Assuring Runtime Quality Requirements for AI-Based Components

Dan Chen[1], Jingwei Yang[1(✉)], Shuwei Huang[2], and Lin Liu[2]

[1] BNU-HKBU United International College, Zhuhai, China
{chendan,jingweiyang}@uic.edu.cn
[2] Tsinghua University, Beijing, China
hsw22@mails.tsinghua.edu.cn, linliu@tsinghua.edu.cn

Abstract. As Artificial Intelligence makes astonishing progress, various AI-embedded applications are being built to unleash their potential. However, all technologies come with their inherent limitations in dealing with unanticipated situations, making it difficult to assure the satisfaction of critical qualities at runtime. This is partly due to the challenge of specifying requirements for quality-critical AI-based components. We argue that for a deployed AI model whose accuracy cannot be validated at runtime, an accuracy-centric specification method is not good enough to support AI application engineering practice. To address this fundamental issue, requirements engineering techniques can help, especially an NFRs-based approach has been proposed for mitigating the impacts of two types of errors caused by uncertainties, so that critical qualities can be assured in the specification of AI-based components. We have implemented our strategy by a combined use of requirements analysis techniques, including modelling goals as in goal-oriented RE and modelling of environment and problems as in problem-oriented RE. We have showcased its application on a Facial Recognition Payment (FRP) system. This work could help create a runtime engineering shield for AI-based components and move forward its application in quality essential scenarios.

Keywords: AI-based Component · Non-function Requirements · Quality Assurance · Requirements Specification · Uncertainty

1 Introduction

Recently, AI technologies have experienced unprecedented advances in systems and algorithms research and have influenced various research areas [1]. However, despite the enormous effort devoted to AI theoretical and applied research, the technologies of AI still possess their own limitations, including difficulties in dealing with uncertainties [2]. The uncertainties can be ascribed to many reasons, originating from the changing environment to the black-box nature of various AI models. Hence, they were questioned due to unpredictable [3] and unexplainable [4]. Such technological limitations bring up a common engineering challenge to build AI into product functions or features, which

G. Guizzardi et al. (Eds.): CAiSE 2024, LNCS 14663, pp. 319–335, 2024.
https://doi.org/10.1007/978-3-031-61057-8_19

is the quality assurance process of these AI-based components [5]. To tackle this issue, much research effort has been spent on AI model calibration and uncertainty estimation [6], and also on identifying uncertainties in a given domain, safety-driven modeling and analysis, and formal verification or quantitative evaluation approaches. However, a generally applicable methodology to ensure the satisfaction of critical qualities for a given domain, such as the safety and reliability of AI-based components at runtime, is not in place yet. On the other hand, driven by the potential social or economic interests, products with AI-based component(s) have been released to the mass market. As a result of inadequate methodological engineering guidance, the technological limitations of AI have been exposed and magnified over time in various application domains. For example, digital addiction is caused by inappropriate use of predictive analytics [7], and catastrophic accidents take place with Autopilot navigating in an unanticipated scenario [8]. As stated in [9], "*the rush to deploy AI-enabled systems just increased the urgency of making progress on how to model, analyze, and safeguard against the inherent uncertainty of our systems*".

From a Requirements Engineering (RE) perspective, given that the primary focus of RE lies in system analysis conducted before the commencement of system construction, RE has emerged as the most challenging endeavor within the context of building an AI-based system, and the uncertainty introduced by AI is a key contributory factor underlying this challenge [10]. To analyze the intrinsic uncertainty of AI during the RE phase, a possible method is to apply the obstacle analysis within the KAOS [11], while treating the AI uncertainty as obstacles to system goals. However, a requirements analysis and modeling technique that has been customized to address AI's uncertainty with a wholistic view is still in need [5]. Ishikawa et al. [12] investigated the influence of AI uncertainty on goal-oriented requirements analysis (GORE) techniques, with a focus on decision-making during the requirements analysis. In [13], the ramifications of AI uncertainty on arguments were explored, and potential support through Goal Structuring Notation (GSN) was introduced in response to these impacts. In addition, there have been some studies on training data uncertainty for AI-based systems [14, 15]. These studies either proposed conceptual high-level ideas that have not been materialized, or failed to emphasize the aspect of quality assurance when dealing with uncertainty. So far, there has not been sufficient intellectual support from the RE community on approaches to mitigating the impacts of uncertainty within AI-based components, to achieve an acceptable level of quality assurance and meet the pragmatic needs of the end-users and stakeholders. Hence, in this work, we choose to analyze AI uncertainty with the quality aspects of the system that AI models are built into and try to address uncertainty with a global view so that all critical qualities are satisfied at the system level at the same time.

The structure of this paper is organized as follows: Sect. 2 briefs on different methods to deal with uncertainty in AI, from the perspectives of AI model, quality assurance, requirements engineering, and obstacle analysis. Section 3 presents the context, rationale, and procedure of our proposed Non-Functional Requirements (NFRs)-based approach to mitigating the impact of uncertainties within the specification of AI-based components. Section 4 illustrates our approach with a Facial Recognition Component in a Mobile Banking App. Section 5 concludes this work and discusses its limitation and potential future extension.

2 Related Work

2.1 Uncertainties in AI

In the AI domain, there are two primary categories of uncertainties: 1) aleatoric uncertainty which is also known as statistical uncertainty, due to inherently random effects, and 2) epistemic uncertainty which is due to inadequate knowledge [16]. More specifically, uncertainty associated with the accuracy of the AI model can be categorized into data uncertainty (as aleatoric uncertainty) and model uncertainty (as epistemic uncertainty) [6]. Data uncertainty is caused by the inherent characteristics of the training data, while model uncertainty is caused by several primary factors listed in Table 1. The above-mentioned uncertainties can serve to quantify the level of confidence or trust in the AI model's prediction and are usually measured through probability [6, 17]. From an engineering perspective, uncertainty is related to "*the feasibility of performance and the extent of value to be delivered*" [12]. In this regard, uncertainty may reside in the requirements and implementation of AI-based systems, and may even exhibit in their operational environments [12]. In this work, we focus on the uncertainty in the requirements specification, which is primarily caused by Factors 3 and 4 in Table 1 i.e., mismatches between historical training data and actual run-time data.

Table 1. The primary factors that induce model uncertainty [6]

Primary factors contributing to model uncertainty
Factor 1 – Errors in the training procedure of the model
Factor 2 – Errors in the structure of the model
Factor 3 – Lacking of knowledge because of unknown data
Factor 4 – Lacking of knowledge because of inadequate coverage of the training data set

Handling uncertainties effectively after the model is deployed in its operational environment is particularly imperative for AI-based applications with critical quality requirements, because systems incorporating AI components may exhibit a substantial error margin attributable to the inherent uncertainty [9], which can constitute a catastrophe for these quality-critical systems, such as autonomous navigation systems. To date, numerous researchers have dedicated their efforts to tackling uncertainty in AI models at run time. For example, a collection of studies has focused on addressing the uncertainty in AI from the perspective of algorithms [6, 18], we refer to these methods as *"inside-out"* approaches. In this regard, uncertainty prediction is the primary approach to address uncertainty issues, which mainly encompasses uncertainty estimation and model calibration [18]. Many approaches for uncertainty estimation/quantification have been proposed to imbue AI models with self-awareness regarding their prediction confidence, such as Single Deterministic Methods, Bayesian Methods, and Ensemble Methods [6]. On the other hand, there are several types of uncertainty calibration methods that have been presented as well, such as Post-processing methods, and Regularization

methods. However, the incorporation of these methods into real-world mission/safety-critical applications remains markedly restricted. Challenges include that the uncertainty linked to a single decision is hard to be assessed, the absence of a standardized evaluation protocol, the susceptibility of certain methods to the training dataset, and more [6].

Other related research endeavors address uncertainty from the perspective of software engineering (SE), which we refer as *"outside-in"* approaches, which tackle uncertainty from an engineering standpoint. For example, Ishikawa et al. [12] emphasized that addressing AI model uncertainty during the requirements analysis is pivotal in ensuring the success of AI-based systems. They proposed to tackle uncertainty issues of the AI model with an emphasis on the area of decision-making of design. In [13], Ishikawa et al. proposed some guidelines through GSN to facilitate employing arguments for AI-based systems given the uncertain nature of AI. Additionally, there has been research focused on addressing uncertainty related to training data for AI-based systems. For example, a framework was presented by Dey et al. [14, 15] delineating a process for data requirements engineering and assessing data uncertainty in the context of AI-based safety-critical systems. Another relevant research thread is SE of self-adaptive systems [19], which includes the exploration of methods dealing with uncertainties in problem definition, system design, and run-time execution. However, there is still a huge gap between the above-mentioned methods and the engineering of general AI-based components with runtime critical qualities.

2.2 Quality Assurance of AI-Based Components

The limitations of AI technologies give rise to a common engineering challenge to most AI-based projects, regarding the quality assurance of AI-based components. It has become one of the most difficult engineering problems related to the application of AI technologies, especially, in scenarios where some critical qualities of the systems must be guaranteed, such as safety, and has recently attracted attention in the research communities of AI and software engineering. In terms of safety assurance, traditionally, there are four common types of methods for enhancing engineering safety, including inherently safe design, safety reserves, safe fail, and procedural safeguards [20]. Within the AI context, Dey et al. [5] have surveyed the safety approaches for machine learning-based systems. The survey was conducted from the software engineering perspective and has mapped existing work into the layers of requirements, design, and verification. Dey et al. concluded that among all three layers of work, the requirements layer has received the least attention, which is alarming. It is imperative to clearly define *"what to verify"*, *"against which metrics to verify"*, and *"what are the qualitative and quantitative targets"*, before conducting any subsequent quality assurance activities, in the phases of design, verification & validation, and maintenance. That is, having clear and complete requirements specifications is the prerequisite to properly conducting the quality assurance process.

Kuwajima et al. [21] explored the unresolved engineering challenges associated with the development of safety-critical AI-based systems. The authors argue that the crucial approach to employing an AI model within safety-critical systems requires the partitioning of the training process of AI models into distinct phases, including requirements

specification, design, and verification, and they found that the biggest challenge in measuring the quality of AI-based systems lies in the requirements specification. Meanwhile, Kuwajima et al. pointed out that the quality of the AI-based system holds greater significance than that of the AI model itself, especially once the AI model approaches maximum accuracy, and highlighted that the adoption of supplementary security analysis methods is imperative to mitigate model uncertainties.

Additionally, Siebert et al. [22] highlighted the necessity for guidelines tailored to quality models applicable to AI-based software systems. The authors developed a specific quality model relevant to a particular industrial use case. Their model covers different dimensions of AI-based systems, including model, data, system, infrastructure, and environment. Nakamichi et al. [23] proposed a methodology to ensure the quality of the development process for AI-based software, by extending the traditional quality model ISO/IEC 25010. A recent study conducted by Ali et al. [24] helps gain insights into quality models regarding AI-based systems, software, and components, indicating that there is no work addressing quality models at the AI component level.

2.3 Requirements Specification of AI-Based Components

The inherent uncertainty in AI applications requires new software requirements specification techniques to fully meet the demands imposed by AI-based systems. For instance, among existing RE approaches and techniques, it has been well discussed how to specify testable and measurable requirements for software systems. Roberson et al. have elaborated on the needs and methods to derive and formulate testable requirements, a.k.a., fit criteria, based on the original requirements and their corresponding rationale [25]. However, this type of method cannot be directly applied to specify requirements for AI-based components because of the limitations discussed in Sect. 2.1, due to uncertainties within AI.

Over the past few years, there has been a burgeoning interest in research on requirements specification for AI. Berry et al. [26] discussed the requirements specification for AI-based systems, and emphasized the utilization of measures, criteria to establish acceptable thresholds for these measures, and a contextual comprehension of the AI to assess its performance. Ahmad et al. presented a framework "RE4HCAI", which is designed for the extraction and modeling of requirements for AI systems with a human-centered focus [27]. Their work touched upon the area of requirements for errors and failure, and the authors proposed the inclusion of error sources, error risks, error types, and methods for error mitigation. However, in terms of mitigating errors, the authors only suggested that *"(user shall) provide suggestions"* and *"to allow user to fix"*, lacking specific solutions tailored to address errors specifically attributable to uncertainty. Maalej et al. [28] presented six aspects that demand tailoring within the AI landscape, one of which is regarding specifying quality requirements precisely and quantifiably. The authors advocate for the utilization of acceptable quality levels, such as "$> 90\%$", rather than fixed metrics, to achieve consensus on exact values for quality criteria within responsible AI systems.

In the context of requirements specification for AI components, Rahimi et al. [29] introduced a method aimed at enhancing the specification of unambiguous requirements, such as the definition of "pedestrians", for AI components by developing a web-based

benchmark. Hu et al. [30] proposed a method for specifying and testing the robustness requirements of AI components grounded in human perception. On the other hand, several attempts have been made to analyze and specify data requirements for AI-based systems [14, 15], and to analyze and specify certain NFRs for AI [30].

Another relevant research thread is requirements modeling techniques for AI, and a recent extensive review, encompassing 43 studies published between 2010 and mid-2021, was conducted pertaining to the RE for AI (RE4AI) field by Ahmad et al. [31, 32], and UML, GORE, and Domain Specific Models have been found as the favored approaches for requirements modeling for AI. This study also shows that the primary emphasis within the RE4AI literature pertained to data requirements and explainability, and no research was identified regarding error addressing during the RE phase [31, 32]. However, according to another recent survey involving industry professionals [33], a desire was expressed by the practitioners to understand the methods for handling errors and specifying their sources during the RE phase, and meanwhile, the significance of both identifying errors and establishing their connection to user needs is also underscored in industrial guidelines.

In summary, the existing literature pertaining to the requirements specification for AI-based systems has endeavored to tackle diverse issues from a multitude of perspectives. However, these proposed methods consistently suffer from the following limitations: 1) They are "*inside-out*" methods and do not address uncertainties at the runtime, but only focus on certain aspects of AI models, e.g., training data. 2) They are too high level and not elaborated enough to provide pragmatic support or guidance toward engineering practice. Moreover, recent works in RE focused on specifying requirements for AI models or processes, instead of treating AI-based components as black-boxes at first. Such a tendency might distract the focus of a specification from "what" to "how" and might eliminate the opportunities for creativity by making upfront design decisions. Overall speaking, there is little systematic work addressing uncertainties in the specification of the AI-based components to ensure runtime critical qualities.

2.4 Obstacle Analysis

Risk analysis constitutes a vital component of the RE phase, which is employed to recognize scenarios capable of inducing failures of the systems [34, 35]. Within the context of goal-oriented RE, "*obstacle analysis is a goal-oriented form of risk analysis*" [35], which can be subdivided into three steps: 1) Identifying as many obstacles as possible; 2) Assessing obstacles' likelihood and consequences severity; 3) Identifying solutions to the obstacles and incorporating them as new goals [35]. KAOS-based [11] obstacle analysis framework, as a mainstream technique for obstacle analysis, has been employed in various application scenarios. For example, obstacle analysis within KAOS has been used to identify contingency requirements for an unpiloted aerial vehicle [36], and a KAOS-based Goal-Obstacle analysis has been conducted to facilitate the specification of system scalability requirements [37]. Overall, the elimination of obstacles has been validated as a necessary pre-condition for the achievement of system goals.

In the engineering process of AI-based systems, uncertainties in data and AI models can be viewed as obstacles for the AI model to make the right prediction. In this

regard, these uncertainties can be treated as obstacles for the AI-based system to fulfill its goals. Hence, the rationale and methodology of obstacle analysis [35] can be referred to when dealing with errors caused by uncertainties in AI-based components. For example, DiMatteo et al. analyzed the consequence severity of false positive and false negative errors in the context of monitoring the inattention of the responsible human in an autonomous vehicle [38]. Provost et al. emphasized the cost of false positive and false negative errors is unlikely to be equal, and presented an Expected Value framework that can be used to calculate the impact of the above two types of errors, in which the calculation of the probabilities and costs of the errors are involved [39].

3 Making Uncertainty Certain by NFR

3.1 The Context

The subsequent discussion is expanded upon the specification of AI-based components, rather than the AI model itself, i.e., the AI model is treated as a black-box. We will focus on the satisfaction of critical qualities at runtime, rather than over a period of time, because the current research in self-learning/adaptation of AI is making great progress in dealing with the latter case. To simplify the discussion, we will focus on the scenarios in which the AI-based component functions as a classifier, i.e., the output from the AI-based component would be a finite set. This assumption, however, does not restrict the type of AI technologies under consideration. For example, deep learning models can be used for classification, whose use case is included in our discussion.

3.2 The Rationale

$\mathcal{E}$ Describes the properties of the environment, $\mathcal{S}$ is *"an optative description of a condition over the shared phenomena at the interface between the machine and the environment"* [40]. $\mathcal{R}$ describes the properties that the environment will exhibit after a machine is installed in the environment whose behavior satisfies $\mathcal{S}$, as shown in Fig. 1 (a). That is, $\mathcal{E}, \mathcal{S} \vdash \mathcal{R}$. However, due to the uncertainties that reside in the machine, i.e., the AI-based component, $\mathcal{S}$ cannot be fully specified. Hence, $\mathcal{R}$ inherits the genetic uncertainties associated with $\mathcal{S}$, making it difficult to use as a reference for quality assurance purposes.

This issue asks for serious consideration for AI-based components because the users must rely on the accuracy of the components to achieve their goals. Naturally, a specification is created centering around the accuracy of the AI-based components, which eventually is drilled down to the accuracy of the AI model. We refer to these types of specification methods as "*accuracy-centric"* ones. However, in the application context, it is uncertain if the accuracy of a pre-trained AI model will remain true. In addition, at runtime, it is not realistic to validate the correctness or accuracy of the AI model. (Note that a probability of a prediction is still yet to be validated.) Hence, a specification based on the accuracy of an AI model (Fig. 1. (b)) is not able to deliver a complete assurance on critical qualities. For example, Fig. 1 (c) shows a specification of Alipay's Facial Recognition Component (FRC), which is modeled with KAOS, **i***, and Problem

Frames (PFs) using RE-Tools [41]. The users are granted the ability to complete their transactions through facial recognition within FRC. This is a security-critical scenario, because an error produced by the AI-based (a.k.a., facial recognition) component may give rise to undesirable consequences, i.e., unauthorized charges. According to [42], their facial recognition component boasts an accuracy rate of 99.99% which is acquired from statistics of historical data and training results, and its implication is that for future unknown application scenarios, especially for those that have not been seen before, there is little guarantee that the above-mentioned accuracy rate would always hold, same for the specification built upon that. Hence, when it comes to building AI into applications, we shall recognize the possibility that the AI models, consequently the AI-based components, may fail and produce False Positive and False Negative results [27], and the fact that it remains unknown when, where, and how it would fail.

In quality-non-critical application scenarios, the above-discussed issue may not be a huge concern. For example, most users can tolerate some errors when using a voice recognition feature of a virtual assistant, such as Siri. When Siri fails to correctly recognize users' voice input, users may choose to switch to other input methods, such as the keyboard. However, in other application scenarios, some critical quality of AI-based components must be guaranteed. For example, the obstacle detection component of an Autonomous Driving system must function properly to guarantee the safety of the driver, passengers, the vehicle itself, and others on the road. In this case, other safety measures must be established to work with the obstacle detection component together to achieve safety at the system level.

As discussed above, because of uncertainties, an "*inside-out*" approach that produces an accuracy-centric specification, cannot assure the fulfillment of critical qualities in AI-based components. Intuitively to try with the other way around, an "*outside-in*" approach may be worthy of some exploration, that is to use various critical qualities to "contain" the ambiguous behavioral boundary of AI-based component. In Siri's case, other non-AI-based input methods are used as an alternative mechanism, to ensure the accessibility of other subsequent virtual assistance functions (shown in Fig. 2 (a)). We could use $\mathcal{A}$ to represent the properties of the alternative mechanism, and because it is a regular, non-AI-based component, $\mathcal{A}$ can be clearly defined using the traditional requirements specification techniques such as PFs [40], **i*** [43], KAOS [11], NFR framework [44], UML [45], without any ambiguity. Hence, when applying $\mathcal{A}$ on top of the specification of the AI-based component $\mathcal{S}$, that is $\mathcal{E}, \mathcal{S}'' \vdash \mathcal{R}$, where $\mathcal{S}'' = \mathcal{S} \bigcap \mathcal{A}$, it can help assure the accessibility of the system. Similarly, in the case of Autonomous Driving, some auxiliary safe measures shall be adopted, to establish a "Safe Zone $\mathcal{Q}$", within the native domain of the AI-based component $\mathcal{S}$, as shown in Fig. 2 (b). When $\mathcal{Q}$ is properly specified without ambiguity, its application on top of $\mathcal{S}$ can help assure the safety of the system, that is $\mathcal{E}, \mathcal{S}''' \vdash \mathcal{R}$, where $\mathcal{S}''' = \mathcal{S} \bigcap \mathcal{Q}$.

To generalize, for an AI-based system that comes with some critical NFRs, these NFRs can be used as constraints on top of the domain of the AI-based component. As shown in Fig. 2 (c), $\mathcal{E}, \mathcal{S}^* \vdash \mathcal{R}$, where $\mathcal{S}^* = \mathcal{S} \bigcap \mathcal{C}_1 \bigcap \mathcal{C}_2 \bigcap \ldots \bigcap \mathcal{C}_n$. Since each $\mathcal{C}_i$ is specified as a precise constraint to satisfy a critical NFR, $\mathcal{S}^*$ will consequently satisfy all required critical NFRs. In this manner, the uncertainties with the AI-based component are mitigated and all of the system-level critical NFRs are assured.

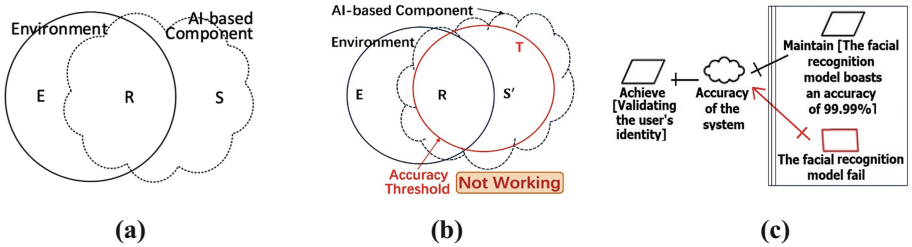

Fig. 1. (a) $\mathcal{E}, S_{ai} \vdash R_{ai}$ (b) $\mathcal{E}, S'_{c=T} \vdash R_{ai}$ (c) Rationalization of goals for AI components

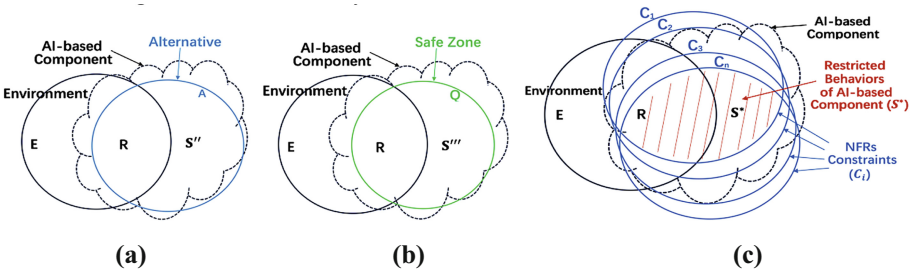

Fig. 2. (a) $\mathcal{E}, S'' \vdash R, S'' = S \bigcap A$ (b) $\mathcal{E}, S''' \vdash R, S''' = S \bigcap Q$ (c) $\mathcal{E}, S^* \vdash R, S^* = S \bigcap C_1 \bigcap C_2 \bigcap \ldots \bigcap C_n$

3.3 An NFRs-Based Approach

So far, we have acknowledged the following facts and observations:

1) The AI-based component that aims at solving classification tasks will surely produce four types of results, including True Positive, True Negative, False Positive (Type I Error), and False Negative (Type II Error) [27].
2) When enforcing critical NFR constraints on top of an AI-based component, the domain properties of the component will surely satisfy these critical NFRs.

Based on the above and the rationale of obstacle analysis [35], an NFRs-based approach to mitigating the impact of uncertainties within the specification of AI-based components is proposed as follows (Fig. 3):

Step 1: Based on the domain context and the acquired user requirements, identify as many system NFRs (both critical and non-critical ones) to be achieved for the AI-based component as possible. Analyze the influence of false decisions (Type–I - False Accept and Type –II - False Reject) of the AI model on these NFRs. Then analyze stakeholders' different tolerance for Type I and Type II errors, which becomes to rationale for a trade-off between conflicting NFRs. Contribution relationships from **i*** framework can be used to represent different influence types [43].

Step 2: Based on the domain knowledge, decompose each affected NFR to system-level constraints $(\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_i)$ that will be further specified formally and built into the AI-based component, by using modeling techniques such as KAOS, **i***, PFs. We recommend using Robertsons' fit criteria as the form of these constraints [25].

Step 3: Correlate each decomposed quality constraint with problem domains and data streams within the problem frame model of the AI-based component. The "Topic Reference" dash line is used to indicate the link between the system requirements and the affected NFRs, and the "Data Association" dash line is used to show the association between the decomposed constraints from Step 2 and the corresponding data stream parameters within the problem frame.

Step 4: Formally define each decomposed constraint based on the corresponding data stream parameters in the problem frame, denoted as $C_1', C_2', \ldots, C_i'$ in Fig. 3. A "Dependency" line is used to show this dependency relationship. We recommend using KAOS to formally define the constraints.

Step 5: Formulate the specification of the AI-based component by enforcing all formal constraints with the trade-off rationale identified in Step 1. Because in this case, the system behavior space becomes $\mathcal{S}^* = \mathcal{S} \bigcap C_1' \bigcap C_2' \bigcap \ldots \bigcap C_n'$, as discussed in Sect. 3.2, the uncertainties with the AI-based component are mitigated and all involved critical NFRs are guaranteed as well.

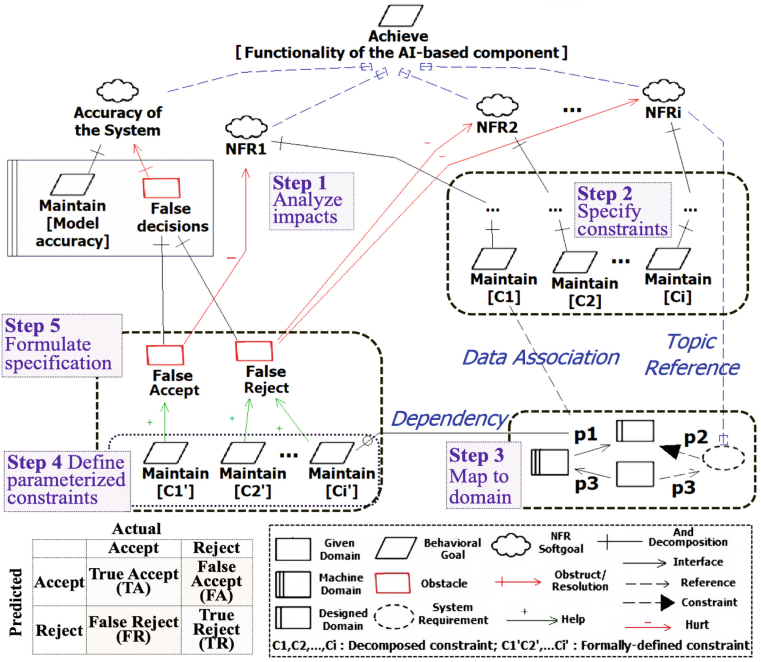

Fig. 3. An NFRs-based approach to mitigating the impact of uncertainties within the specification of AI-based components.

4 A Running Example for the Approach

In this section, the aforementioned Facial Recognition Component (FRC) is used as an example to illustrate our proposed approach. Figure 4 shows a partial use case of a Mobile Banking App (MBA), in which the "*Facial Recognition*" use case will be implemented as an AI-based component.

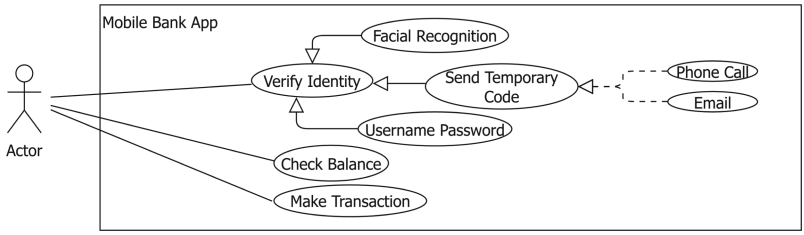

Fig. 4. The domain context of a Mobile Banking App.

Some example requirements of the FRC are given as follows:

"The FRC shall have good performance and the facial recognition process should be expeditious."

"The FRC shall allow users to freely switch between the following methods to verify their identities: 1) Automated verification with facial recognition through facing scanning; 2) Manual verification with their password; 3) Secondary verification using a temporary code sent via pre-registered phone or email."

"The FRC must be accurate and secure, and shall guarantee that the transaction is executed by the user in person."

Our proposed approach is executed step-by-step as follows:

Step 1: By studying the domain context, related NFRs are identified as follows:

(1) **NFR1**: Performance – the facial recognition process shall be expeditious.
(2) **NFR2**: Accessibility – the system shall be accessible to users.
(3) **NFR3**: Security – the system shall be secure.
(4) **NFR4**: Accuracy – the system shall exhibit a high degree of accuracy.

As shown in Fig. 5, NFR4 Accuracy is directly fulfilled by the facial recognition AI model at the operational level. As discussed in Sect. 3.2, we would anticipate Type I False Accept and Type II False Reject errors from the AI model. In the use case of completing a transaction, a Type I error will hurt NFR3 Security, and a Type II error will hurt NFR1 Performance and NFR2 Accessibility. In the business context of the FRC, False Accept poses significant risk and is not acceptable. Comparatively, the adverse consequences of False Reject are relatively moderate and can be tolerated to some extent. Hence, the satisfaction of NFR3 Security shall be prioritized over that of NFR1 Performance and NFR2 Accessibility. Similarly, because the impact of Type II False Reject errors on NFR2 Accessibility (*Hurt*) is more severe than on NFR1 Performance (*Some-*), the satisfaction of NFR2 Accessibility shall be prioritized over that of NFR1 Performance.

Step 2: NFR 1, 2, and 3 are written into fit criteria as follows:

(1) $C1$: Performance – The facial recognition process shall be completed within 2 s.
(2) $C2$: Accessibility – When facial recognition fails or is timed out, the system shall prompt the user with an option to switch to a secondary verification method. Available options include using a password, or a temporary verification code sent via a pre-registered phone number or email.
(3) $C3.1$: Security – At the initial setup of the facial recognition feature or when making large transactions or conducting transactions with high frequency, a secondary verification is required.
(4) $C3.2$: Security – If the user exhibits abnormal behaviors while making a transaction, a secondary verification is required. For a failed secondary verification, the account shall be temporarily protected until further verification is successfully completed.

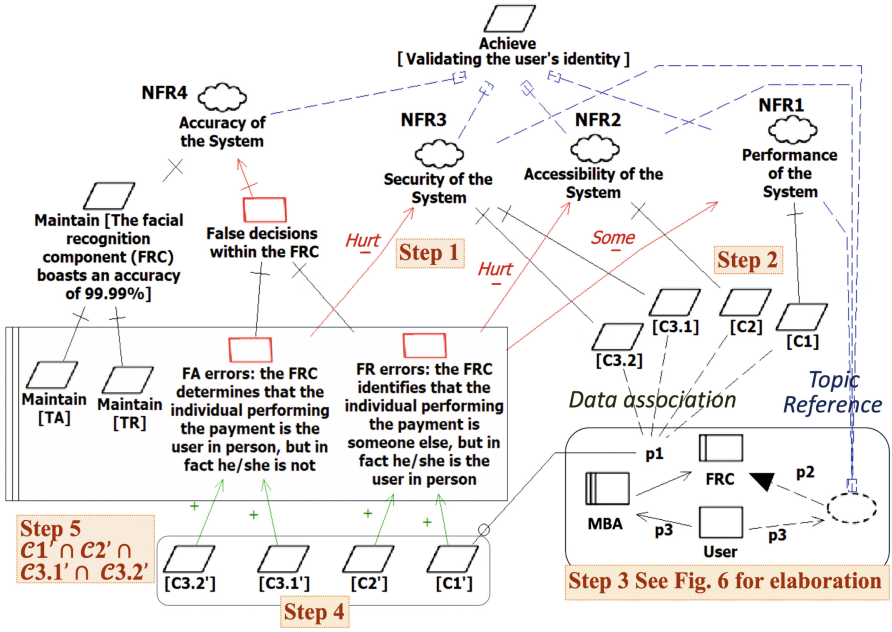

Fig. 5. An illustration of the application of our approach on FRC.

Step 3: Each decomposed quality constraint is associated with corresponding the data streams in the problem frame of the AI-based component (Fig. 6). Based on such referencing and data association, business-level fit criteria can be contextualized into operational-level constraints in Step 4.

Step 4: Formally define each decomposed constraint (***C1***, ***C2***, ***C3.1***, ***C3.2***) based on the data stream parameters, specifically, p1, p4, p7, and p10, in the problem frame. Some parameters and domains are defined as follows: *s: FacialRecognitionSystem, u: User, ta: TransactionAmount, tf: TransactionFrequency, duo: SecondaryVerification, vc: VerificationCode, pn: PhoneNumber, e: Email, pw: Password, td: TimeofDay, tz: TimeZone, ip: IP*.

(1) $\boldsymbol{C}1^{'}$: Performance – $\forall$ *s, u: Recognition(s, u)* $\wedge$ *s.response_time* $\leq$ *2 s* $\Rightarrow$ *timeOut* $=$ *False*
(2) $\boldsymbol{C}2^{'}$: Accessibility – $\forall$ *s, vc, e, pn, pw, u: Recognition(s, u)* $\wedge$(*s.timeout* $=$ *True* $\vee$ *s.state* $=$ *"Reject")* $\Rightarrow$ *TriggerAlternative(s, vc, e, pn, pw)*
(3) $\boldsymbol{C}3.1^{'}$: Security – $\forall$ *s, ta, tf, duo, vc, pn, e, pw, u: u.type* $=$ *"new"* $\vee$ *ta* $\geq$ *u.transactionAverage()* $\vee$ *tf* $\geq$ *u.frequencyAverage()* $\Rightarrow$ *Require(s, duo, vc, pn)* $\vee$ *Require(s, duo, vc, e)* $\vee$ *Require(s, duo, pw)*
(4) $\boldsymbol{C}3.2.1^{'}$: Security – $\forall$ *s, duo, td, tz, ip, vc, e, pn, pw, u: u.type* $=$ *"existing"* $\wedge$ *(td* $\notin$ *u.activityPattern()* $\vee$ *u.mostFrequent (tz)* $=$ *False* $\vee$ *ip* $\notin$ *u.topIP(2))* $\Rightarrow$ *Require(s, duo, vc, pn)* $\vee$ *Require(s, duo, vc, e)* $\vee$ *Require(s, duo, pw)*
(5) $\boldsymbol{C}3.2.2^{'}$: Security – $\forall$ *s, duo, u: u.type* $=$ *"existing"* $\wedge$ *duo.state* $=$ *"Failed"* $\Rightarrow$ *s.state* $=$ *"Reject"* $\wedge$ *u.type* $=$ *"suspended_temp"*

Step 5: Enforcing all the formally defined constraints by applying $\mathcal{S}^{*} = \mathcal{S} \bigcap \boldsymbol{C}1^{'} \bigcap \boldsymbol{C}2^{'} \bigcap \boldsymbol{C}3.1^{'} \bigcap \boldsymbol{C}3.2^{'}$. According to Step 1, NFR Security has higher priority over NFR Performance and Accessibility, that is $\boldsymbol{C}3.1^{'} \bigcap \boldsymbol{C}3.2^{'}$ shall be prioritized over $\boldsymbol{C}1^{'}$ and $\boldsymbol{C}2^{'}$, and $\boldsymbol{C}2^{'}$ shall be prioritized over $\boldsymbol{C}1^{'}$.

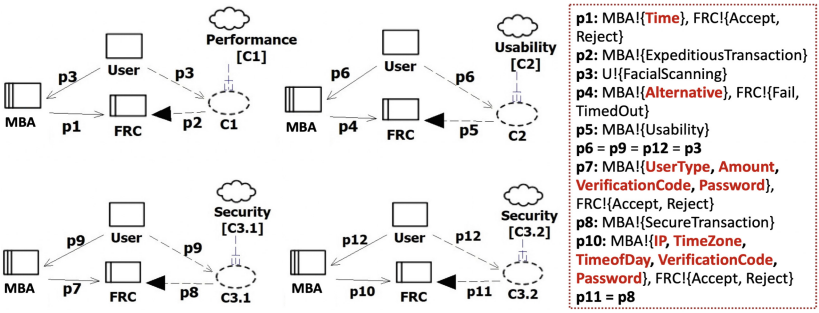

Fig. 6. Correlating problem domains, data streams (p*i*), with each quality constraint C*i*.

Hence, the NFRs of the mobile banking application shall be specified as follows:

(1) **Security-1**: [*Priority-high*], $\forall$ *s, duo, ta, tf, td, tz, ip, vc, e, pn, pw, u: u.type* $=$ *"new"* $\vee$ *(u.type* $=$ *"existing"* $\wedge$ *(ta* $\geq$ *u.transactionAverage()* $\vee$ *tf* $\geq$ *u.frequencyAverage()*$\vee$ *td* $\notin$ *u.activityPattern()* $\vee$ *u.mostFrequent (tz)* $=$ *False* $\vee$ *ip* $\notin$ *u.topIP(2)))* $\Rightarrow$ *Require(s, duo, vc, pn)* $\vee$ *Require(s, duo, vc, e)* $\vee$ *Require(s, duo, pw)*

Security-2: [*Priority-high*] – $\forall$ *s, duo, u: u.type* $=$ *"existing"* $\wedge$ *duo.state* $=$ *"Failed"* $\Rightarrow$ *s.state* $=$ *"Reject"* $\wedge$ *u.type* $=$ *"suspended_temp"*.
(2) **Accessibility**: [*Priority-medium*], $\forall$ *s, vc, e, pn, pw, u: Recognition(s, u)* $\wedge$(*s.timeout* $=$ *True* $\vee$ *s.state* $=$ *"Reject")* $\Rightarrow$ *TriggerAlternative(s, vc, e, pn, pw)*
(3) **Performance**: [*Priority-low*], $\forall$ *s, u: Recognition(s, u)* $\wedge$ *s.response_time* $\leq$ *2 s* $\Rightarrow$ *s.timeOut* $=$ *False*

5 Conclusion and Discussion

In this paper, an NFRs-based approach to mitigating the impacts of uncertainty in AI-based components in quality-critical scenarios has been proposed. We also illustrated how to use our proposed approach to specify requirements for a real-world AI-based component – a Facial Recognition Component, with a combination of various RE techniques, including KAOS, **i***, and PFs. To the best of our knowledge, we are among the first to link Type I and Type II errors of AI-based components for NFRs analysis and prioritization, and to address uncertainties at the runtime for quality assurance of the AI-based system.

Our proposed approach is geared towards AI-based components used as classifiers, which stands out as one of the foremost AI tasks that have attracted considerable attention from participation in the industry [33]. However, it could also be applicable to the AI-based components that are intended to address non-classification tasks, such as regression problems, as long as the quality quantification of the regression output is specified with discrete values, e.g., a numeric threshold. Hence, it would be interesting to explore and see if our approach could be applied in scenarios of a similar nature.

The limitation of our approach is that it can assure the runtime satisfaction of critical qualities of the AI-based component, while AI models remain as black-boxes. By applying our approach, the uncertainties can be contained, but not eliminated completely, which means AI's behaviors are still uncertain, but at least all the critical qualities are assured. Although the requirements are inherently incomplete [46], in order to achieve sufficient completeness of the requirements while accepting the existence of uncertainty, studying how to maximize the power of AI with all critical qualities assured at the same time will be one of our future research endeavors.

Acknowledgments. This study was supported by the Guangdong Provincial Key Laboratory of Interdisciplinary Research and Application for Data Science, BNU-HKBU United International College (project code 2022B1212010006) and the National Key R&D Program of China, Tsinghua University (project code 2021YFC2701702).

References

1. AITopics. https://aitopics.org/. Accessed 23 Nov 2023
2. Association for uncertainty in Artificial Intelligence. https://www.auai.org/. Accessed 23 Nov 2023
3. Yampolskiy RV.: Unpredictability of AI. arXiv preprint arXiv:1905.13053v1 (2019)
4. Yampolskiy, R.V.: Unexplainability and incomprehensibility of artificial intelligence. arXiv preprint arXiv:1907.03869v1 (2019)
5. Dey, S., Lee, S.W.: Multilayered review of safety approaches for machine learning-based systems in the days of AI. J. Syst. Softw. **176**, 110941 (2021)
6. Gawlikowski, J., Tassi, C.R.N., Ali, M., Lee, J., Humt, M., Feng, J., et al.: A survey of uncertainty in deep neural networks. Artif. Intell. Rev. **56**(Suppl 1), 1513–1589 (2023)
7. Yang, J., Liu, L.: What users think about predictive analytics? A survey on NFRs. In: 2020 IEEE 28th International Requirements Engineering Conference (RE), pp. 340–345 (2020)

8. NHTSA Office of Defects Investigation #PE 16–007, https://static.nhtsa.gov/odi/inv/2016/INCLA-PE16007-7876.PDF. Accessed 23 Nov 2023
9. Ozkaya, I.: What is really different in engineering AI-enabled systems? IEEE Softw. **37**(4), 3–6 (2020)
10. Ishikawa, F., Yoshioka, N.: How do engineers perceive difficulties in engineering of machine-learning systems? - questionnaire survey. In: 2019 IEEE/ACM Joint 7th International Workshop on Conducting Empirical Studies in Industry (CESI) and 6th International Workshop on Software Engineering Research and Industrial Practice (SER&IP), pp. 2–9. IEEE, Canada (2019)
11. Van Lamsweerde, A.: Goal-oriented requirements engineering: a guided tour. In: Proceedings Fifth IEEE International Symposium on Requirements Engineering, pp. 249–262. IEEE, Toronto, Canada (2001)
12. Ishikawa, F., Matsuno, Y.: Evidence-driven requirements engineering for uncertainty of machine learning-based systems. In 2020 IEEE 28th International Requirements Engineering Conference (RE), pp. 346–351. IEEE, Zurich (2020)
13. Ishikawa, F., Matsuno, Y.: Continuous argument engineering: tackling uncertainty in machine learning based systems. In: Gallina, B., Skavhaug, A., Schoitsch, E., Bitsch, F. (eds.) Computer Safety, Reliability, and Security. SAFECOMP 2018. LNCS, vol. 11094, pp. 14–21. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99229-7_2
14. Dey, S.: Evidence-driven data requirements engineering and data uncertainty assessment of machine learning-based safety-critical systems. In: 2022 IEEE 30th International Requirements Engineering Conference (RE), pp. 219–224. IEEE, Melbourne (2022)
15. Dey, S., Lee, S. W.: A Multi-layered collaborative framework for evidence-driven data requirements engineering for machine learning-based safety-critical systems. In: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing (SAC 2023), pp. 1404–1413. ACM, New York (2023). https://doi.org/10.1145/3555776.3577647
16. Hüllermeier, E., Waegeman, W.: Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. Mach. Learn. **110**(3), 457–506 (2021)
17. Hong, Y., et al.: Statistical perspectives on reliability of artificial intelligence systems. Qual. Eng. **35**(1), 56–78 (2023)
18. Mohseni, S., Wang, H., Xiao, C., Yu, Z., Wang, Z., Yadawa, J.: Taxonomy of machine learning safety: a survey and primer. ACM Comput. Surv. **55**(8), 157:1–157:38 (2022)
19. Weyns, D.: Software engineering of self-adaptive systems. In: Cha, S., Taylor, R., Kang, K. (eds.) Handbook of Software Engineering, pp. 399–443. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-00262-6_11
20. Möller, N., Hansson, S.O.: Principles of engineering safety: risk and uncertainty reduction. Reliab. Eng. Syst. Saf. **93**(6), 798–805 (2008)
21. Kuwajima, H., Yasuoka, H., Nakae, T.: Engineering problems in machine learning systems. Mach. Learn. **109**(5), 1103–1126 (2020). https://doi.org/10.1007/s10994-020-05872-w
22. Siebert, J., et al.: Towards guidelines for assessing qualities of machine learning systems. In: Shepperd, M., Brito e Abreu, F., Rodrigues da Silva, A., Pérez-Castillo, R. (eds.) Quality of Information and Communications Technology. QUATIC 2020. Communications in Computer and Information Science, vol. 1266, pp. 17–31. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58793-2_2
23. Nakamichi, K., et al.: Requirements-driven method to determine quality characteristics and measurements for machine learning software and its evaluation. In: 2020 IEEE 28th International Requirements Engineering Conference (RE), pp. 260–270, IEEE, Switzerland (2020)
24. Ali, M.A., Yap, N.K., Ghani, A.A.A., Zulzalil, H., Admodisastro, N.I., Najafabadi, A.A.: A systematic mapping of quality models for AI systems, software and components. Appl. Sci. **12**(17), 8700 (2022)

25. Robertson, S., Robertson, J.: Mastering the Requirements Process: Getting Requirements Right. 3rd eds. Addison-Wesley Professional (2012)
26. Berry, D.M.: Requirements engineering for artificial intelligence: what is a requirements specification for an artificial intelligence?. In: Gervasi, V., Vogelsang, A. (eds.) Requirements Engineering: Foundation for Software Quality. REFSQ 2022. LNCS, vol. 13216, pp. 19–25. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-98464-9_2
27. Ahmad, K., Abdelrazek, M., Arora, C., Baniya, A.A., Bano, M., Grundy, J.: Requirements engineering framework for human-centered artificial intelligence software systems. Appl. Soft Comput. **143**(C), 110455 (2023)
28. Maalej, W., Pham, Y.D., Chazette, L.: Tailoring requirements engineering for responsible AI. Computer **56**(4), 18–27 (2023). https://doi.org/10.1109/MC.2023.3243182
29. Rahimi, M., Guo, J. L., Kokaly, S., Chechik, M.: Toward requirements specification for machine-learned components. In 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW), pp. 241–244. IEEE, Korea (2019)
30. Hu, B.C., Salay, R., Czarnecki, K., Rahimi, M., Selim, G., Chechik, M.: Towards requirements specification for machine-learned perception based on human performance. In: 2020 IEEE Seventh International Workshop on Artificial Intelligence for Requirements Engineering (AIRE), pp. 48–51. IEEE, Zurich, Switzerland (2020)
31. Ahmad, K., Bano, M., Abdelrazek, M., Arora, C., Grundy, J.: What's up with requirements engineering for artificial intelligence systems?. In 2021 IEEE 29th International Requirements Engineering Conference (RE), pp. 1–12. IEEE, USA (2021)
32. Ahmad, K., Abdelrazek, M., Arora, C., Bano, M., Grundy, J.: Requirements engineering for artificial intelligence systems: a systematic mapping study. Inf. Softw. Technol. **158**, 107176 (2023)
33. Ahmad, K., Abdelrazek, M., Arora, C., Bano, M., Grundy, J.: Requirements practices and gaps when engineering human-centered artificial intelligence systems. Appl. Soft Comput. **143**, 110421 (2023)
34. Aydemir, F. B., Giorgini, P., Mylopoulos, J.: Multi-objective risk analysis with goal models. In: 2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS), pp. 1–10. IEEE, France (2016)
35. Cailliau, A., Van Lamsweerde, A.: Handling knowledge uncertainty in risk-based requirements engineering. In: 2015 IEEE 23rd International Requirements Engineering Conference (RE), pp. 106–115. IEEE, Canada (2015)
36. Lutz, R., Patterson-Hine, A., Nelson, S., Frost, C.R., Tal, D., Harris, R.: Using obstacle analysis to identify contingency requirements on an unpiloted aerial vehicle. Requirements Eng. **12**, 41–54 (2007)
37. Duboc, L., Letier, E., Rosenblum, D.S.: Systematic elaboration of scalability requirements through goal-obstacle analysis. IEEE Trans. Software Eng. **39**(1), 119–140 (2012)
38. DiMatteo, J., Berry, D.M., Czarnecki, K.: Requirements for monitoring inattention of the responsible human in an autonomous vehicle: the recall and precision tradeoff. In: REFSQ Workshops (2020)
39. Provost, F., Fawcett, T.: Data science for Business: What you need to know about Data Mining and Data-Analytic Thinking. Inc, O'Reilly Media (2013)
40. Jackson, M.: Problem Frames: Analyzing and Structuring Software Development Problems. Addison-Wesley, USA (2000)
41. Supakkul, S., Chung, L.: The RE-Tools: a multi-notational requirements modeling toolkit. In: 2012 20th IEEE International Requirements Engineering Conference (RE), pp. 333–334. IEEE, Chicago, IL, USA (2012)
42. Alipay documentation center. https://opendocs.alipay.com/open/20180402104715814204/intro. Accessed 6 Dec 2023

43. Yu, E.S.: Towards modelling and reasoning support for early-phase requirements engineering. In: Proceedings of ISRE'97: 3rd IEEE International Symposium on Requirements Engineering, pp. 226–235. IEEE, USA (1997)
44. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Springer, New York (2012)
45. Rumbaugh, J., Jacobson, I., Booch, G.: The Unified Modeling Language Reference Manual. Addison-Wesley, UK (1998)
46. Arango, G., Freeman, P.: Application of artificial intelligence. ACM SIGSOFT Soft. Eng. Notes **13**(1), 32–38 (1988)