1	16:09:12	16:12:24	16:12:29	00:03:17	1		
P1	18:43:01				0.33		
P1	21:31:07	21:31:59	21:32:11	00:01:04	1		
P1	07:27:59	07:28:11	07:28:15	00:00:16	1		
P1	13:52:28	13:52:42	13:52:52	00:00:22	1		

The spreadsheet file contains, therefore, the following data:

- the *User ID* identifies the simulated user among the eleven user profiles represented;
- the *Subtask X* columns relate to the three subtasks previously defined for the call for help by an explicit request;
- the *Ey-1-S* value measures the total task time computed as the final time minus the starting time of the explicit interaction, providing how efficient this task performance was;
- the *Ef-1-S* value measures how effective this task is by considering how many sub-

tasks were completed;

- the columns *Fell (sim.)* and *Fell (app.)* concern the fall events registered in the simulation and in the application, respectively. Once a fall is detected, it is supposed to include "1" to the corresponding cell of each log data. If *Fell (sim.)* presents some content, so should *Fell (app.)*, to inform that the fall event has been properly registered in both, and hence, that the implicit interaction triggered an event on the software application. Otherwise, it means that even though the user agent fell down (what can be seen in the simulation log data), the software application did not register the occurrence and did not trigger any action to call for help. Then the number of falls in the simulation log data is divided by the number of falls in the interaction log data to calculate the RHe-new value.

Table 5.9 presents all events of P1 (two implicit and six explicit interactions). We observe the following:

- for the explicit interaction, P1 performed all subtasks (Ef-1-S = 1) in five executions. Only in one of the executions, P1 accessed the application but could not use it (this corresponds to the third blue row in Table 5.7 and the last row in Table 5.8 (Ef-1-S = 1/3 = 0.33);
- for the two implicit interactions (falling down), the first one was simulated but not properly executed in the software application, as opposed to the second one (which corresponds to the red rows in Tables 5.7 and 5.8, respectively).

5.5.8 Quality in use through measures

The QinU measures are then calculated from the collected and merged data. Auxiliary scripts were employed to extract information from the MySQL database (interaction log data) and to retrieve relevant content from the simulation log data. These scripts are pub-

licly available on GitHub[12], within the "auxiliary" directory. The necessary information from both was then matched for calculation and uploaded in a Google Sheet for posterior design of charts for better visualization of the results. Figure 5.16 illustrates which tools are applied in this final phase.

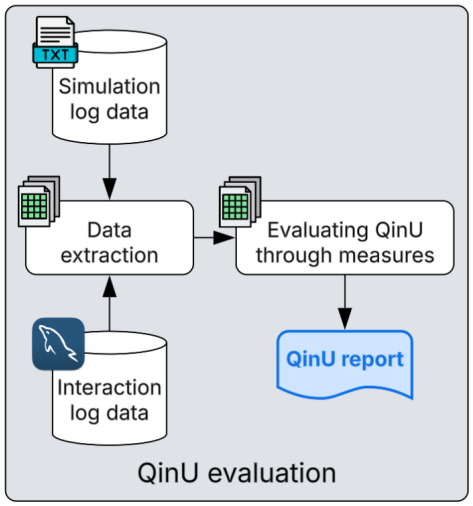


Figure 5.16: QinU evaluation and its components.

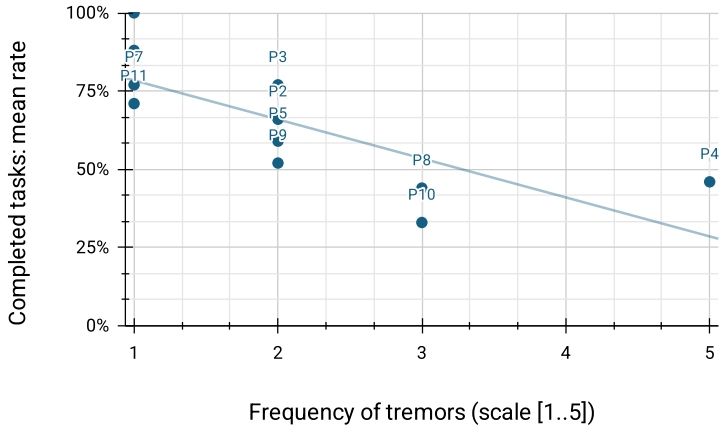
For the two measures concerned by this implementation, the calculation has been performed as follows:

- for the effectiveness evaluation, a mean of the completed tasks is calculated on top of each one of the values found in the *Ef-1-S* column for each user agent. The mean of the tasks completed of the evaluated software application is **0.89**;
- for the efficiency evaluation, a mean task time is calculated among all the values found in the *Ef-1-S* column for each user agent. The mean task time for P1 is **1 minute and 29 seconds**.

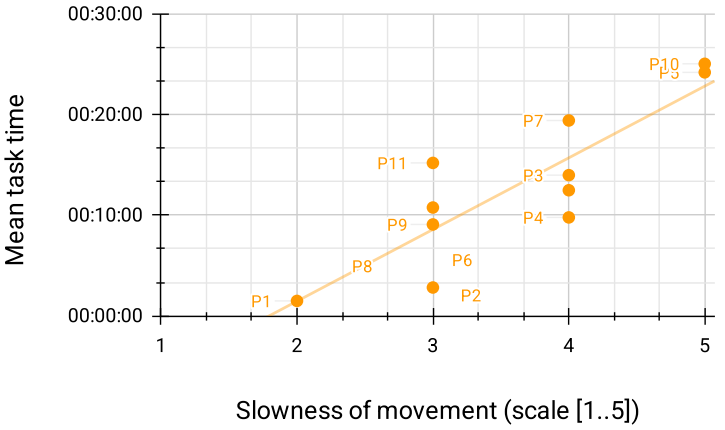
The QA analyst decides, according to established thresholds (usually defined by domain experts), whether the results are good enough or not for the targeted software application. To support this decision, the QinU report may provide visual representations of the results in the form of charts like the ones shown in Figure 5.17. We chose to present the two QinU measures in relation to the parameters that impact them the most. Note that the tests have

[12] `https://github.com/mpangeloni/thesis/`

Figure 5.17: Charts presenting the results of the evaluated QinU characteristics for eleven simulated individuals.



(a) Effectiveness evaluation



(b) Efficiency evaluation

been performed with the eleven different user profiles mentioned before and the means are calculated on 24-hour simulation data.

As illustrated in Figure 5.17, user profiles indeed influence the interactions of user agents with the application under evaluation.

Figure 5.17a presents a chart with the effectiveness evaluation for each one of the user agents according to the frequency of tremors. Since *mean tasks completed* and *frequency of tremors* are well correlated ($R^2 = 0.564$), we can confirm that the effectiveness when using the evaluated software application is higher for people that present smaller frequencies

of tremors. By analyzing this statistics, the QA analyst can notice that the product under evaluation is not very well adapted for people who suffer with this issue.

Figure 5.17b provides a chart with the *mean task time* for the efficiency evaluation according to the *slowness of the movements*. Since the variable and the outcome have a strong correlation ($R^2 = 0.742$), we can confirm that the slower the agent is in their movements, the longer it takes to complete each subtask while using the software application. The correlations between the results and the parameters of each user agent are not surprising, as the model incorporates probabilities that naturally drive the outcomes in this direction.

With this second experimental study it has been possible to confirm that data mining techniques can be employed to provide real-world patterns for the agents' behavior. In the following section, we conduct a third experimental study, applied to a third-party application rather than to one specifically developed for the purposes of the methodology.

5.6 Third experimental study: applying QUESTS on a third-party application

To apply the QUESTS methodology in a third-party application, the targeted software for this experimental study [188] was ParkinsonCom, previously introduced in Section 5.2.2. The application was executed on an Android emulator in order for it to receive input from the ABMS through the ABMS observer.

As previously introduced in Section 5.2.2, we chose to evaluate two features of the application: (i) requesting help and (ii) using the medicine alarm. Feature (ii) allows the user to set up different alarms for the times when they need to take their medicine. The user is supposed to shut it off only after taking their assigned remedy for that time. As it is known, Parkinson's patients are usually required to take various medication. Because of that, we established that the alarm would go off five different times to represent the medicine that

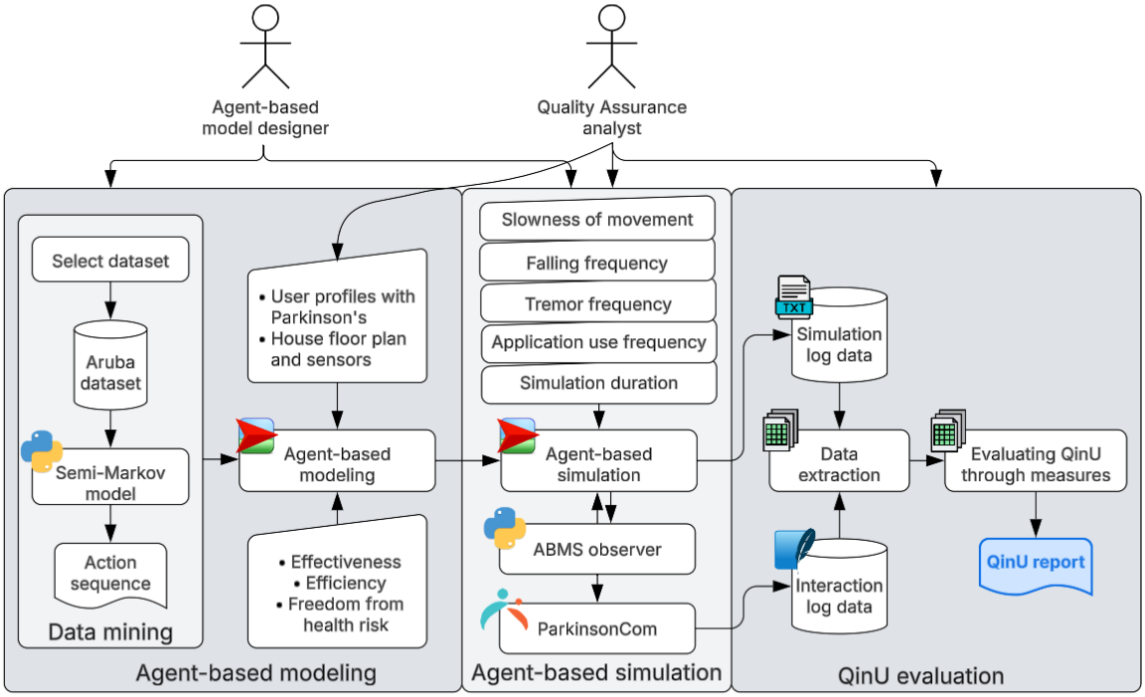


Figure 5.18: Overview of the second experimental study implementing the QUESTS methodology.

they tend to need the most – COMT suppressant, MAO suppressant, Amantadina, Levodopa, Pramipexol.

Furthermore, in this third experimental study we used the whole set of information from the answers provided by ParkinsonCom project participants – the 26 people from both France and Belgium (see Table 5.1), as opposed to the first two experimental studies that used only information from the eleven French participants.

Since we are testing a real application (rather than a proof of concept like the "Help Me!" application), and considering that most Parkinson's patients are elderly and may face health risks if the system fails, we set quality requirements not only for effectiveness and efficiency but also for freedom from health risk[13]. In contrast to the previous experimental study, effectiveness in this study incorporates both explicit and implicit interactions. Efficiency is also calculated as the mean of the two evaluated features: the help request and the medicine alarm.

[13]The emergency request feature is critical for reducing risks to people with Parkinson's and therefore operates in a loop until a registered contact confirms their arrival to provide help, as explained in Section 5.2.2.

The generation of the user agent's behavior takes into consideration the user's profile, the environment around them, and the outcome. Listed below are described the principles behind possible behaviors: the moving from one spatial position to another, potential falls and potential uses of the application. These statements below depend on the simulation parameters and include stochasticity (see Figure 5.19):

- the generation of the moving behavior depends on the coordinates defined by the action sequence from the behavior patterns and the *slowness-of-movement* of the user agent;
- the generation of falls depends on the *falling-frequency* and also on the current **spatial situation** of the user agent. The higher this parameter is, the higher the chances of the user agent to fall down. Whatever the frequency is, the chances double when the user agent is in the bathroom or the kitchen, places established as more slippery than the others (see Table 5.6 in Section 5.5.3.3).

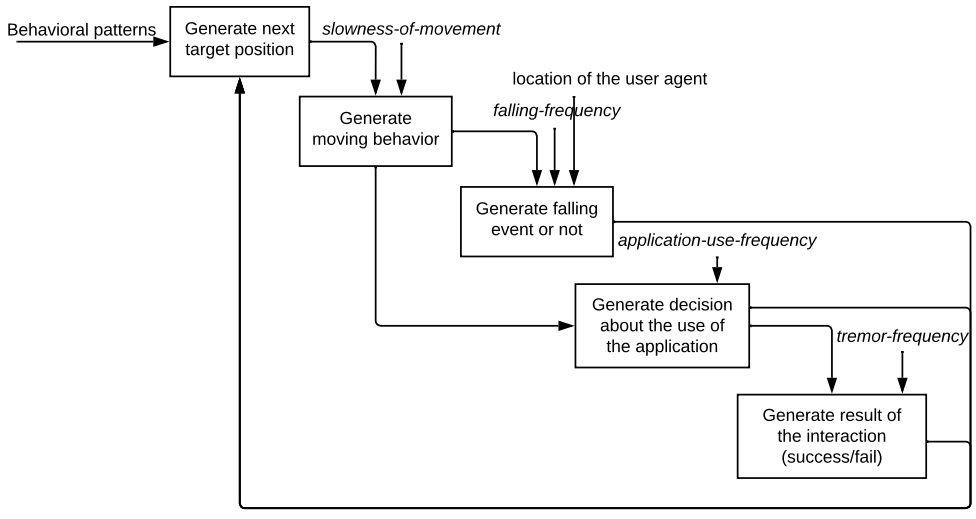


Figure 5.19: Main steps to generate the behavior of the user agents.

5.6.1 The Overview, Design concepts and Details protocol

To facilitate the recreation of this experimental study by other researchers, the following paragraphs describe the ABMS built for this study according to the ODD protocol introduced in Section 2.2.3. Furthermore, a classical template of the ODD protocol is presented in Appendix B.

Purpose and patterns. The purpose of this ABMS is to simulate a person living by themself in a smart environment, and interacting with smart devices around them. The events are supposed to be perceived by an ABMS observer that then transmits the occurrences in the form of inputs to an external application that is under QinU evaluation.

Entities, state variables, and scales. The model includes two types of agents: user agents, which represents the end user of an application in a smart environment (which, in this ABM, can be the resident of the smart home or the contact called through the emergency request), and device agents, which are 31 motion sensors. The user agent's profile is a combination of static parameters: *slowness-of-movement*, *tremor-frequency*, and *falling-frequency*. The possible states of the user agent are either "fallen" or "standing". The states of the device agents represent their activation/deactivation state ("ON" or "OFF"), and this depends on the distance from the user agent at that moment. The dimension of the environment and the configuration of the rooms reproduce the house's floor plan provided with the Aruba dataset [182], which was selected for implementing the process mining technique used to generate behavioral patterns for the agents. The environment serves as the spatial configuration in which the user agent moves.

Process overview and scheduling. The simulation time proceeds in discrete steps, where one time step represents one second in real time. The amount of hours that should be simulated is set up through the *simulation-duration* parameter in the GUI. When the simulation is finished, the ABMS provides a simulation log data with relevant events that happened throughout (see Figure 5.20).

Basic principles. The main idea of the ABMS is to simulate as many contexts of use as possible, following the QA analyst's plans. This approach avoids placing actual people in a smart environment solely for testing purposes during the first assessment of the QinU. Instead, software agents are the responsible ones to trigger interactions with the evaluated software application, creating enough artificial data for an evaluation, providing then supporting material for QA analysts to make evaluate software applications.

The user agent's moving decision is made according to the coordinates provided by the process mining technique (detailed in Section 5.5.2). As for the interactions that happen with the software application, the ABMS observer interacts with the software application on behalf of the user agent.

Sensing. The device agents have a fixed size sensing radius for noticing the proximity of the user agent, i.e., *set sensing-radius (3 + size / 2)*. Furthermore, there is the falling detection, which represents the accelerometer of the device in which the application is executed.

Interaction. Interaction between agents in this model happen when the user agent moves around the simulated smart environment, and their proximity is perceived by the device agents (31 motion sensors). These sensors react when the user agent moves close to them by switching on, and when they move farther, switching off. When the device agent is "ON", its surroundings become red to represent the device lighting up.

Other than the interaction between the user agent and the device agents previously described, this ABM also creates a second user agent (*hatches* in NetLogo) to inform the application that help has arrived. After that, this user agent leaves (*dies* in NetLogo).

Stochasticity. The events of falling and using the software application are stochastic. The probability of falling is influenced by the *falling-frequency* parameter: the higher its value, the more likely it is for the user agent to fall down to the ground. On top of that, the spatial situation also influences the possible occurrence of this event: using common sense knowledge, we added a slippery level attribute to the different rooms that compose

the apartment. That way, if the user agent is either in the bathroom or the kitchen, they are twice as likely to fall down. Table 5.6 in Section 5.5.3.3 represents such probabilities.

The probability of successfully using the application depends on the *tremor-frequency* parameter. In our approach, we allow the user agent to not use the application at all. But, if the user agent does decide to use it, their rate of success decreases with the increase of their tremor level. The higher the frequency of tremors, the more likely they are to be unsuccessful in their endeavor.

The stochasticity in this ABM also occurs according to where the user agent is at the moment that the medicine alarm goes off. From that moment, wherever the user agent is, they will move to the bathroom – designated place to take the medicine. The slower the user agent is, the longer it will take for them to get there.

Furthermore, stochasticity also arises when a user agent calls for help and one of the three registered contacts responds. The contacts are called sequentially, with only one ultimately responding if the previous does not. In the application, a waiting period of one minute is allocated for each contact.

Observation. The output file of the model is a text file with the simulation's occurrences: when each sensor was switched "ON" and "OFF", when the user agent went from one room to another, when they fell down to the ground and where they were at the moment, and whether they succeeded in explicitly using the software system or not.

Initialization. The initialization of the model is done through a *setup* procedure. The scenario is presented according to the house's floor plan from the Aruba dataset [182]. The sensor positions are set up based on the data of this dataset, as illustrated in Figure B.1 in Appendix B.

Regarding the number of agents, there is only one user agent and 31 device agents, and the environment is the same for every simulation. It must be reset every time the parameters change, as this means different people are being simulated.

Input data. The necessary input data for this model are the parameters in the GUI (*slowness-of-movement*, *tremor-frequency*, *falling-frequency*, *application-use-frequency* and *simulation-duration*). At each initialization of NetLogo, each parameter presents its previous value (from when the model was saved for the last time). It is important to properly set up the parameters for a more accurate simulation.

In the ABMS designed for this experimental study, there are no elements of *emergence*, *adaptation*, *objectives*, *learning*, *prediction* or *collectives*, as well as no submodels.

5.6.2 Generated data

The activity of the ABMS observer and the interactions between the different components and actors of the simulation are represented in the activity diagram seen in Figure 5.20. The QA analyst is responsible for starting the simulation process through launching the ABMS observer, then:

- The ABMS observer opens the emulator and accesses the ParkinsonCom application. Then, it opens the ABMS in NetLogo. From that point on, a parallel process begins with actions run by the QA analyst, the ABMS, its observer, and the software application.
- The QA analyst sets up the simulation parameters values and starts the simulation, which makes the agent perform actions. The events are registered in the simulation log data and go on while the simulation does not reach its final time;
- At the same time, the ABMS observer keeps reading whatever happens in the simulation, checking the agents' status and actions. If the user agent fell (which triggers an implicit interaction) or used the software application (which triggers an explicit interaction), another agent will appear at the smart home's entrance and go to the main user agent. Then, they will inform the application that help has been provided before leaving again. If it is time to take a medicine, the agent goes to the bathroom

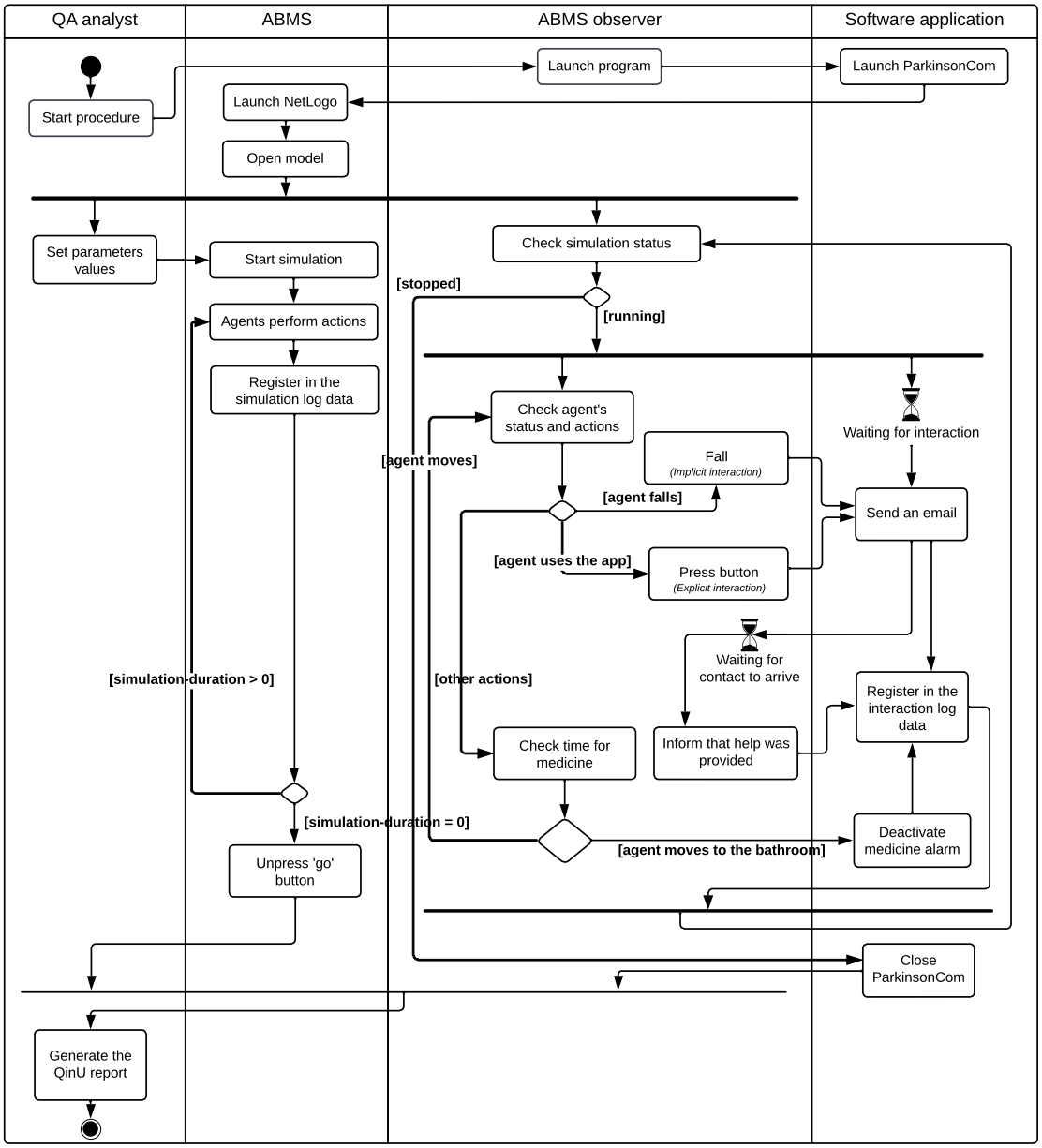


Figure 5.20: Activity diagram for the third experimental study.

and deactivates the medicine alarm. In the meantime, ParkinsonCom is active and awaits any interaction. Once it happens, the occurrences are supposed to be saved in the interaction log data;

- Once the established time is over, the 'go' button in the ABMS is deactivated, the simulation is stopped and the software application under evaluation is closed, bringing the process to an end. From there, both the simulation and the interaction log data are ready to be examined by the QA analyst.

Considering the QinU characteristics to be evaluated (effectiveness, efficiency and freedom from health risk), it is important to identify when the user agent uses the software application, and its position in the smart environment (e.g., for elderly people, falling down in some places can be more dangerous than in others). The relevant events from the simulation are saved in the *simulation log data* with their simulation timestamp.

Table 5.7 presents an extract of the log data generated by a simulation execution, showing one instance of each possible interaction. The first column presents the timestamp for the simulation clock, while the second column shows the events that were established to be printed, such as a motion sensor being switched on or off, the user agent going from one room to another, the outcome from a procedure that generates a number – which then triggers (or not) interactions within the software application.

Table 5.7 presents an extract of the simulation log data from one execution with different occurrences. We note that:

- the two first rows registered two motion sensors switched on at first, which means that the user agent moved near them;
- the following two rows present the same motion sensors now switching off, as the user agent moves farther from it;
- in the fifth row, *room_changing* notifies that the user agent left the bedroom and entered the living room;

Table 5.10: Sample of the simulation log data from NetLogo.

time in the simulation	***event***
08:00:01	sensor M002 ON
08:00:01	sensor M007 ON
08:00:27	sensor M002 OFF
08:00:28	sensor M007 OFF
08:00:29	room_changing bedroom living room
08:00:31	sensor M008 ON
...	...
10:51:42	"I hear the alarm. I will take my medicine."
...	...
11:02:03	"I took my medicine. I am deactivating the alarm."
...	...
15:00:00	random_number: 4 [0..9] "I am requesting help in the application."
...	...
19:31:37	"I fell in the bathroom."
...	...

- at 08:30:00, the medicine alarm goes off and the user agent moves to the bathroom (designated place to take their medicine);
- at 08:38:41 (after switching motion sensors on and off along the way to the designated place), the user agent deactivates the medicine alarm. This occurrence must be read by the ABMS observer, that transmits the information to ParkinsonCom and deactivates the actual alarm in the application;
- at 15:00:00, it shows an explicit interaction displaying the generated number (number 4) within its possible range [0..9]. This value determines whether the agent uses the application or not;
- at 19:31:37, the user agent fell down in the bathroom. This event must be read by the ABMS observer, which then should trigger an action in the software application for the registered primary contact to be called for help (simulating an implicit interaction).

Once the ParkinsonCom application is launched, it automatically launches two

databases: (i) the first one is essential to run the application – it saves texts, contacts and reminders that the user provides to the software – and (ii) one for the logs – it saves the navigation steps and actions that the user performs. Database management is particular to each and every software application, and varies according to its developers and stakeholders. Our methodology does not require any changes to the application under evaluation, but instead, promotes an approach that can be adapted to different software. This fact further reinforces that the QA analyst is expected to know how to manage the system under evaluation.

For our QinU measurement, we will need to extract data only from (ii), the log database. This database has five tables, but only one of them – the action table – is relevant for our study. Moreover, this table presents nine columns in total, but only four are included in the sample below. This is due to the other five columns being strictly related to features of the application that are not under evaluation.

The data generated by the interaction with the application is recorded in a SQLite database – which was accessed through the GUI of SQLiteStudio – and it includes the real time that ParkinsonCom was being used, which is different from the time inside the simulation. Table 5.8 contains an extract of the data generated by the interaction with the application during the same execution presented in Table 5.7. Column *type* presents the type of the action saved in that instance of the database. The database is set to record 21 different kinds of actions, however, we will only analyze *validation_notification_reminder* and *emergency_request*, which are recorded in the database when the user agent deactivates the medicine alarm and calls for help (whether clicking on the button or falling down), respectively; *startTime* presents the date and the time an action is initiated; *endTime* presents the time an action is finished – it can be null, as it is not needed for an instantaneous actions, such as clicking a button; *id_reminder* presents the ID of the medicine alarm concerned by the respective action. It can be null if that instance is not related to the *validation_notification_reminder* action.

Table 5.11: Sample of interaction log data.

type	***startTime***	***endTime***	***id_reminder***
...	...	...	...
validate_notification_reminder	2025-05-01 08:40:21	null	1
...	...	...	...
emergency_request	2025-05-01 10:02:37	null	null
...	...	...	...
emergency_request	2025-05-01 13:58:31	null	null
...	...	...	...

We point out the following:

- the activity *validate_notification_reminder* presents the real-world time when the medicine alarm was deactivated – it was established it would go off at 8:30 a.m., which does not match with the simulation clock (as seen in Table 5.7). For that reason, the ABMS observer is responsible for analyzing how long the user agent takes to perform actions in the simulation and take that same time – according to the real-world clock – to interact with the application. That way, it registers the simulated time as expected in the real application. This demonstrates the necessity for the ABMS observer to identify and deliver key data from the simulation to the evaluated application;
- the activity *emergency_request* has the same trigger for both the explicit and the implicit interaction. The colors in Tables 5.7 and 5.8 have the purpose of showing which occurrences they correspond to in both log data. To know what event triggered the call out for help, it is necessary to check in the simulation log data which event unfolded.

5.6.3 QinU evaluation through measures

As previously mentioned, we decided to evaluate the QinU for ParkinsonCom considering effectiveness, efficiency and freedom from health risk. In this third experimental study, we applied the same QinU measures than in the previous experimental study for effectiveness and efficiency (refer to Table 5.4).

For freedom from health risk, as previously detailed in Section 4.2.2, we adapted a measure from the ISO/IEC 25022:2016 standard [22]. This measure is originally named "Safety of people affected by use of the system" (identified as RHe-3-G), and calculates the incidence of hazard to the users. This adaptation was named "User safety risk" (see Table 5.12). In this third experimental study, it is applied exclusively for the execution of the task by implicit interaction. That means that if the system fails to detect an event (in this study, the user falling) and does not trigger a response (which would be requesting help), it may jeopardize the user's health. Regarding the medicine alarm feature, for simplicity, we assumed that the user always clicks the button to confirm they have taken the medicine. Alternatively, we could have set a maximum waiting time (since the user must eventually confirm it for the pop-up to be closed and allow the application to be used again) and, in that case, consider the user to be at health risk.

Table 5.12: QinU measure applied in the third experimental study (based on the ISO/IEC 25022:2016 standard [22]).

ID	**Name**	**Description**	**Measurement function**
RHe-A	User safety risk from implicit interactions	Rate of failures for implicit interactions	$Z = ((B - A)/B) * 100$; A = number of implicit events in the interaction log data; B = number of implicit events in the simulation log data

Regarding the quality measurements, we point out the following:

- for effectiveness, we observed the moments when the user agent decided to request help using the software application, taking into consideration if they were successful or not. It is known that Parkinson patients may have difficulties to click on buttons due to their tremors, therefore, the higher the *tremor-frequency* parameter, the more likely the user agent is to not be able to click on the button for requesting help through the application. This is particularly important in this case, as such button is a small icon in the top right corner of the screen (see Figure 5.2). To measure that, it is necessary to analyze in the interaction log data the amount of times they succeeded in doing so, and in the simulation log data, the total amount of times the agent has

started this task (Ef-1-S in Table 5.4). Effectiveness was also evaluated for the implicit interaction of this task. Regarding the other explicit interaction performed by the user agent (deactivating the medicine alarm once it goes off), for the sake of simplicity, we fixed that they will always be able to click on this button, as it is a large one. As consequence, we did not evaluate the effectiveness for this task;

- for efficiency, we observed how long the user agent took to request help and to deactivate the medicine alarm. For that, we established that once the alarm goes off, the user agent heads to the bathroom (place designated for the user agents to take their medicine), and will only switch it off once the action is completed. The higher the *slowness-of-movement* parameter, the longer it takes for this event to transpire (Ey-1-S in Table 5.4); and,
- freedom from health risk is applied exclusively for the action of requesting help by implicit interaction. That means that if the system fails to detect an event (the user falling) and does not trigger a response (which would be requesting help), it may jeopardize the user's health. To measure that, it is necessary to observe how many times the user agent fell in the simulation log data, and how many times it was not triggered a help request in the interaction log data (RHe-A in Table 5.4).

The simulation and interaction log data are merged in a form equivalent to that of the previous experimental study (see Table 5.9) and used to calculate the measures. Table 5.13 presents the calculated measures for each one of the simulation executions, representing twenty-six different people. The ID column identifies the user agent related with each row. T1 and T2 represent the two evaluated tasks – requesting help and deactivating the medicine alarm – where *(exp)* and *(imp)* mean explicit and implicit, to refer to the kind of interaction being assessed for this characteristic. Each line is the result of the measures for each user, computed in the same way.

For the three measures concerned by this implementation, the calculation has been performed as follows:

Table 5.13: Measures calculated for each user agent
(T1 = requesting for help, T2 = deactivating alarm, *imp* = implicit, *exp* = explicit)

	Effectiveness		**Efficiency (min:sec)**		**Freedom from health risk**
ID	**T1 *(exp)***	**T1 *(imp)***	**T1 *(exp)***	**T2 *(exp)***	**T1 *(imp)***
P1	100%	100%	01:07	02:01	0%
P2	90%	71.42%	02:56	03:12	28.58%
P3	96%	100%	03:48	05:58	0%
P4	42.14%	100%	04:02	06:23	0%
P5	77.5%	70.58%	05:10	06:04	29.42%
P6	96%	60%	03:22	04:36	40%
P7	100%	80%	04:09	06:08	20%
P8	81.25%	62.5%	03:53	05:47	37.5%
P9	84.61%	100%	02:49	03:34	0%
P10	76%	81.81%	06:07	08:18	18.19%
P11	90%	96%	02:32	04:05	4%
P12	92.3%	66.66%	03:41	05:22	33.34%
P13	55.55%	53.33%	03:05	07:11	46.67%
P14	52.72%	50%	00:52	01:46	50%
P15	78.94%	66.66%	02:42	03:57	33.34%
P16	44.6%	40%	06:55	07:27	15.4%
P17	50%	55.66%	04:17	03:19	22.34%
P18	92%	100%	05:33	06:45	0%
P19	81%	35%	07:38	08:12	16.4%
P20	100%	50%	06:49	07:29	50%
P21	100%	66.66%	06:18	07:08	33.34%
P22	88%	90%	04:07	06:37	10%
P23	94.5%	71.42%	03:14	03:28	28.58%
P24	100%	80%	02:25	03:51	20%
P25	48%	100%	06:10	06:56	0%
P26	100%	40%	05:17	06:29	60%

- for the effectiveness evaluation, according to the measure Ef-1-S (tasks completed), a mean is calculated considering the total of tasks started in the simulation log data (20) over the total of the task execution in the application in the interaction log data (18). The mean for P2 in T1 *(exp)* is 90% and in T1 *(imp)* is 71.42%, generating a total mean of 80.71% for effectiveness of the evaluated features;
- for the efficiency evaluation, a mean task time is calculated on top of all the time taken to perform each evaluated explicit interaction (Ey-1-S) computed from the interaction log data. The mean task time for both evaluated interactions is 3 minutes and 4 seconds; and,
- for the freedom from health risk evaluation, according to the measure RHe-A (user

safety risk), we compute that the total of falls is five in the interaction log data and seven in the simulation log data, which means the system failed twice, and as consequence, the result is 28.58% ($2/7 \times 100$) RHe-A value. This is probably due to the internet connection issues during the 24 hours of execution.

To visualize the results, we generated charts by crossing the three evaluated measures (effectiveness, efficiency, and freedom from health risk) with the variables that most affect them. We first analyzed each measure in relation to the user traits that had the greatest impact, and then in relation to the environment (i.e., rooms of the house) where the occurrences took place. This approach makes it possible to investigate how different elements of the context of use (both user and environment) influence the outcomes.

Regarding effectiveness, the user traits that impact it the most are *tremor-frequency* (affecting explicit interactions with the application) and *falling-frequency* (affecting implicit interactions). Figure 5.21 illustrates how these parameters influence the occurrence of events in the ABMS. In this example, effectiveness is calculated as the mean of completed tasks, considering the relevant interactions for the evaluated feature (i.e., clicking the button for explicit interaction and falling down for implicit interaction).

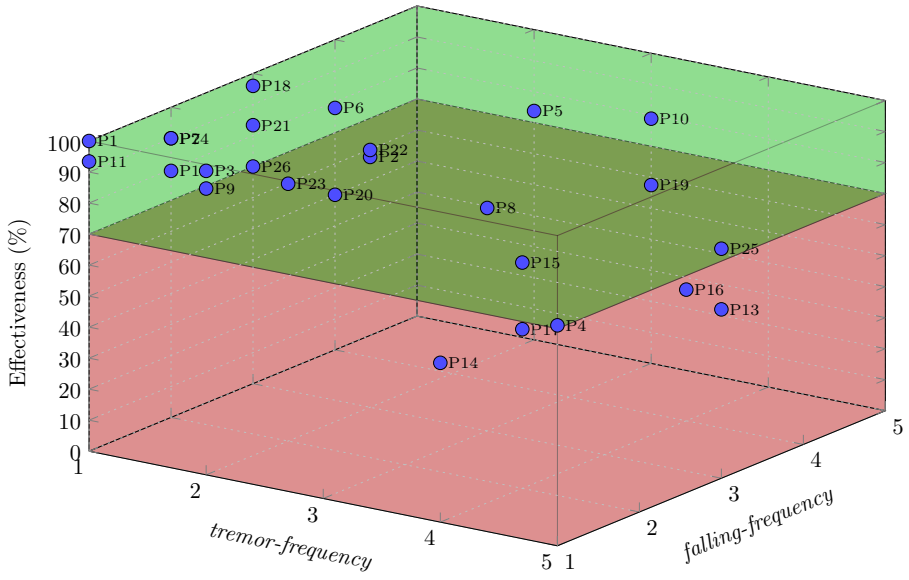


Figure 5.21: Mean effectiveness of the completed tasks for each user agent (P1 to P26) and their respective parameters for tremors and falls.

Figure 5.21 shows a threshold of 70% considered acceptable for the mean effectiveness of the evaluated features of the targeted application. This value is intended to be determined by each QA analyst during their own assessment. In the example presented, the QUESTS methodology measured that ParkinsonCom achieved a mean effectiveness of at least 70% in 22 out of the 26 evaluated user profiles.

Additionally, the location of the user agent (i.e., their spatial position) affects the likelihood of a fall, as discussed in Section 5.5 (see Table 5.6 for reference values). Therefore, to analyze how the environment influences effectiveness, we focused only on implicit interactions. Figure 5.22 shows the mean effectiveness by location, i.e., where users were at the time of a fall. For example, across all user agents, 18 falls occurred in the bedroom. Of these, 15 were correctly registered by the application, resulting in a mean effectiveness of 83.33%.

Overall, the application was triggered in all rooms of the environment. Effectiveness was slightly lower in rooms defined as slippery (bathroom and kitchen), likely due to the higher number of implicit interactions in these rooms, which increases the chance of failures. The living room showed a similar decrease in effectiveness, possibly explained by the habits of the person in the dataset, since its motion sensors were "ON" more frequently than those in other non-slippery rooms. These findings illustrate how real-world data can shape the outcomes of simulations grounded in such datasets. Finally, effectiveness is also influenced by the Internet connection, as the evaluated application relies on connectivity.

Regarding efficiency, the most influential user traits are *slowness-of-movement* and *tremor-frequency*. Figure 5.23 presents the mean efficiency for each user agent, calculated from the average time taken to complete the two explicit tasks in the application: requesting help and deactivating the medicine alarm. *Tremor-frequency* primarily affects the help request, while *slowness-of-movement* impacts both tasks by slowing navigation within the application and increasing the time needed to reach the bathroom to deactivate the alarm. These results highlight how user traits shape the speed and responsiveness of interactions



Figure 5.22: Mean effectiveness for each room in the environment.

with the system.

In Figure 5.23, the colored dashed lines represent projections of the data onto the different coordinate planes to help interpretation. The blue dashed line corresponds to the projection onto the *slowness-of-movement* × *tremor-frequency* plane (x–y). The green dashed line corresponds to the projection onto the *slowness-of-movement* × efficiency plane (x–z). Finally, the red dashed line corresponds to the projection onto the *tremor-frequency* × efficiency plane (y–z).

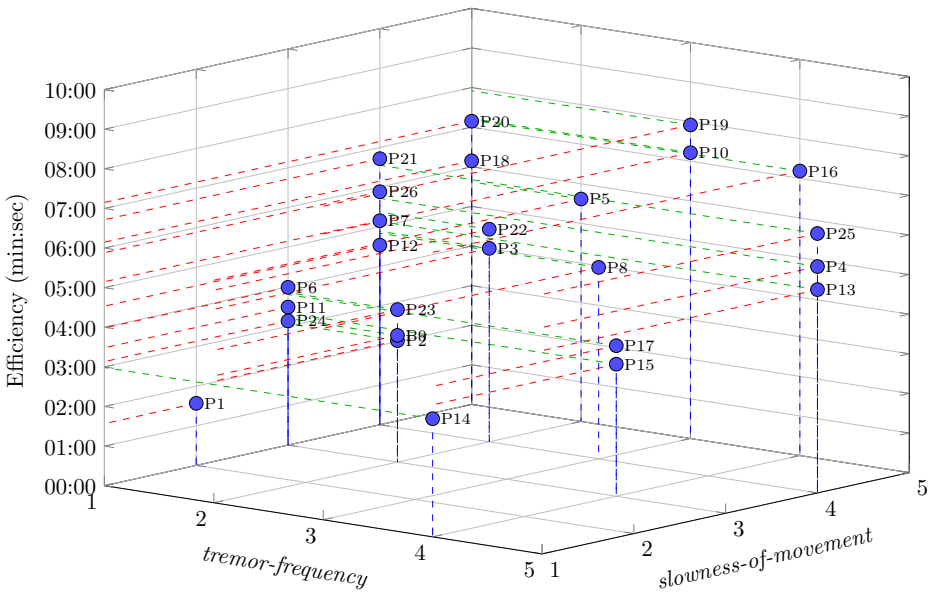


Figure 5.23: Mean efficiency of the completed tasks for each user agent (P1 to P26) and their respective parameters for tremors and slowness.

We did not define a fixed threshold for efficiency. In general, lower values indicate better performance, but the outcome depends on the user's traits. In particular, high tremor frequency (≥ 4, e.g., P16, P25) and/or high slowness of movement (≥ 4, e.g., P16, P19) resulted in tasks being executed with lower efficiency, which is considered acceptable.

Concerning the environment and its effect on efficiency, Figure 5.24 shows the mean efficiency for all user agents combined, calculated from both explicit interactions simulated in the application. For the task of deactivating the medicine alarm, efficiency depends on the user's location, since the alarm must be turned off in the bathroom (designated place for taking medicine). The farther the user is from the bathroom when the alarm goes off, the longer it takes to complete this task. This explains why the bedroom, which is one of the farthest rooms from the bathroom (see the simulated house floor plan in Figure 5.13), may exhibit the lowest efficiency.

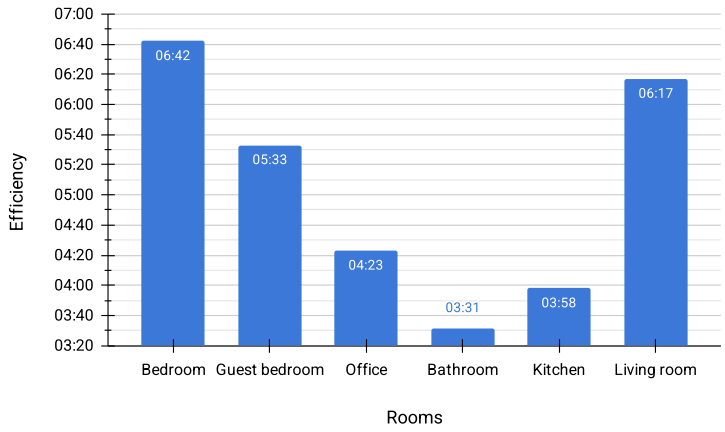


Figure 5.24: Mean efficiency for each room in the environment.

Since freedom from health risk was evaluated only through the implicit interaction of requesting help (i.e., falling), and this event is determined solely by the user trait *falling-frequency* and the environment where the user agent is located, the corresponding charts consider only one variable.

Figure 5.25 presents the evaluation of this measure (*user safety risk*) against *falling-frequency* for each user agent. The results show a correlation between *falling-frequency*

and the user safety risk: the more often a user agent falls, the higher the likelihood that some occurrences will not be transmitted to the application. In this implementation, a device agent (representing an Internet-connected sensor) triggered the system's response. Consequently, this measure is also tied to the performance of that resource, which may explain why the observed correlation is not particularly strong, as it depends more on external factors than on the ABMS itself.

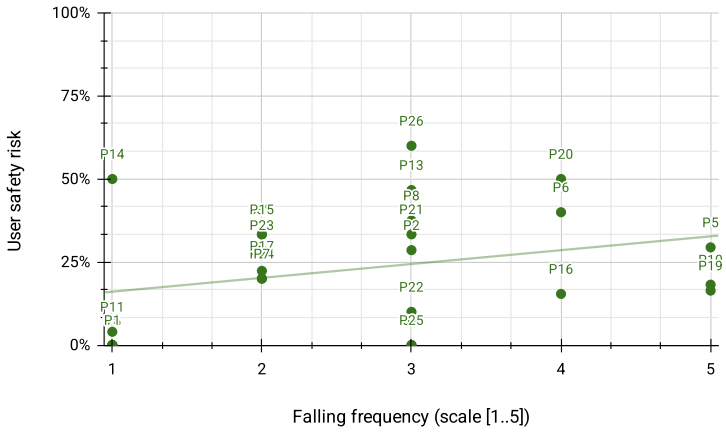


Figure 5.25: Mean user safety risk for each user agent.

From the perspective of the environment, Figure 5.26 presents the mean user safety risk for all user agents combined. Although there is no strong correlation between application failures and specific rooms, the same pattern observed for effectiveness in Figure 5.22 also applies here: user safety risk was slightly higher (i.e., freedom from health risk was lower) in rooms defined as slippery (bathroom and kitchen) and in the living room, where the user agent spent considerable time.

This is likely due to the higher number of implicit interactions in these rooms, which increases the likelihood of failures in the application's response. It is worth noting that the values are higher because we are evaluating user safety risk, a QinU measure applied to assess the freedom from health risk characteristic. Accordingly, lower risk values correspond to higher freedom from health risk.

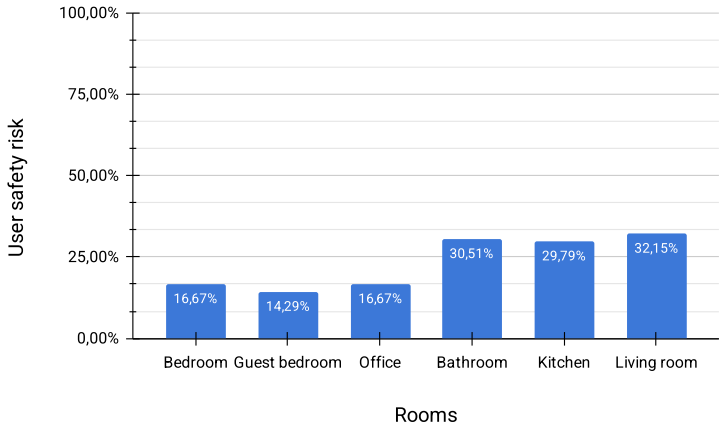


Figure 5.26: Mean user safety risk for each room in the environment.

5.7 Discussion and analyses of the results

5.7.1 Extending the applicability

Even though the implementation of QUESTS in this thesis was limited to two types of sensors due to time and scheduling constraints, the potential of the methodology extends far beyond these initial boundaries.

One of the most powerful advantages of simulation lies in its capacity to replicate and explore countless scenarios by systematically varying the elements that define the context of use (detailed in Section 2.1.3): users, tasks, resources, and environments. Users may differ in skill level, habits, or impairments, and while some may share common traits, each unique combination of characteristics influences how they interact with software systems. This variability makes it possible to evaluate performance across a truly diverse population – without the practical and ethical challenges of assembling large groups of participants.

Although the experimental studies in this thesis focus on people with motor disabilities, the methodology is equally applicable to other groups. QUESTS could, for instance, be used to assess QinU in applications designed for people who are blind or have low vision, or for people who are deaf or hard of hearing. Instead of bringing together partic-

ipants with diverse disabilities during the early stages of evaluation – a process that can be logistically complex and time-consuming – researchers can simulate agents with these characteristics. Such simulations provide early insights into potential usability challenges and allow developers to refine their systems before conducting, finally, real-world testing with representative users for these specific situations.

Tasks within the simulation must capture the full range of activities that users perform to provide input to the system, covering both explicit and implicit interactions. Importantly, the QUESTS methodology also makes it possible to test scenarios that would otherwise be undesirable, unsafe, impractical, or ethically challenging with real users. For example, in the second and third experimental studies (Sections 5.5 and 5.6, respectively), falls were simulated to test whether the application could correctly detect and respond to them – something that may have to be staged with real participants. Similarly, other rare but important scenarios can be explored, triggered during the performance of daily activities. This could include a user forgetting to close a fridge – leading to higher energy consumption or spoiled food – or leaving the stove on, which could create a serious fire hazard. By modeling such edge cases alongside routine behaviors, the QUESTS methodology enables a more comprehensive evaluation of system performance under both normal and exceptional circumstances. For these events to be transmitted, the necessary resources – such as sensors – must detect the condition and pass the information to the application.

On the resource side, simulations can be used to test the reliability of sensors, devices, and network infrastructures, under both stable and unstable conditions. Although faulty resources were not modeled in any of the experimental studies due to the limited timeframe, device errors – such as a sensor failing to detect an input – could be developed in the ABMS in future extensions.

In addition, multiple sensors can operate concurrently to support the same or different applications. For instance, in the case of domestic appliances, an application could rely on sensors to monitor refrigerator contents and suggest recipes based on available ingredients,

while a separate sensor could be used to detect whether the door remains open. In a simulation, stochastic variations in agents' behavior – such as deciding to cook, to snack, or to ignore the suggestion – could produce different outcomes. Similarly, with a stove example, simulations could reproduce scenarios where a sensor fails to detect that the burner is still on, where the alert is delayed due to network latency, or where multiple warnings must be escalated until someone responds. By exploring these possibilities, the QUESTS methodology allows for the evaluation of not only the intended functionality of the system, but also its resilience to errors, delays, or failures in the supporting infrastructure.

Finally, environments can be adjusted to reflect different contexts, such as a quiet home, a busy workplace, a hospital ward, or even outdoor conditions subject to noise, crowding, or interruptions. By combining variations across these dimensions, simulations allow to explore not just isolated cases but the full spectrum of potential contexts of use. Different environments include technical, physical, social, cultural and organizational dimensions. Each one of these dimensions can vary independently, and this complexity increases further when considering all possible combinations. Regarding the technical environment, different gadgets can be modeled depending on the software application under evaluation, including desktops, smartphones, tablets, smartwatches, motion sensors, temperature sensors, and more. These choices, in turn, affect the physical environment, as the devices are placed within it. In the three experimental studies conducted in this thesis, we modeled a smart home where each room could have a distinct impact. However, larger houses, entire buildings, or even cities could have their software application evaluated. Regarding the social environment, additional user agents – such as caretakers, relatives, or neighbors – can also be modeled and simulated. This would certainly influence the use of the software application, as more people interact with it in different ways.

This capacity to scale and diversify testing enables the early identification of weaknesses, the anticipation of usability issues, and the generation of actionable insights that would be prohibitively expensive, time-consuming, or even impossible to obtain through

conventional user studies. Most importantly, the QUESTS methodology can account for the richness and variability of real-world interaction while remaining efficient and adaptable.

Regarding the number of potential scenarios to be executed in our third experimental study, we note that:

- The user profile (U) consisted of three traits, each with five possible values (1 to 5). This configuration could generate 125 ($5 \times 5 \times 5$) profiles. However, as discussed in Section 4.5, the QA analyst may restrict this variability depending on the case. In this study, to ground the study in real-world data, we kept the 26 original user profiles of ParkinsonCom's participants;
- The resources (R) included 31 motion sensors (each with values "ON" or "OFF") and one accelerometer with two possible values ("default" or "fallen"), generating 2^{32} potential scenarios. For the purposes of this application, however, the relevant aspect was simply detecting presence or absence in each space[14]. The 31 motion sensors were distributed as follows: kitchen (5), bathroom (2), office (4), guest bedroom (4), bedroom (7), living room (7), and corridor (3). Presence in a given space was defined as at least one sensor "ON"; absence was defined as all sensors "OFF". This same logic applied to all spaces. Therefore, the variation of scenarios regarding presence/absence could be reduced to 256 (2^7 for the motion sensors $\times$ 2^1 for the accelerometer);
- There were two tasks (T) and seven environments (E), of which two were classified as slippery and five as non-slippery.

Table 5.14 summarizes the number of potential scenarios to be executed in this experimental study.

[14] In other applications, all areas of a scenario may be relevant. For instance, a sensor at the kitchen entrance and another near the oven could support an application focused on children's safety.

Table 5.14: Elements of the context of use and the maximum number of potential scenarios.

Users	26
Tasks	2
Environments	7
Resources	$2^8 = 256$
$U \times T \times E \times R$	93,184

5.7.2 Validating the results

During the ParkinsonCom project, participants were asked to complete the Computer System Usability Questionnaire (CSUQ, previously mentioned in Section 3.4.1). Developed by IBM in 1995 as a modified version of the PSSUQ [189], CSUQ is a widely used instrument for assessing the usability of computer systems, software applications, websites, and other interactive systems [190]. It measures usability through its three characteristics: effectiveness, efficiency, and satisfaction. The questionnaire uses a scale ranging from "strongly agree" to "strongly disagree", allowing researchers to identify usability strengths and weaknesses and guide system improvements.

We attempted to use the results from the participants of the ParkinsonCom project to validate the outcomes of the QUESTS methodology by comparing them with different mean scores related to (i) overall utility satisfaction, (ii) system usefulness, (iii) quality of information, (iv) interface quality, and (v) general satisfaction. No correlations were found between the outcomes of the user agents (each representing a participant, with traits they had described themselves) and the responses of the real participants in CSUQ.

This result is expected, as we acknowledged that simulating user satisfaction would not be feasible. For example, P4 and P21 reported opposite tremor symptoms on the Parkinson's Well-Being scale (introduced in Section 5.3): one rated themselves a 1 on a scale of 1 to 5 for tremors, while the other rated a 5. Yet both reported the same overall utility satisfaction score of 5.3 (on a scale of 1 to 7). This illustrates further that satisfaction cannot be simulated, as it involves personal opinion, which is subjective and shaped by perception

and individual preferences.

In contrast, if log data from participants' actual use of the application had been available (which was not the case since user testing focused on communication features, as previously explained in Section 5.2.2), such data could have been applied to support model calibration. For instance, knowing the average time taken by users with severe slowness of movement to navigate a given feature can inform the definition of an input parameter, allowing it to be based on real-world data rather than on arbitrary assumptions, as discussed in Sections 5.4 and 5.5.

Furthermore, in the case where such data is available, the outcomes of the QUESTS methodology can be directly compared with real usage patterns. In this regard, since simulations may explore extreme cases, a suitable form of validation is to have real end users interact with the application beforehand, and use this log data to ground in real-world data all possible inputs to the application under evaluation.

5.8 Threats to validity

The experimental studies were analyzed based on Wohlin et al.'s [191] four validity threats, as outlined below.

Construct validity concerns both the validity of the ABMS and the computation of the QinU measures. Ideally, the measures obtained from the simulation would be compared with those from actual use of the evaluated software application (see Section 5.7.2). However, our goal is to provide an initial evaluation that avoids experimenting with real users. For this reason, the usual model validation based on comparing simulation output with real data is not applicable. To ensure at least partial validity of the model, we grounded the agents' behavior in real data used as input for the modeling process. We applied a process mining technique to a real-world dataset [182] to define patterns of mobility for the user agent. Moreover, the computation of measures was defined in accordance with ISO/IEC

standards [21, 22, 44].

Internal validity is threatened by the sensitivity of the ABMS to parameter values. Changes in these values can significantly alter the outcomes of a simulation. To mitigate this threat, we used real information from people with Parkinson's disease who tested the real ParkinsonCom application. The *simulation-duration* was set to 24 hours, using artificial data generated by the process mining. Moreover, the *application-use-frequency* set up by the QA analyst can affect the simulation results since a lower value creates more opportunities for the user agent to engage with the application. Therefore, this threat is under control as long as the QA analyst is aware of these limits and can run the simulation as many times as they wish, setting different values and analyzing the results.

External validity regards the generalizability of the methodology. Although the approach has been defined in a generic way, establishing the steps that must be followed for each new context, we prefer to be prudent and recognize this risk should be explored in future studies.

Conclusion validity concerns whether the final QinU measures really reflect the evaluation of the application considering the different values defined for the simulation. The simulation ran for 24 hours based on real-user data and behavior in the environment. We, therefore, consider this threat well mitigated and under control. However, it is worth noting that the model may omit certain features – for example, additional user traits or device-specific characteristics – that could have an impact on the interactions. For instance, it is important to highlight that we did not recreate any sensor failure to represent hardware issues, as this was out of the scope of the thesis.

5.9 Applying the QUESTS methodology: a step-by-step guide

From the application of the QUESTS methodology, we can bring up the necessary steps for its implementation. As the proposed methodology can be applied to evaluate the QinU of different types of software application, it is important to adapt all the features for the approach to work. A step-by-step guide for implementation is listed below.

i. Establish which features of the targeted application will be evaluated;
   - Identify implicit and explicit input to use such features;
   - Analyze whether the application communicates or exchanges data with other devices.

ii. Establish which QinU characteristics will be measured;
   - Identify the types of output necessary for the measurement;
   - Identify the types of input that need to be modeled to get the expected output.

iii. Identify user profiles;
   - Analyze the target audience for the evaluated application;
   - Identify which user characteristics are relevant in the context of use of the evaluated application, such as age or mobility.

iv. Identify environment and surroundings;
   - Analyze what are the possible conditions for the surroundings of the user at the moment of application usage;
   - Analyze what are the possible events in the environment that would trigger an input in the evaluated application;
   - Analyze if the smart devices interact between themselves;

- For example, determine whether the user is in a slippery place (as seen in the second and third experimental studies presented in this chapter), a crowded area, or a noisy environment, as these conditions can influence both user behavior and system performance.

v. Search for real-world datasets to ground the agents' behaviors;

- Look for datasets derived from the real use of sensors and applications that correspond to the phenomena intended to be modeled and simulated (based on what was identified in the previous steps);
- Clean and prepare the selected data so it can be used as input for the ABMS.

vi. Implement the ABMS;

- Model the users as agents, and define which tasks they should perform to fulfill their goals;
- Analyze if the smart devices in this scenario should be modeled as agents;
- Model the environment and the resources placed in it;
- Represent the explicit and implicit input that would be read by the application.

vii. Implement the ABMS observer;

- Develop a script that can read and analyze the events that occur in the ABMS, as well as interact with the software application under evaluation in order to transmit relevant occurrences in the form of input.

viii. Analyze the database schema;

- Design the structure in which data from simulations will be stored and identify how the targeted application stores the information generated from its use.

ix. Extract and merge information for the QinU report;

- Prepare a script that is able to collect information from both simulation log data and interaction log data and merge them into a spreadsheet, such as an Excel or

a Google Sheet, calculating the results according to the previously established measures.

x. Generate visual representations to facilitate interpretation;

- Create charts and graphs of the collected data in order to ease the understanding of patterns, highlight key findings, and provide an enhanced comprehension of the results for subsequent analysis and discussion.

Explicitly detailing the procedure contributes to enhancing transparency and reproducibility for other studies. Although certain constraints remain, the presented framework offers a practical resource for researchers seeking to adopt or adapt this approach in their own work.

5.10 Conclusion

The feasibility of the proposed approach was demonstrated in three different experimental studies. They targeted two different software applications that could receive explicit input directly from the user and implicit input via sensors.

This chapter has also presented possible threats to the validity of the experimental studies and a step-by-step guide to implement the approach. The performed experimental studies allow to indicate the contributions, strengths, constraints and limitations of the QUESTS methodology, which will be the topic of discussion in the following chapter, as well as future work.

Chapter 6

Conclusion and perspectives

The previous chapter detailed the implementation of three experimental studies, their results and the threats to their validity, as well as a step-by-step guide to implement the QUESTS methodology.

This chapter leads with final considerations and the main contribution of the thesis, then outlines the strengths, constraints and limitations of the QUESTS methodology before discussing other perspectives and future work.

6.1 Final considerations

This thesis proposes the QUESTS methodology, an innovative manner to evaluate the QinU of smart environment applications, with particular attention to the variability inherent in different contexts of use. QUESTS provides practical means for Quality Assurance (QA) analysts to assess software quality across multiple scenarios without the need for real users to interact with the system in a physical environment.

This study was motivated as the human interaction with the cyber-physical world is evolving. This pervasive connectivity, enabled by identification, sensing, and processing

capabilities, allows devices to interact seamlessly with their surroundings. However, alongside the benefits of the IoT paradigm, new challenges emerge. The rapid pace of technological change, application heterogeneity, device diversity, and other issues distinguish IoT applications from traditional ones [192].

Hence, it is essential to test and evaluate the software quality of systems before they are deployed to end users. This process serves not only to ensure user safety but also to support the approval and adoption of such applications. Guaranteeing the dependability of these tools becomes particularly important for senior citizens and individuals with reduced mobility, as software flaws in these cases may pose far greater risks than for users without such limitations.

Building on this idea, the QUESTS methodology was developed to avoid involving actual users in the initial stage of QinU evaluation for smart environment applications. This, in turn, made it possible to test different features of the applications under evaluation across multiple contexts of use. To validate the feasibility of the proposed approach, three experiments were implemented. They evaluated three QinU characteristics overall: effectiveness, efficiency, and freedom from health risk. The evaluated applications received explicit input from user agents and implicit input from user agents and device agents, reflecting common interaction patterns in smart environment systems. The QUESTS methodology was defined on top of the following scientific challenges (introduced in Section 1.2):

i. to model and simulate user interactions in smart environments using agents, and considering the variability of the context of use;
ii. to ground the simulation on real patterns;
iii. to calculate the QinU measures using log data collected from simulations.

To overcome scientific challenge (i), an ABMS was developed representing a smart environment with a resident in it, that would move around and perform interactions towards the evaluated application. Each experimental study had the ABMS adapted to the targeted

application in question.

To overcome scientific challenge (ii), we selected a publicly available dataset from CASAS, for which its data had been gathered from 31 sensors placed in the house of an elderly lady that lived by herself. Then, Python scripts were created to clean and prepare the data to be read, interpreted, and find patterns in it to be provided to the ABMS.

To overcome scientific challenge (iii), we adapted QinU measures from the ISO/IEC 25022:2016 standard so they could be applied to synthetic data instead of data generated from human-computer interaction with real potential end users during usability tests.

In summary, QUESTS integrates concepts from three distinct domains:

- *software engineering*, specifically QinU, by employing a model that considers the context of use – users, tasks, environments, and resources – during software evaluation;
- *agent-based modeling and simulation*, to simulate the behavior of a user living independently in a smart environment; and,
- *data mining techniques*, particularly process mining, to guide simulation behavior using real-world data.

6.2 Contributions

The contributions of this thesis are the following:

i. a proposition of a methodology for the evaluation of QinU based on simulation with software agents taking into account a large variability of the context of use. This allows for a change in the combination of the end users and the tasks they perform in the application, as well as the environment in which the users are placed with its resources, i.e., sensors;
ii. the grounding of the agents behavior on real data using DM techniques;
iii. experimental studies implementing the proposed approach showing its feasibility in

different scenarios adapting to the respective features;

iv. a catalog of how QinU has been measured in software applications found in smart environments;

v. the adaptation of QinU measures as proposed by ISO/IEC standards to allow the data extraction from the smart environment simulation;

vi. the proposal of measures for the newly defined QinU characteristics to be evaluated by simulation;

vii. the identification of which characteristics from the QinU model (ISO/IEC 25019:2023 [21]) that must be evaluated considering the perceptions of the final end user (satisfaction, experience, trustworthiness and compliance) and the ones that can be evaluated through simulation (effectiveness, efficiency, accessibility, suitability, freedom from economic risk, freedom from environmental and societal risk, freedom from health risk and freedom from human life risk).

Contribution (i) concerns the overall methodology, which integrates ABMS and QinU measures to evaluate software applications within smart environments. This approach allows for the assessment of applications without placing actual users in unfamiliar or potentially unsafe spaces strictly for testing purposes, while simultaneously enabling the variation of the context of use in a controlled and systematic manner. Contribution (ii) involves the extraction of meaningful patterns from the selected dataset and providing them to the ABMS, ensuring that the simulation is guided by real-world data. Contribution (iii) relates to the iterative refinement of the three experimental studies performed, with each successive experimental study building on the insights and improvements derived from the previous ones. Contribution (iv) emerges from the literature review conducted in Section 3.4, which provided an overview of the current state of the art and revealed how software applications in smart environments have been evaluated. Contribution (v) refers to the adaptation of QinU measures, as proposed by the ISO/IEC 25022:2016 standard, to allow the inclusion of data obtained from simulations, extending then the applicability of these mea-

sures to synthetic yet realistic scenarios. Contribution (vi) concerns the proposal of measurements based on the description of the QinU characteristics form ISO/IEC 25019:2023 [21] that do not have any measures proposed by any standards, such as accessibility and suitability. Lastly, contribution (vii) refers to the analyses of which QinU characteristics can be evaluated by simulation.

6.3 Strengths, constraints and limitations

Considering the design and application of the QUESTS methodology, several strengths can be highlighted:

- *capacity to evaluate QinU without involving real users* – QUESTS methodology enables a QinU evaluation without requiring the participation of real users. Instead, software agents – configured with varying characteristics – simulate user behavior. When combined with diverse environmental parameters, this setup allows for the creation of multiple scenarios and contexts of use. This variability eliminates the need for assembling a large group of people to represent distinct user situations, which would be necessary for accurate in-vivo evaluations. In contrast, real-user testing would require prolonged exposure to the application to naturally generate relevant events;
- *generation of diverse contexts of use for QinU evaluation* – QUESTS connects directly to the real application, recreating events and running as many simulations as needed to gather sufficient data for analysis, while also varying the elements in the context of use; and,
- *flexibility across various technologies* – the *ABMS observer* component in QUESTS is adaptable to different types of applications, making the methodology customizable according to the specific needs of the QA analyst. In this implementation, to test an application for mobile devices, we applied commands from the AutoIt library in the Python script to interact with the application under evaluation, as ParkinsonCom

was running on an emulator locally installed on the computer. If the QA analyst targets an online application, such as a webpage, the observer part can employ other libraries, such as Selenium, for example.

Together, these advantages illustrate how the QUESTS methodology can support scalable and adaptable evaluations in ways that traditional user-based testing cannot easily achieve.

Aside from its strengths, the methodology also presents certain constraints that define the boundaries within which it can be effectively applied, as follows:

- *dependence on a new actor for QinU evaluation* – the involvement of an ABM designer is essential from the outset. This individual works alongside the QA analyst – typically responsible for quality evaluation – to build and refine the simulation model. This collaboration represents a shift from traditional QA processes and introduces a new layer of complexity;
- *need of coding the ABMS and its observer for each application to be evaluated* – it is necessary that all evaluations have an ABMS that properly represents the usage and the context around the targeted application, as well as an ABMS observer that knows what to analyze and extract. Because of that, both elements need to have its source code adapted to whatever the end goal is with the specific QinU evaluation at hand;
- *need of hardware dedicated exclusively for the approach* – in the three experimental studies presented in this thesis, both of the applied Python libraries used to provide input to the targeted application take control over the mouse cursor. For that reason, it has been required that a computer works exclusively on running the simulation. An implementation using different libraries and/or tools requires further study and might not work in the same manner.

Despite the advantages of using an ABMS approach, there are some limitations:

- *model incompleteness* – a simulation inherently reflects only selected aspects of real-world dynamics. During model construction, the ABM designer chooses which entities and characteristics to represent. This selective representation can omit important real-world behaviors, limiting the model's fidelity;
- *different timeframes between simulation and reality* – simulations and the real world operate on different time scales (i.e., the clocks). This can be problematic if precise event timing is required to evaluate certain QinU properties. One possible solution is to consider the use of a digital twin rather than a simple simulation, which would allow for real-time interactions. Nevertheless, the asynchronous nature of simulations is also advantageous, enabling QA analysts to obtain results significantly faster than they would through real-world testing; and,
- *limited application scope* – so far, QUESTS has been applied to a software developed with the purpose of being used in a small online application and in a third-party application. To validate its broader applicability, it needs to be adopted by other researchers and practitioners across various software domains.

Overall, these limitations highlight the challenges of evaluating QinU through simulations, including potential gaps in model fidelity, differences in timing between simulated and real-world events, and the current restricted scope of applications. They emphasize the need for careful interpretation of results and further validation across diverse software environments.

The identified advantages and constraints not only support the formulation of conclusions but also creates space for envisioning future research and broader perspectives, as discussed in the following section.

6.4 Future work

While this thesis has addressed a wide range of aspects, it also paves the way for further exploration, offering new opportunities for advancing research on these subjects.

Regarding the QinU perspective, one possible direction for future work involves exploring different dimensions of the context of use: the experimental studies in this thesis have been limited to the physical dimension of the context of use, as users of smart environment applications operate in a physical environment [193]. Other dimensions can be modeled, such as the social environment: if another person is in place to help the primary user with the application, the context of use changes. The experimental study presented in Section 5.6 adds a second user agent that represents a person to come for help when called, however, no interaction between these user agents has been developed. This could be represented in the model by adding another user agent with their own behavior to collaborate with the simulated resident of the smart environment. In addition, a technical environment could be developed to support the evaluation of applications accessed through various types of devices. These modeled devices can be utilized by the user agents under different circumstances (e.g., accessing the evaluated application through the computer or the smartphone depending on where they are or what time it is), thereby allowing the context of use to vary depending on the specific device in use at a given moment.

Furthermore, another possible direction would be proposing measures for the newly defined QinU characteristics. As no ISO/IEC standards have been published to propose measures for the evaluation of the characteristics defined in the most recent QinU standard (ISO/IEC 25019:2023 [21]), it would be valuable to develop and validate quality measures for them. Moreover, it is important that these measures consider both implicit and explicit input, to evaluate all sorts of interactions that can happen in smart environments.

Tackling the DM stage, some techniques could be applied to multiple datasets, which then could provide patterns for different types of agents all throughout the simulation. We

have discussed about implementing an ABMS that transmits events to different software applications, therefore generating more than one interaction log data throughout one single simulation.

Concerning validation, one possible direction for future work is to compare simulated and real-world outcomes: one way to further validate the implementation of the approach is by conducting empirical studies that apply the QUESTS methodology and comparing the results with data obtained from real user interactions with the evaluated application.

In respect to recent advances, it would be interesting to investigate the integration of digital twins into the QUESTS methodology, as this would enable real-time interactions. Digital twins are virtual replicas of physical systems that maintain a continuous data exchange with their real-world counterparts [194], and their use could help address these challenges by providing a more synchronized and realistic testing environment. While the asynchronous nature of simulations remains advantageous – allowing QA analysts to obtain results much faster than through real-world testing – it may also introduce challenges for the future implementation of the methodology.

Moreover, the recent integration of Large Language Model (LLM) into ABMS provides new directions towards simulating realistic user behaviors, including decision-making and social interactions. However, the survey from [195] shows that there are few examples of LLM-empowered ABMS concerned with the interaction with software systems. A representative work in this category is provided by [196], where LLM agents emulate some human individual traits to simulate human behaviors in interaction with the recommender system. In such studies, the (real) user satisfaction regarding the resulting recommendations is measured, but the goal remains to simulate user preferences, not to evaluate the quality of the interaction with the system before its use by real users.

Another direction for future work would to be to generalize the methodology, developing a general-purpose tool that supports the methodology across various applications, independent of the specific platform used to implement the ABMS or its observer. In addi-

tion, applying the QUESTS methodology in real-world QA practice, putting the approach in action, either within software companies developing smart environment applications or through independent initiatives, and fostering collaboration between QA analysts and agent-based modeling experts in the context of software quality evaluation.

In summary, this thesis has developed a methodology for evaluating the Quality in Use of smart environment applications, while employing agent-based modeling and simulation to reproduce artificial human-computer interactions and data mining techniques to ground such in real-world data. The results demonstrate that a first instance of the QinU can be assessed without relying on real users, while still generating diverse contexts of use and providing actionable insights for software evaluation. Looking ahead, these findings open opportunities for applying the methodology across a wider range of applications and domains, as well as for integrating it with real-time systems or digital twins to further enhance its accuracy and applicability.

Appendix

A. Quality in use

Definition of Quality in use characteristics

The definitions of the following QinU characteristics have been taken verbatim from the ISO/IEC 25019:2023 standard [21].

Beneficialness: extent of benefit resulting from the use of a product, system or service.

Usability: extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use.

Accessibility: extent to which products, systems, services, environments and facilities can be used by people from a population with the widest range of user needs, characteristics and capabilities to achieve identified goals in identified contexts of use.

Suitability: extent to which behaviours or outcomes, or both, of a product meet specified quality requirements when used.

Freedom from risk: extent to which a product or system mitigates the potential risk to economic status, human life, health, society, financial values, enterprise activities, or the environment.

Freedom from economic risk: extent to which a product or system mitigates the potential risk to financial status, efficient operation, commercial property, reputation, or other aspects in the intended contexts of use.

Freedom from environmental and societal risk: extent to which a product or system mitigates the potential risk to the environment and society at large in the intended contexts of use.

Freedom from health risk: extent to which a product or system mitigates the potential risk to people's health in the intended contexts of use.

Freedom from human life risk: extent to which a product or system mitigates the potential risk to people's lives in the intended contexts of use.

Acceptability: extent to which a human response is favourable when accepting or installing a product, system or service software tool designed to perform some frequently used function.

Experience: extent to which users or stakeholders accumulate knowledge or skill acquired over time, especially that gained in a particular profession.

Trustworthiness: extent to which users or stakeholders have confidence that their expectations are met in a verifiable way.

Compliance: extent to which a user or other stakeholder has confidence that a product, system, software, or service in use meets requirements, as required by rules or laws.

Proposed measures for Quality in use characteristics

Table 6.1 presents one measure for each QinU characteristic proposed by the ISO/IEC 25022:2016 standard [22].

Table 6.1: Proposed measures for QinU characteristics [22].

Characteristic	**Name**	**Description**	**Measurement function**	**Method**
Effectiveness	Tasks completed	The proportion of the tasks that are completed correctly without assistance	X = A/B; A = Number of unique tasks completed; B = Total number of unique tasks attempted	Measure user performance
Efficiency	Task time	The time taken to successfully complete a task	X = T; T = Task time	Measure user performance
Satisfaction	Overall satisfaction	The overall satisfaction of the user	$Z = \sum Ai$; Ai = response to a question	Questionnaire
Freedom from economic risk	Return on investment (ROI)	The return on investment	X = (A − B)/B; A = Additional benefits obtained; B = Invested amount	Business analytics
Freedom from environmental risk	Environmental impact	The environmental impact of the manufacture and use of the product or system compared to a target	X = Aa/At; A = Environmental impact (a = actual, t = target)	Usage statistics
Freedom from health risk	Safety of people affected by use of the system	The incidence of hazard to people affected by use of the system	X = A/B; A = Number of people put at hazard; B = Total number of people who could be affected by use of the system	Usage statistics
Pleasure (user experience)	User pleasure	The extent to which the user obtains pleasure compared to the average for this type of system	X = A; A = Psychometric scale value from a pleasure questionnaire	Questionnaire
Trustworthiness	User trust	The extent to which the user trusts the system	X = A; A = Psychometric scale value from a trust questionnaire	Questionnaire
Suitability		The extent to which the system meets the requirements...	X = A/B; A = total of scenarios that the QA analyst find the application suitable for the context of use; B = total of scenarios	Questionnaire

B. ODD protocol

Table 6.2 presents brief descriptions of each element of the ODD protocol according to the ODD Guidance and Checklists [84].

Table 6.2: Elements from the ODD protocol [84].

Overview	*1. Purpose and patterns* A concise statement of the purpose(s) of the model.
	2. Entities, state variables and scales A description of what is represented in the model (which can also be called an "ontology").
	3. Process overview and scheduling A summary overview of what the model does as it executes: what processes are executed and in what order.
Design concepts	*4.1 Basic principles:* relates the model to existing ideas to place it within its larger context. *4.2 Emergence:* addresses that the system behavior can emerge from the agents and environment instead from equations or rules. *4.3 Adaptation:* identifies what decisions the agents make, in response to what stimuli. *4.4 Objectives:* defines the objective measure used to evaluate decision alternatives. *4.5 Learning:* describes agents that change how they produce adaptive behavior over time as a consequence of their experience. *4.6 Prediction:* explains how agents predict future conditions and decision consequences. *4.7 Sensing:* addresses information the agents use. *4.8 Interaction:* addresses how agents interact. *4.9 Stochasticity:* describes where and how stochastic processes – those driven by pseudorandom numbers – are used in the model. *4.10 Collectives:* describes aggregations of agents that affect, and are affected by, the agents. *4.11 Observation:* describes how information is collected.
Details	*5. Initialization* A description of how the model is initialized.
	6. Input data A definition of the input data that drives the ABMS.
	7. Submodels A description of any submodels present in the ABMS.

C. Data mining

This section presents the DM steps taken towards preparing the data to be applied in the QUESTS methodology.

Data cleaning

The first error (motion sensors' states different than "ON" or "OFF") was resolved by replacing the incorrect entry with the value we judged as the most plausible (for example, replacing "OFcF" with "OFF" or "O" with "ON").

Table 6.3: Errors found in the "Sensor state" column of the motion sensors.

Row	Date	Time	Sensor ID	Sensor state	Description
275,451	2010-12-13	10:00:12.46	M019	OFcF	
277,012	2010-12-13	12:34:07.55	M018	OFcF	
277,372	2010-12-13	12:57:14.32	M017	OFcF	
279,632	2010-12-13	20:03:37.82	M020	OcFF	
886,911	2011-02-23	15:28:52.87	M009	O	Relax begin
900,914	2011-02-25	18:13:47.29	M019	OF	Meal preparation end

In the case of temperature sensors, some values presented random letters along with the expected value, while others were found to be unrealistic, such as 225° C, 235° C, 245° C and 285° C. Other than that, a single temperature sensor is described as indicating the end of an activity, even though such devices are typically expected to report only the room temperature. We hypothesize that this activity end marker was added to the same entry because it represents the conclusion of the dataset, thereby providing a termination point for the last activity recorded.

Finally, Table 6.5 lists atypical patterns found in the dataset, where instead of displaying the sensor ID, a descriptive text was shown. Then, in the following column, there was either an "OFF" state (which led us to believe that, in this case, it referred to a motion sensor) or a number. We discarded such information.

Table 6.4: Atypical patterns from the temperature sensors.

Row	Date	Time	Sensor ID	Sensor state	Description
277,190	2010-12-13	12:38:55.75	T002	285	-
319,016	2010-12-18	20:32:58.68	T002	245	-
320,651	2010-12-18	22:09:01.76	T001	225	-
321,347	2010-12-18	23:55:11.16	T003	235	-
275,999	2010-12-13	10:32:33.63	T002	28.55c	-
276,914	2010-12-13	12:23:45.98	T004	26cc	-
1,719,558	2011-06-11	23:58:10.00	T002	25	Sleeping end

Table 6.5: Atypical patterns found in the dataset.

Row	Date	Time	Sensor ID	Sensor state
1,476,693	2011-05-10	18:42:45.16	c	OFF
1,523,044	2011-05-17	11:40:03.01	LEAVEHOME	180
1,523,584	2011-05-17	14:58:04.90	LEAVEHOME	300
1,526,729	2011-05-17	18:31:56.04	LEAVEHOME	300
1,530,032	2011-05-18	12:22:23.31	LEAVEHOME	300
1,530,061	2011-05-18	14:12:13.73	ENTERHOME	6592

As part of the sanity checks, we first verified that all initiated activities had corresponding endpoints – this condition was fully satisfied. Subsequently, to ensure no activities were overlooked, we compared the number of detected activities with the counts provided in the accompanying README file. The analysis showed a complete match across all activities, with the exception of “relaxing”, which was detected more frequently in our analysis than reported in the documentation. This discrepancy is likely due to a typographical error in the dataset annotations.

Focusing solely on the data collected from motion sensors, the dataset was further analyzed to identify any anomalous activation patterns –such as sensors that were activated but never deactivated, or instances where a sensor transitioned to a state it was already in (e.g., an “ON” sensor being activated again). Upon detection, such irregular entries were excluded from the dataset, resulting in the removal of a total of 64 records.

Additionally, three extreme cases were identified in the dataset concerning unusually long activation durations of motion sensors, all occurring on November 16, 2022. On this date, all sensors remained active for extended periods that deviated significantly from the

typical activation patterns observed. It is assumed that this anomaly resulted from a hardware malfunction affecting the devices during the night, as the irregular activity began shortly after midnight (00:04:09) and ceased before 9:00 a.m.

Table 6.6: *pvalues* resulted from different functions.

Sensor	normal	exp	Weibull	gamma	chi2
M001	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M002	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M003	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M004	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M005	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M006	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M007	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M008	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M009	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M010	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M011	0.00000000	0.00000017	**0.07269303**	0.03149707	0.03150784
M012	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M013	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M014	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M015	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M016	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M017	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M018	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M019	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M020	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M021	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M022	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M023	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M024	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M025	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M026	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M027	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M028	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M029	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M030	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
M031	0.00000000	0.00000000	0.00004517	0.00000000	0.00000000

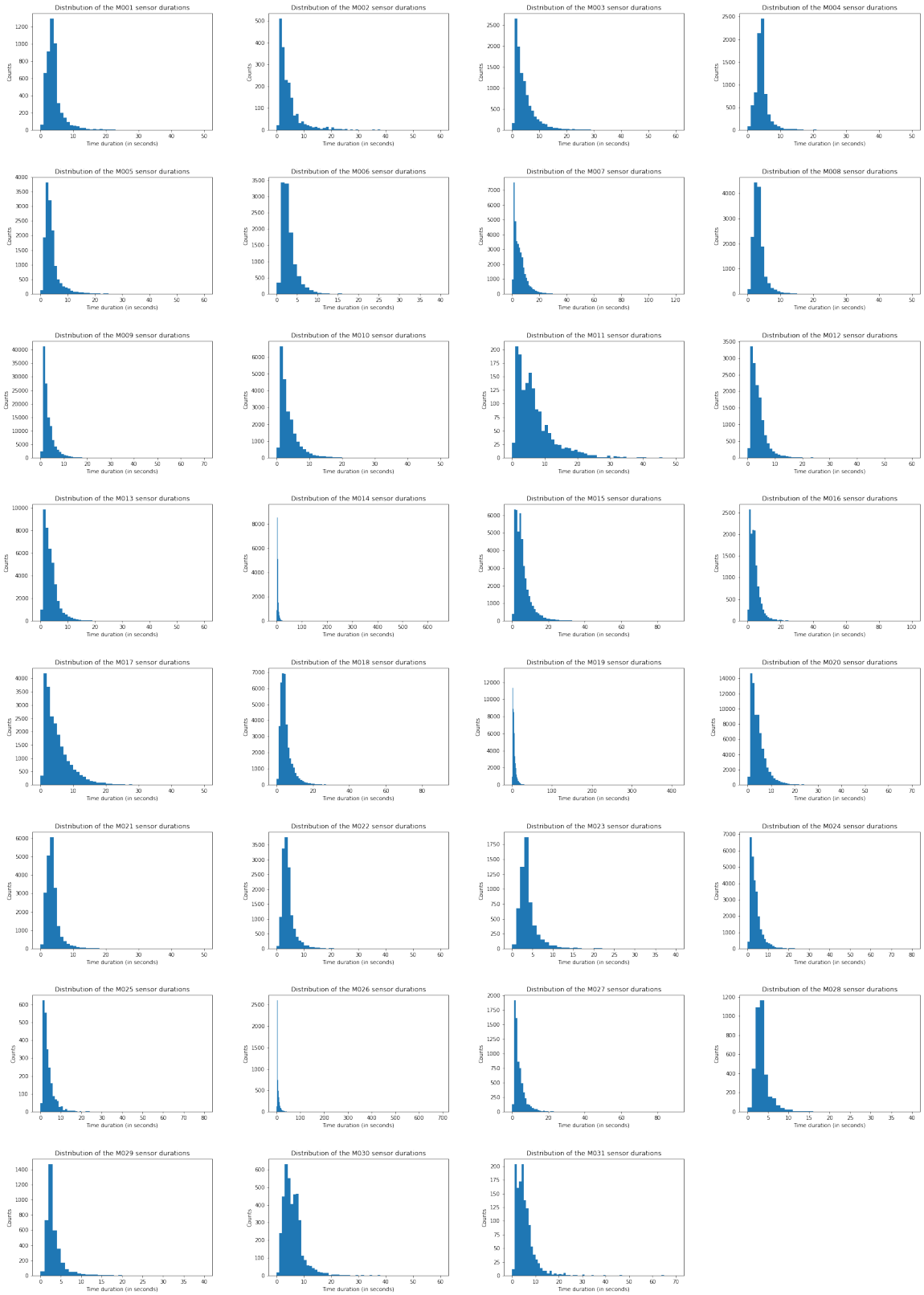


Figure 6.1: Motion sensors.

Table 6.7: Matrix comparing the solution outcome and the original source material.

0.04	0.02		0.03	0.02		0.04		0.01														0.01							
0.01	0.03	0.13		0.01		0.05							0.04									0.02							
	0.01	0.03		0.01		0.02																							
			0.05	0.01		0.05																0.01							
		0.03	0.06		0.03	0.10																		0.01					
						0.05	0.04												0.01										
0.02		0.01	0.02		0.02																								
				0.02	0.03	0.01	0.02		0.01										0.03		0.02								
								0.04	0.04																				
							0.04	0.01		0.01	0.04	0.01	0.02						0.04	0.01		0.01							
											0.03							0.01	0.03			0.02							
								0.02			0.01	0.01																	
												0.01	0.05						0.04	0.01									
																0.01	0.01		0.01										
														0.02	0.05	0.02		0.07											
																		0.02		0.01		0.01							
																	0.03	0.01	0.01		0.01								
														0.02		0.01		0.01											
							0.01						0.01																
							0.02						0.02				0.03		0.02	0.01									
								0.01										0.01	0.02	0.04	0.01	0.04	0.02					0.02		
						0.02			0.01	0.01									0.04	0.01		0.04	0.13		0.01		0.01	0.03		
								0.02										0.01					0.05							
								0.02				0.01	0.02					0.01	0.02	0.01			0.08	0.03	0.09	0.09	0.02			
								0.01																0.01	0.03	0.02	0.01			
						0.02		0.01											0.01	0.02			0.02	0.07	0.05	0.01	0.03			
			0.02			0.02								0.01					0.02	0.04	0.01	0.04	0.02		0.02	0.01	0.01		0.03	
						0.03							0.01				0.01		0.01	0.02	0.04	0.03					0.02	0.04	0.06	
				0.01																0.03	0.02						0.02	0.05	0.08	
													0.01						0.09	0.04	0.06		0.02						0.11

Bibliography

[1] Mark Weiser. "The computer for the 21st century". In: *SIGMOBILE Mob. Comput. Commun. Rev.* 3.3 (July 1999), pp. 3–11. ISSN: 1559-1662.

[2] Rebeca C. Motta, Káthia Marçal de Oliveira, and Guilherme H. Travassos. "An evidence-based roadmap for IoT software systems engineering". In: *Journal of Systems and Software* 201 (2023), p. 111680. ISSN: 0164-1212.

[3] Gary White, Vivek Nallur, and Siobhán Clarke. "Quality of Service Approaches in IoT: A Systematic Mapping". In: *Journal of Systems and Software* 132 (May 2017), pp. 186–203.

[4] Manuel Silverio, Suresh Renukappa, and Subashini Suresh. "What is a smart device? - a conceptualisation within the paradigm of the internet of things". In: *Visualization in Engineering* 6 (May 2018).

[5] Renat Faizrakhmanov, Mohammad Reza Bahrami, and Alexey Platunov. "Prototype, method, and experiment for evaluating usability of smart home user interfaces". In: *Computer Standards & Interfaces* 92 (2025), p. 103903. ISSN: 0920-5489.

[6] Liang Yu et al. "A Review of Deep Reinforcement Learning for Smart Building Energy Management". In: *IEEE Internet of Things Journal* PP (May 2021).

[7] Zaib Ullah et al. "Applications of Artificial Intelligence and Machine learning in smart cities". In: *Computer Communications* 154 (Mar. 2020).

[8] Moojan Ghafurian, Colin Ellard, and Kerstin Dautenhahn. “An investigation into the use of smart home devices, user preferences, and impact during COVID-19”. In: *Computers in Human Behavior Reports* 11 (June 2023), p. 100300.

[9] Haitham Samih. “Smart cities and internet of things”. In: *Journal of Information Technology Case and Application Research* 21 (Apr. 2019), pp. 1–10.

[10] Monica Anderson. “Smart device interest, perceived usefulness, and preferences in rural Alabama seniors”. In: *Scientific Data* 11 (May 2024).

[11] George Demiris et al. “Senior residents’ perceived need of and preferences for “smart home” sensor technologies”. In: *Int. J. of Technology Assessment in Health Care* 24.1 (2008), pp. 120–124.

[12] Arian Vrančić, Hana Zadravec, and Tihomir Orehovački. “The Role of Smart Homes in Providing Care for Older Adults: A Systematic Literature Review from 2010 to 2023”. In: *Smart Cities* 7.4 (2024), pp. 1502–1550. ISSN: 2624-6511.

[13] Ephraim Asiya et al. “Performance evaluation of planar dipole antenna for smart home applications”. In: *Heliyon* 9 (Aug. 2023), e19186.

[14] Nadine Andrea Felber et al. “Mapping ethical issues in the use of smart home health technologies to care for older persons: a systematic review”. In: *BMC Medical Ethics* 24.1 (2023).

[15] Olugbenga Timo Oladinrin et al. “Smart Home Technologies to Facilitate Ageing-in-Place: Professionals Perception”. In: *Sustainability (Switzerland)* 15.8 (2023).

[16] Rainara Maia Carvalho, Rossana Maria de Castro Andrade, and Káthia Marçal de Oliveira. “AQUArIUM - A Suite of Software Measures for HCI Quality Evaluation of Ubiquitous Mobile Applications.” In: *J. of Systems and Software* (2018), pp. 101–136.

[17] Robert Harmon, Enrique Castro-Leon, and Sandhiprakash Bhide. “Smart cities and the Internet of Things”. In: Aug. 2015, pp. 485–494.

[18] Alexandra Queirós et al. “Usability, accessibility and ambient-assisted living: A systematic literature review”. In: *Universal Access in the Information Society* 12 (Jan. 2013), pp. 1–10.

[19] C. Mao and D. Chang. “Review of cross-device interaction for facilitating digital transformation in smart home context: A user-centric perspective”. In: *Adv. Engineering Informatics* 57 (2023), p. 102087.

[20] Department of Economic United Nations and Population Division Social Affairs. “Population 2030: Demographic challenges and opportunities for sustainable development planning”. In: *Economical and Social Affairs* (2015).

[21] ISO/IEC. *25019:2023. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Quality-in-use model.* 2023.

[22] ISO/IEC. *25022:2016. Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use.* 2016. URL: https://www.iso.org/fr/standard/35746.html.

[23] *ISO 9241-11. Ergonomic requirements for office work with visual display terminals (VDT) s- Part 11 Guidance on usability.* Standard. Geneva, CH, 1998.

[24] E. Cioroaica, B. Buhnova, and T. Kuhn. “Predictive Simulation within the Process of Building Trust”. In: *2022 IEEE 19th Int. Conf. on Software Architecture Companion, ICSA-C 2022.* 2022, pp. 47–48.

[25] Y. Du et al. “A data-driven method for user satisfaction evaluation of smart and connected products”. In: *Expert Systems with Applications* 210 (2022), p. 118392.

[26] Claire McCallum, John Rooksby, and Cindy Gray. “Evaluating the Impact of Physical Activity Apps and Wearables: Interdisciplinary Review”. In: *JMIR mHealth and uHealth* 6 (Mar. 2018), e58.

[27] Torhild Holthe et al. “Usability and acceptability of technology for community-dwelling older adults with mild cognitive impairment and dementia: A systematic literature review”. In: *Clinical Interventions in Aging* 13 (2018), pp. 863–886.

[28] Ria Lyzara et al. “E-Government Usability Evaluation: Insights from A Systematic Literature Review”. In: *Proc. ACM 2nd Int. Conf. on Software Engineering and Information Management*. 2019, pp. 249–253.

[29] Inocencio Maramba, Arunangsu Chatterjee, and Craig Newman. “Methods of usability testing in the development of eHealth applications: A scoping review”. In: *Int. J. of Medical Informatics* 126 (2019), pp. 95–104.

[30] Yusufu Gambo and Muhammad Zeeshan Shakir. “Review on self-regulated learning in smart learning environment”. In: *Smart Learning Environments* 8.1 (2021).

[31] Lívia Luize Marengo and Silvio Barberato-Filho. “Involvement of Human Volunteers in the Development and Evaluation of Wearable Devices Designed to Improve Medication Adherence: A Scoping Review”. In: *Sensors* 23.7 (2023).

[32] S. Ntoa et al. “Uxami observer: An automated user experience evaluation tool for ambient intelligence environments”. In: *Advances in Intelligent Systems and Computing* 868 (2018), pp. 1350–1370.

[33] M. Pavlovic et al. “Determining a Framework for the Generation and Evaluation of Ambient Intelligent Agent System Designs”. In: *Adv. in Intelligent Systems and Computing* 1069 (2020), pp. 318–333.

[34] Ahlem Assila, Káthia Marçal de Oliveira, and Houcine Ezzedine. “Standardized Usability Questionnaires: Features and Quality Focus”. In: *Elec. J. of Comp. Sc. and Inf. Tech.* 6.1 (Dec. 2016), pp. 15–31.

[35] Georgia Dede et al. “Evaluation of end user requirements for Smart Home applications and services based on a decision support system”. In: *Internet of Things* 16 (July 2021), p. 100431.

[36] A. Schmidt. “Interactive context-aware systems interacting with ambient intelligence”. In: *Ambient intelligence* (2005), pp. 159–179.

[37] E. Bonabeau. "Agent-based modeling: Methods and techniques for simulating human systems". In: *Proceedings of the National Academy of Sciences of the United States of America* 99 (2002), pp. 7280–7287.

[38] Geeske Scholz et al. "Social Agents? A Systematic Review of Social Identity Formalizations". In: *Journal of Artificial Societies and Social Simulation* 26.2 (2023), p. 6. ISSN: 1460-7425.

[39] P. Windrum, G. Fagiolo, and A. Moneta. "Empirical validation of agent-based models: Alternatives and prospects". In: *J. of Artificial Societies and Social Simulation* 10.2 (2007).

[40] ISO/IEC. *25010:2011. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.* 2011.

[41] ISO/IEC. *25010:2023. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Product quality model.* 2011.

[42] ISO/IEC. *25023:2016. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality.* 2016.

[43] Sergio Salomón García et al. "Towards automatic evaluation of the Quality-in-Use in context-aware software systems". In: *Journal of Ambient Intelligence and Humanized Computing* 14 (Feb. 2022).

[44] ISO/IEC. *9241-11:2018. Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts.* 2018.

[45] ISO/IEC. *25063:2017. Systems and software engineering - Systems and software product Quality Requirements and Evaluation (SQuaRE) - Common Industry Format (CIF) for usability: Context of use description.* 2018.

[46] Stuart Russel and Peter Norvig. *Artificial intelligence - A modern approach. 4th ed.* 2021.

[47] Stan Franklin and Arthur Graesser. “Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents”. In: Jan. 1996, pp. 21–35.

[48] Nicholas R. Jennings, Katia Sycara, and Michael Wooldridge. *A Roadmap of Agent Research and Development*. 1998.

[49] W. Baldwin, Brian Sauser, and Robert Cloutier. “Simulation Approaches for System of Systems: Events-based versus Agent Based Modeling”. In: *Procedia Computer Science* 44 (Dec. 2015).

[50] Preston White and Ricki Ingalls. “Introduction to Simulation”. In: Dec. 2009, pp. 12–23.

[51] C. M. Macal. “Everything you need to know about agent-based modelling and simulation”. In: *Journal of Simulation* 10.2, SI (May 2016), pp. 144–156.

[52] Luca Serena et al. “A review of multilevel modeling and simulation for human mobility and behavior”. In: *Simulation Modelling Practice and Theory* 127 (May 2023), p. 102780.

[53] Adelinde M. Uhrmacher and Danny Weyns. *Multi-Agent System: Simulation and Applications*. Taylor & Francis Group, 2009. ISBN: 978-750-8400.

[54] Eric Silverman et al. “Situating agent-based modelling in population health research”. In: *Emerging Th. in Epidemiology* 18.1 (2021).

[55] Beth Barnes, Sarah Dunn, and Sean Wilkinson. “Replicating capacity and congestion in microscale agent-based simulations”. In: *Travel Behaviour and Society* 29 (2022), pp. 308–318. ISSN: 2214-367X.

[56] Nicholas Aiden Keller. “Towards Data-driven Simulation Modeling for Mobile Agent-based Systems”. Doctoral Dissertation. Georgia State University, 2017.

[57] Ying Zhou et al. “Using data-driven methods to detect financial statement fraud in the real scenario”. In: *International Journal of Accounting Information Systems* 54 (2024), p. 100693. ISSN: 1467-0895.

[58] Andrew Collins, Matthew Koehler, and Christopher Lynch. "Methods That Support the Validation of Agent-Based Models: An Overview and Discussion". In: *Journal of Artificial Societies and Social Simulation* 27.1 (2024), p. 11. ISSN: 1460-7425.

[59] Tamara Munzner. "A Nested Model for Visualization Design and Validation". In: *IEEE Transactions on Visualization and Computer Graphics* 15.6 (2009), pp. 921–928.

[60] Yanjun Gan et al. "A comprehensive evaluation of various sensitivity analysis methods: A case study with a hydrological model". In: *Environmental Modelling & Software* 51 (2014), pp. 269–285. ISSN: 1364-8152.

[61] Jordi Vallverdú. "What are Simulations? An Epistemological Approach". In: *Procedia Technology* 13 (2014). SLACTIONS 2013: Research conference on virtual worlds – Learning with simulations, pp. 6–15. ISSN: 2212-0173.

[62] Pedro Silva et al. "Simulation - Concepts and Applications". In: Sept. 2010, pp. 429–434. ISBN: 978-3-642-16323-4.

[63] Sharad Sharma et al. "Modeling human behavior during emergency evacuation using intelligent agents: A multi-agent simulation approach". In: *Information Systems Frontiers* 20.4, SI (Aug. 2018), pp. 741–757.

[64] Rouba Iskandar et al. "Agent-based simulation of seismic crisis including human behavior: application to the city of Beirut, Lebanon". In: *Simulation* 100.4 (2024), pp. 357–377.

[65] Jaekoo Joo et al. "Agent-based simulation of affordance-based human behaviors in emergency evacuation". In: *Simulation Modelling Practice and Theory* 32 (Mar. 2013), pp. 99–115.

[66] Edoardo Battegazzorre et al. "IdealCity: A hybrid approach to seismic evacuation modeling". In: *Advances in Engineering Software* 153 (Mar. 2021).

[67] Sharad Sharma, Harpreet Singh, and Atul Prakash. "Multi-agent modeling and simulation of human behavior in aircraft evacuations". In: *IEEE Transactions on Aerospace and Electronic Systems* 44 (Nov. 2008), pp. 1477–1488.

[68] Hossein Moradi et al. “A Framework for Constructing Agent-based Models for Evaluating Emergency Evacuation Procedures; A Case Study Applied to Egaleo, Greece”. In: *58th Hawaii International Conference on System Sciences*. Vol. Proceedings of the 58th Hawaii International Conference on System Sciences. Hawaii, United States, Jan. 2025.

[69] E. Bakshy and U. Wilensky. *NetLogo Team Assembly model*. 2007. URL: http://ccl.northwestern.edu/netlogo/models/TeamAssembly.

[70] Minyoung Choi and Jae-Suk Yang. “Exploring the Complexities of Negotiation: Strategies for Successful Intra- and Inter-Team Negotiation in Organizations”. In: *Journal of Artificial Societies and Social Simulation* 27.3 (2024), p. 4. ISSN: 1460-7425.

[71] Jan Drchal, Michal Čertický, and Michal Jakob. “Data-driven activity scheduler for agent-based mobility models”. In: *Transportation R. Part C: Emerging Tech.* 98 (2019), pp. 370–390.

[72] C. Macal and Michael North. “Agent-based modeling and simulation: ABMS examples”. In: Dec. 2008, pp. 101–112.

[73] Franziska Klügl and Ana Bazzan. “Agent-Based Modeling and Simulation”. In: *AI Magazine* 33 (Sept. 2012), pp. 29–40.

[74] U. Wilensky. *NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.* http://ccl.northwestern.edu/netlogo/. 1999.

[75] Arnaud Saval et al. “Dealing with mixed and non-normative traffic. An agent-based simulation with the GAMA platform”. In: *PLoS ONE* 18.3 March (2023).

[76] Michael J. North, Nicholson T. Collier, and Jerry R. Vos. “Experiences creating three implementations of the repast agent modeling toolkit”. In: *ACM Trans. Model. Comput. Simul.* 16.1 (Jan. 2006), pp. 1–25. ISSN: 1049-3301.

[77] Kostiantyn Cherednichenko et al. “Modelling and Optimization of Airport Security Screening System with AnyLogic Simulation: A Case of Dublin Airport”. In: *Lecture Notes in Intelligent Transportation and Infrastructure* Part F230 (2025), pp. 381–397.

[78] Giuseppe D’Ambrosio and Sean Luke. “Hybrid Agent-Based and Discrete Event Simulation in Mason”. In: 2023, pp. 84–94.

[79] Nelson Minar et al. “The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations”. In: *Santa Fe Institute Working Paper* 96-06-042 (July 1996).

[80] Aisling J. Daly et al. “Quo vadis, agent-based modelling tools?” In: *Environmental Modelling & Software* 157 (2022), p. 105514. ISSN: 1364-8152.

[81] Romain Reulier, Gregory Bonnet, and Florentin L’Homme. “Comparison of execution times of three multi-agent platforms (Netlogo/GAMA/Repast simphony) based on the LASCAR model”. In: *Journal of Simulation* 18.6 (2024), pp. 1128–1138.

[82] Ahmed Laatabi et al. “ODD+2D: An ODD based protocol for mapping data to empirical ABMs”. In: *Journal of Artificial Societies and Social Simulation* 21 (Mar. 2018).

[83] Volker Grimm et al. “A standard protocol for describing individual-based and agent-based models”. In: *Ecological Modelling* 198.1 (2006), pp. 115–126. ISSN: 0304-3800.

[84] Volker Grimm et al. “The ODD Protocol for Describing Agent-Based and Other Simulation Models: A Second Update to Improve Clarity, Replication, and Structural Realism”. In: *J. of Artificial Societies and Social Simulation* 23.2 (2020), p. 7. ISSN: 1460-7425.

[85] Carole Adam and Benoit Gaudou. “BDI agents in social simulations: a survey”. In: *Knowledge Engineering Review* 31.3 (2016), pp. 207–238.

[86] Miguel Baigorri and Fermin Mallor. “Including learning and forgetting processes in Agent-Based simulation models: Application to police intervention in out-of-hospital cardiac arrests”. In: *Expert Systems with Applications* 260 (2025).

[87] Alice Micolier et al. “Li-BIM, an agent-based approach to simulate occupant-building interaction from the Building-Information Modelling”. In: *Engineering Applications of Artificial Intelligence* 82 (2019), pp. 44–59. ISSN: 0952-1976.

[88] Fatima Abdallah, Shadi Basurra, and Mohamed Medhat Gaber. “A Hybrid Agent-Based and Probabilistic Model for Fine-Grained Behavioural Energy Waste Simulation”. In: *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*. 2017, pp. 991–995.

[89] Elie Azar and Carol Menassa. “A conceptual framework to energy estimation in buildings using agent based modeling”. In: Dec. 2010, pp. 3145–3156.

[90] Matteo Richiardi et al. “A Common Protocol for Agent-Based Social Simulation”. In: *Journal of Artificial Societies and Social Simulation* 9 (Jan. 2006).

[91] F. Veronese et al. “Realistic human behaviour simulation for quantitative ambient intelligence studies”. In: *Technology and Disability* 28.4 (2017), pp. 159–177.

[92] Jennifer Renoux and Franziska Klugl. “SIMULATING DAILY ACTIVITIES IN A SMART HOME FOR DATA GENERATION”. In: *2018 Winter Simulation Conference (WSC)*. 2018, pp. 798–809.

[93] Alejandro Guerra-Hernández, Amal Seghrouchni, and Henry Soldano. “Distributed Learning in Intentional BDI Multi-Agent Systems.” In: *Proc. of the Fifth Mexican Int. Conf. in CS*. 2004, pp. 225–232.

[94] Yves Wautelet and Manuel Kolp. “Business and Model-Driven Development of BDI Multi-Agent Systems”. In: *Neurocomputing* 182 (Dec. 2015).

[95] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005. ISBN: 0120884070.

[96] Afrasiyab Khan et al. “Data Mining in Healthcare: An Overview of Applications, Techniques, and Challenges”. In: *International Journal of Artificial Intelligence & Mathematical Sciences* 3 (May 2025), pp. 1–10.

[97] Barry Leventhal. "An introduction to data mining and other techniques for advanced analytics". In: *Journal of Direct, Data and Digital Marketing Practice* 12 (Oct. 2010).

[98] Dimple Patil et al. "Machine learning and deep learning: Methods, techniques, applications, challenges, and future research opportunities". In: Oct. 2024, pp. 28–81. ISBN: 978-81-981367-4-9.

[99] Guangqi Chen and Zheng Han. "The rise of deep learning: AI and engineering applications under the spotlight of the 2024 Nobel prize". In: *Intelligent Geoengineering* 2.1 (2025), pp. 14–21. ISSN: 3050-6190.

[100] Maximilian Weisenseel et al. "Process Mining on Distributed Data Sources". In: (June 2025).

[101] Wil Aalst. "Process Mining: Overview and Opportunities". In: *ACM Transactions on Management Information Systems* 3 (July 2012), pp. 7.1–7.17.

[102] Wil Aalst et al. "Process Mining Manifesto". In: vol. 99. Aug. 2011, pp. 169–194. ISBN: 978-3-642-28107-5.

[103] Lingkai Yang et al. "Using semi-Markov models to identify long holding times of activities of daily living in smart homes". In: Aug. 2023, pp. 748–753.

[104] Lingkai Yang et al. "A multi-components approach to monitoring process structure and customer behaviour concept drift". In: *Expert Systems with Applications* 210 (Aug. 2022), p. 118533.

[105] Sally McClean et al. "Using Semi-Markov Models for Generating, Validating, and Analyzing Artificial Smart Home Processes". In: Dec. 2024, pp. 300–312. ISBN: 978-3-031-77570-3.

[106] Ronny S. Mans et al. "Application of Process Mining in Healthcare – A Case Study in a Dutch Hospital". In: vol. 25. Jan. 2008, pp. 425–438. ISBN: 978-3-540-92218-6.

[107] Tim R. de Boer et al. “Process Mining on National Health Care Data for the Discovery of Patient Journeys of Older Adults”. In: *Journal of the American Medical Directors Association* 26.1 (2025), p. 105333. ISSN: 1525-8610.

[108] Greg Van Houdt et al. “An empirical evaluation of unsupervised event log abstraction techniques in process mining”. In: *Information Systems* 121 (2024), p. 102320. ISSN: 0306-4379.

[109] Rob H. Bemthuis and Sanja Lazarova-Molnar. “Towards integrating process mining with agent-based modeling and simulation: State of the art and outlook”. In: *Expert Systems with Applications* 281 (2025), p. 127571.

[110] Wil Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Vol. 136. Jan. 2011. ISBN: 978-3-642-19344-6.

[111] Jonathan Brock et al. “Process Mining Data Canvas: A method to identify data and process knowledge for data collection and preparation in process mining projects”. In: *Procedia CIRP* 119 (2023). The 33rd CIRP Design Conference, pp. 602–607. ISSN: 2212-8271.

[112] Wil Aalst. *Process Mining: Data Science in Action*. Jan. 2016. ISBN: 9783662498507.

[113] Hamdi Kavak et al. “Big data, agents, and machine learning: towards a data-driven agent-based modeling approach”. In: *Proceedings of the Annual Simulation Symposium*. Baltimore, Maryland: Society for Computer Simulation International, 2018.

[114] Franziska Klügl. “A validation methodology for agent-based simulations”. In: *Proceedings of the 2008 ACM Symposium on Applied Computing*. Fortaleza, Ceara, Brazil: ACM, 2008, pp. 39–43.

[115] Andrew Collins, Matthew Koehler, and Christopher Lynch. “Methods That Support the Validation of Agent-Based Models: An Overview and Discussion”. In: *Journal of Artificial Societies and Social Simulation* 27.1 (2024), p. 11.

[116] Yakup Turgut and Cafer Erhan Bozdag. "A framework proposal for machine learning-driven agent-based models through a case study analysis". In: *Simulation Modelling Practice and Theory* 123 (2023), p. 102707. ISSN: 1569-190X.

[117] Badis Hammi et al. "Survey on smart homes: Vulnerabilities, risks, and countermeasures". In: *Computers & Security* 117 (2022), p. 102677. ISSN: 0167-4048.

[118] Luca Sabatucci et al. "A model for automatic selection of IoT services in ambient assisted living for the elderly". In: *Pervasive and Mobile Computing* 95 (2023), p. 101845.

[119] Shrikant Tiwari Amit Kumar Tyagi Shabnam Kumari. "Internet of Things for Smart Healthcare: A Survey". In: 2024, pp. 19–41.

[120] Lenin Erazo et al. "Quality Assessment Approaches for Ambient Assisted Living Systems: A Systematic Review". In: *Applied Technologies*. Springer International Publishing, 2020, pp. 421–439.

[121] Stuart McDonald, Dave Towey, and Vladimir Brusic. "Social Impact of Smart Environments: Software Engineering Perspectives and Challenges". In: *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*. 2022, pp. 1592–1597.

[122] N.K. Suryadevara et al. "Forecasting the behavior of an elderly using wireless sensors data in a smart home". In: *Engineering Applications of Artificial Intelligence* 26.10 (2013), pp. 2641–2652. ISSN: 0952-1976.

[123] Sara Kebir. "Approches légères pour une solution de maison intelligente pour personnes âgées". Doctoral Dissertation. Université d'Artois, 2023.

[124] A.C. Tricco et al. *Rapid reviews to strengthen health policy and systems: a practical guide*. World Health Organization, 2017, xix, 119 p.

[125] Barbara Ann Kitchenham, David Budgen, and Pearl Brereton. *Evidence-Based Software Engineering and Systematic Reviews*. Chapman & Hall/CRC, 2015. ISBN: 1482228653.

[126] A.C. Tricco et al. “A scoping review on the conduct and reporting of scoping reviews”. In: *BMC medical research methodology* 16.1 (2016), pp. 1–10.

[127] B. Cartaxo, G.H. Lima Pinto, and S.C. Branco Soares. “Rapid Reviews in Software Engineering”. In: *ArXiv* abs/2003.10006 (2020).

[128] Maria Paula Corrêa Angeloni, Káthia Marçal de Oliveira, and Emmanuelle Grislin-Le Strugeon. “A First Literature Study on Predictive Quality in Use Evaluation for Smart Environments”. In: *Engineering Interactive Computer Systems. EICS 2023 International Workshops and Doctoral Consortium*. Cham: Springer Nature Switzerland, 2024, pp. 3–10. ISBN: 978-3-031-59235-5.

[129] Maria Paula Corrêa Angeloni et al. “A Review on Quality in Use Evaluation of Smart Environment Applications: What’s Next?” In: *Companion of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS ’23 Companion), June 27–30, 2023, Swansea, United Kingdom*. 2023. ISBN: 979-8-4007-0206-8.

[130] Maria Paula Corrêa Angeloni et al. “A Tertiary Study on Quality in Use Evaluation of Smart Environment Applications”. In: *Research Challenges in Information Science*. Ed. by João Araújo et al. Cham: Springer Nature Switzerland, 2024, pp. 115–130. ISBN: 978-3-031-59465-6.

[131] M. Petticrew and H. Roberts. *Systematic reviews in the social sciences: A practical guide*. Oxford, UK: Blackwell Publishing Ltd, 2006.

[132] Emmanuelle Grislin-Le Strugeon et al. “A systematic mapping study on agent mining”. In: *Journal of Experimental & Theoretical Artificial Intelligence* 34 (Jan. 2021), pp. 1–26.

[133] D. Ghosh, A. Olewnik, and K. Lewis. “Application of Feature-Learning Methods Toward Product Usage Context Identification and Comfort Prediction”. In: *Journal of Computing and Information Science in Engineering* 18.1 (2018).

[134] S. D. Patil et al. “Predictive asset availability optimization for underground trucks and loaders in the mining industry”. In: *OPSEARCH* 58.3 (2021), pp. 751–772.

[135] K. Petersen and C. Gencel. "Worldviews, Research Methods, and their Relationship to Validity in Empirical Software Engineering Research". In: *23rd Int. W. on Software Measurement and the 8th Int. Conf. on Software Process and Product Measurement*. Oct. 2013, pp. 81–89.

[136] Rebeca C. Motta, Káthia M. de Oliveira, and Guilherme H. Travassos. "A conceptual perspective on interoperability in context-aware software systems". In: *Information and Software Technology* 114 (2019), pp. 231–257. ISSN: 0950-5849.

[137] M. J. O'Grady, G. M. P. O'Hare, and S. Poslad. "Smart environment interaction: A user assessment of embedded agents". In: *J. of Ambient Intelligence and Smart Environments* 5.3 (2013), pp. 331–346.

[138] S. Karnouskos et al. "Assessing the Integration of Software Agents and Industrial Automation Systems with ISO/IEC 25010". In: *Proc. IEEE 16th Int. Conf. on Indus. Informatics*. 2018, pp. 61–66.

[139] C. Rizopoulos et al. "Evaluating the user experience of a mobile user in a smart city context". In: *Int. J. Intell. Eng. Inform.* 3.2-3, SI (2015), pp. 120–143.

[140] S. Möller et al. "Chapter 14 - Evaluation of Multimodal Interfaces for Ambient Intelligence". In: *Human-Centric Interfaces for Ambient Intelligence*. Oxford: Academic Press, 2010, pp. 347–370.

[141] A. Cesta et al. "The ROBOCARE project: The user's view". In: *Assistive Technology Research Series* 22 (2008), pp. 85–104.

[142] M. A. Walker et al. "PARADISE: A framework for evaluating spoken dialogue agents". In: *35th Annual Meeting of the A. for Comp. Linguistics and 8th Conf. of the Eu. Chapter of the A. for Comp. Linguistics*. 1997, pp. 271–280.

[143] G. Li et al. "Reliability evaluation of integrated energy systems based on smart agent communication". In: *Applied Energy* 167 (2016), pp. 397–406.

[144] J. Carbo, N. Sanchez-Pi, and J. M. Molina. "Agent-based simulation with NetLogo to evaluate ambient intelligence scenarios". In: *J. of Simulation* 12.1 (2018), pp. 42–52.

[145] Z. Lin and L. Dong. “Clarifying Trust in Social Internet of Things”. In: *IEEE Trans. on Knowledge and Data Engineering* 30.2 (2018), pp. 234–248.

[146] L. Dybkjær, N. O. Bernsen, and W. Minker. “Evaluation and usability of multimodal spoken language dialogue systems”. In: *Speech Communication* 43.1-2 (2004), pp. 33–54.

[147] J. Kirakowski and M. Corbett. “SUMI: the Software Usability Measurement Inventory”. In: *British Journal of Educational Technology* 24 (Oct. 2006), pp. 210–212.

[148] M. Hassenzahl, M. Burmester, and F. Koller. “AttrakDiff: Ein Fragebogen zur Messung wahrgenommener hedonischer und pragmatischer Qualität”. In: *Mensch & Computer* (2003), pp. 187–196.

[149] J. Brooke. “SUS: A quick and dirty’ usability scale”. In: *Usability Evaluation in Industry* (1996), pp. 189–194.

[150] K. S. Hone and R. Graham. “Towards a tool for the Subjective Assessment of Speech System Interfaces (SASSI)”. In: *Natural Language Engineering* (2000), pp. 287–303.

[151] C. Wohlin. “Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering”. In: *Proc. of the 18th Int. Conf. on Evaluation and Assessment in Software Engineering*. London, UK: A. for Computing Machinery, 2014. ISBN: 9781450324762.

[152] Barbara Kitchenham et al. “Systematic literature reviews in software engineering – A systematic literature review”. In: *Information and Software Technology* 51.1 (2009), pp. 7–15.

[153] Dolors Costal et al. “How Tertiary Studies perform Quality Assessment of Secondary Studies in Software Engineering”. In: *XXIV Iberoamerican Conf. on Software Eng. (ESELAW@CIbSE)* (2021).

[154] Maria Paula Corrêa Angeloni et al. *Research protocol and results for A Tertiary Study on Quality in Use Evaluation of Smart Environment Applications*. Mar. 2024. URL: https://hal.science/hal-04507278.

[155] Maria Paula Corrêa Angeloni et al. *Initial results for A Tertiary Study on Quality in Use Evaluation of Smart Environment Applications*. Mar. 2024. URL: https://hal.science/hal-04506543.

[156] Ainhoa Apraiz, Ganix Lasa, and Maitane Mazmela. "Evaluation of User Experience in Human–Robot Interaction: A Systematic Literature Review". In: *Int. J. of Social Robotics* 15.2 (2023), pp. 187–210.

[157] Rute Bastardo et al. "Methodological quality of user-centered usability evaluation of ambient assisted living solutions: A systematic literature review". In: *Int. J. of Environmental Research and Public Health* 18.21 (2021).

[158] Ignacio Díaz-Oreiro et al. "UX Evaluation with Standardized Questionnaires in Ubiquitous Computing and Ambient Intelligence: A Systematic Literature Review". In: *Advances in Human-Computer Interaction* 2021 (2021).

[159] Mostafa Haghi Kashani et al. "A systematic review of IoT in healthcare: Applications, techniques, and trends". In: *J. of Network and Computer Applications* 192 (2021).

[160] Petra Maresova et al. "Health-Related ICT Solutions of Smart Environments for Elderly-Systematic Review". In: *IEEE Access* 8 (2020), pp. 54574–54600.

[161] Rute Bastardo, João Pavão, and Nelson Pacheco Rocha. "A Scoping Review of the Inquiry Instruments Being Used to Evaluate the Usability of Ambient Assisted Living Solutions". In: *Proc. of the 15th INSTICC Int. Joint Conf. on Biomedical Eng. Systems and Tech. - vol. 5: HEALTHINF*. 2021, pp. 320–327.

[162] Mohamed-Amine Choukou et al. "Evaluating the acceptance of ambient assisted living technology (AALT) in rehabilitation: A scoping review". In: *Int. J. of Medical Informatics* 150 (2021).

[163] Sara Colantonio et al. "Are Active and Assisted Living applications addressing the main acceptance concerns of their beneficiaries? Preliminary insights from a scop-

ing review". In: *Proc. of the 15th ACM Int. Conf. on Pervasive Tech. Related to Assistive Environments*. 2022, pp. 414–421.

[164] Wendy Moyle, Jenny Murfield, and Katarzyna Lion. "The effectiveness of smart home technologies to support the health outcomes of community-dwelling older adults living with dementia: A scoping review". In: *Int. J. of Medical Informatics* 153 (2021).

[165] Vanessa Nasr and Maryam Zahabi. "Usability Evaluation Methods of Indoor Navigation Apps for People with Disabilities: A Scoping Review". In: *Proc. of the 2022 IEEE Int. Conf. on Human-Machine Systems*. 2022, pp. 1–6.

[166] Gabriella Tónay et al. "Methods to assess the effectiveness and acceptance of information and communication technology-based assistive technology for older adults: A scoping review". In: *Int. J. of Rehabilitation Research* 46.2 (2023), pp. 113–125.

[167] Guilherme E. Konopatzki Filho, Guilherme Corredato Guerino, and Natasha M. C. Valentim. "A Systematic Mapping Study on Usability and User EXperience Evaluation of Multi-Touch Systems". In: *Proc. ACM 21st Brazilian Symp. on Human Factors in Computing Systems*. 2022.

[168] Joseane O.V. Paiva, Rossana M.C. Andrade, and Rainara Maia Carvalho. "Evaluation of Non-Functional Requirements for IoT Applications". In: *Int. Conf. on Enterprise Information Systems, ICEIS - Proceedings*. Vol. 2. 2021, pp. 111–119.

[169] Maria Saleemi, Maria Anjum, and Mariam Rehman. "Ubiquitous healthcare: a systematic mapping study". In: *J. of Ambient Intelligence and Humanized Computing* 14.5 (2023), pp. 5021–5046.

[170] ISO/IEC. *9126. Software engineering — Product quality*. 2001. URL: https://www.iso.org/standard/22749.html.

[171] ISO/IEC. *25012:2008. Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Data quality model*. 2008.

[172] ISO/IEC. *14598-5:1998. Information technology — Software product evaluation — Part 5: Process for evaluators*. 1998.

[173] ISO/IEC. *25000: Systems and Software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*. 2011.

[174] Stéphane Galland et al. *Environment Model for Multiagent-Based Simulation of 3D Urban Systems*. Apr. 2013.

[175] Hanjiao Wang, Yongbin Zhang, and Xiuli Fu. "Innovative applications of black box testing: a new strategy for teaching assessment in higher education". In: *IEEE Access* PP (Jan. 2025), pp. 1–1.

[176] Káthia Marçal de Oliveira et al. "Human-centred design process when users have an evolving profile: a case study in Parkinson's disease domain". In: *Proc. ACM Hum.-Comput. Interact.* 9.4 (June 2025).

[177] Káthia Marçal de Oliveira et al. "ParkinsonCom Project: Towards a Software Communication Tool for People with Parkinson's Disease". In: *Universal Access in Human-Computer Interaction. Design Methods and User Experience: 15th International Conference, UAHCI 2021, Held as Part of the 23rd HCI International Conference, HCII 2021, Virtual Event, July 24–29, 2021, Proceedings, Part I*. Berlin, Heidelberg: Springer-Verlag, 2021, pp. 418–428. ISBN: 978-3-030-78091-3.

[178] ParkinsonCom. *ParkinsonCom*. 2020. URL: https://parkinsoncom.eu.

[179] UCB France. *Le Parkinson's Well-Being Map*. 2016. URL: https://www.ucb-france.fr/documents/Well-Being-Map_FR.pdf.

[180] Maria Paula Corrêa Angeloni et al. "Using Agent-Based Modeling and Simulation for Quality in Use Evaluation of Ambient Assisted Living Applications". In: *Proc. of the International Conference on Ubiquitous Computing and Ambient Intelligence*. 2024, pp. 151–162.

[181] Seth Tisue and Uri Wilensky. *NetLogo: A simple environment for modeling complexity*. https://api.semanticscholar.org/CorpusID:196182066. 2014.

[182] School of Electrical Engineering and Washington State University Computer Science. *CASAS: Center of Advanced Studies in Adaptive Systems*. 2011. URL: https://casas.wsu.edu/datasets/.

[183] Simon Walk et al. "How to apply Markov chains for modeling sequential edit patterns in collaborative ontology-engineering projects". In: *International Journal of Human-Computer Studies* 84 (2015), pp. 51–66. ISSN: 1071-5819.

[184] Parviz Asghari, Elnaz Soleimani, and Ehsan Nazerfard. "Online human activity recognition employing hierarchical hidden Markov models". In: *Journal of Ambient Intelligence and Humanized Computing* 11 (Mar. 2020).

[185] Tim van Kasteren, Gwenn Englebienne, and B. Krose. "Activity recognition using semi-Markov models on real world smart home datasets". In: *JAISE* 2 (Jan. 2010), pp. 311–325.

[186] Stella Pachidi, Marco Spruit, and Inge van de Weerd. "Understanding users' behavior with software operation data mining". In: *Computers in Human Behavior* 30 (2014), pp. 583–594. ISSN: 0747-5632.

[187] Marc Jaxa-Rozen and Jan H. Kwakkel. "PyNetLogo: Linking NetLogo with Python". In: *J. of Artificial Societies and Social Simulation* 21.2 (2018), p. 4. ISSN: 1460-7425.

[188] Maria Paula Corrêa Angeloni et al. *QUESTS: Quality in Use Evaluation of Smart environment Technologies by agent-based Simulations*. Submitted to the Journal of Systems and Software. 2025.

[189] James R. Lewis. "IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use". In: *International Journal of Human-Computer Interaction* 7 (Feb. 1995), pp. 57–.

[190] Adelfa Acala and Hidear Talirongan. "Assessing user satisfaction and usability of a university portal: a quantitative study utilizing the computer system usability questionnaire (CSUQ)". In: 14 (Oct. 2023), pp. 408–415.

[191] Claes Wohlin et al. *Experimentation in Software Engineering*. Springer, May 2012. ISBN: 978-3-642-29043-5.

[192] Rebeca Motta, Valéria Silva, and Guilherme Travassos. "Towards a more in-depth understanding of the IoT Paradigm and its challenges". In: *Journal of Software Engineering Research and Development* 7 (Aug. 2019), p. 3.

[193] Danny Weyns et al. "Environments in multiagent systems". In: *The Knowledge Engineering Review* 20 (June 2005), pp. 127–141.

[194] Xin Liu et al. "A systematic review of digital twin about physical entities, virtual models, twin data, and applications". In: *Advanced Engineering Informatics* 55 (2023), p. 101876. ISSN: 1474-0346.

[195] Chen Gao et al. "Large language models empowered agent-based modeling and simulation: a survey and perspectives". In: *Humanities and Social Sciences Communications* 11.1 (Sept. 2024), p. 1259. ISSN: 2662-9992.

[196] An Zhang et al. "On Generative Agents in Recommendation". In: *Proc. of the 47th International ACM SIGIR Conf. on Research and Development in Information Retrieval*. 2024, pp. 1807–1817.