

RESEARCH ARTICLE

An effective approach for eliciting requirements and ensuring traceability: A case study of automated teller machine

Habiba Azrin Mim*

Department of Computer Science and Engineering, East West University, Dhaka 1212, Bangladesh

Abstract – Requirements elicitation plays a great role in the successful completion of a software project. Elicitation is a productive effort to extract project-associated information from the stakeholders. To gather complete, concise, and clear requirements, the concept of requirements elicitation allows for various analyses and techniques. Requirement elicitation is significant due to the lack of efficiency in accurately articulating their demands for the intended system. Therefore, requirements engineers realize the need to perform elicitation to ensure that the requirements produced are easily accessible, usable, and applicable according to the client's demands. A software project can be considered successful if it satisfies all the client's requirements. In certain situations, changes in requirements from the client side may arise during the development of the intended software. This research focuses on the challenges of this context, mentioned as follows: many stakeholders are unable to clearly express their needs for the intended system, and the client-side requirements may change often. In this case, traceability plays a vital role in making the necessary changes in the components of the corresponding requirements. Therefore, the impact of any change can be easily visible & the changes can be applied correspondingly. After all, a project's success can be achieved by ensuring traceability among the components used in eliciting requirements. So, this research proposes an effective approach for eliciting functional and non-functional requirements and ensuring traceability among the corresponding components used in eliciting requirements. In addition, a tool has been designed to ensure traceability among functional requirements, non-functional requirements, and other corresponding components. If any change request arises for any of the requirements, all the corresponding components associated with that requirement can be traced with the help of this tool. Overall, this proposed approach has been applied successfully to a case study of an Automated Teller Machine using this tool.

Article History

Received : 28 July 2024

Revised : 6 March 2025

Accepted : 25 March 2025

Published : 28 March 2025

Keywords

Requirement elicitation

Requirements

Techniques

Traceability

Functional

Non-functional

1. Introduction

Requirements provided by the clients can be considered as the primary pillars for building the software. Software should be tailor-made to fit the client's requirements. In software engineering, users' expectations of a software product can be described as requirements. Requirements engineering includes the collection, analysis, and documentation of software requirements from a client. The features and functionalities of the target software are called the software requirements. Requirements can be categorised into two groups: functional and non-functional. The goal of requirements engineering is to interpret clients' exact requirements so that the intended software product will receive appreciation and satisfaction from its clients. Elicitation can be regarded as the process of collecting information from the end-user stakeholder/client [1]. Successful software cannot be built without proper requirements elicitation. Requirement elicitation is the first step in the software development life cycle. It is also known as the requirement gathering or collection phase. The software can be considered successful if it meets all its client's expectations.

Requirement engineering (RE) contributes significantly to developing the exact software according to the stakeholder's needs [2]. Proper elicitation of functional requirements (FRs) ensures stakeholder commitment and reduces the risk of failure at an early stage. Similarly, proper elicitation of Non-Functional Requirements (NFRs) increases stakeholder satisfaction and future scope for the industry [2]. Usually, the main priority is given to the system features during the software development process, which are documented as functional requirements [3]. To be specific, the FRs can be regarded as the behaviours or actions that the system supports [4]. To develop new software, elicitation of functional requirements specifies the client's needs, structures communication among stakeholders, and provides substantial evidence that controls project progress in software development-related fields [5]-[10]. Modelling of functional requirements can be regarded as a medium between project stakeholders to expedite communication. Thus, it is inevitable that stakeholders and end-users articulate requirements unequivocally, and that stakeholders, such as requirements engineers, interpret them correctly for the Developer and QA teams of the corresponding project [11]. On the contrary, non-functional requirements relate to system functionality and do not directly affect the system's features [12]. Functional requirements may vary due to the diverse functionalities of different software products. The representation of the NFRs can be done in different ways, depending on the justifications for their use and the project's level [13].

Requirements are extracted using various existing requirement elicitation techniques. There are four categories of requirement elicitation techniques according to their nature of communication: (i) Traditional Techniques, (ii) Collaborative Techniques, (iii) Cognitive Techniques, and (iv) Observational Techniques. The traditional techniques are

as follows: Interview, Questionnaire, Data gathered from existing systems, and survey. The collaborative techniques are as follows: Focus Group, Brainstorming, JAD, Prototyping, Workshop, Storyboarding, Models, Use-case/Scenarios, and Viewpoint. The cognitive techniques are as follows: Document Analysis, Card Sorting, Protocol Analysis, Laddering, and Repository Grid. The observational techniques are as follows: Observation and Ethnography/Social Analysis. The software project team uses any of these techniques as needed and according to their expertise to elicit requirements.

Software requirements vary for various reasons; frequent changes are inevitable [14]. In the case of necessary changes to any requirement, it is essential to ensure traceability so that its associations are updated along with that requirement. Traceability of requirements provides the commitment to ensure the synchronisation between the client's requirements and system evolution [15]. Software products should be developed according to the client's requirements to achieve high quality and client satisfaction [16]. Association and communication among stakeholders are very significant for achieving success in eliciting software requirements and ensuring traceability [17]. From the above discussion, it is observed that the proper selection and implementation of requirements elicitation techniques, along with ensuring traceability among the corresponding components used in requirements elicitation, can be a significant approach for conducting this research. In addition, a tool has been developed to perform a case study on an automated teller machine (ATM) to verify the validity of this proposal.

2. Related Work

2.1 Functional Requirement

Functional requirements are the basic facilities and services the client needs the system to provide. All the essential functionalities expected to be incorporated into the intended software as a part of the contract can be known as functional requirements.

2.2 Viewpoints

The Viewpoints (VPs) method is regarded as a technique for eliciting requirements. Viewpoints can be taken based on the types of users using the system. Under the viewpoints, the specific system requirements will be described correspondingly. Viewpoints are introduced at the very beginning of the requirements engineering process to structure the mechanism for eliciting requirements [18]. Identification of viewpoints and organisation of system requirements under them at this earliest stage eliminates the possibility of missing requirements and provides a mechanism to ensure traceability by linking requirements to their corresponding viewpoints [18]. Here, viewpoints are the end-users of the system who perform various activities within it that can be regarded as system requirements and functional requirements.

2.3 Operation Criteria

Operation Criteria (OC) are the prerequisites that must be met to perform any operation in the system. These can also be considered terms and conditions or rules that are mandatory for performing any operation in the system. Some important factors need to be kept in mind while defining OC for any system:

- The environment where the system will be used.
- The type of data on which the system will work to perform any kind of operation.
- The type of audience/end-users who will use the system.

2.4 Non-functional Requirements

Non-functional requirements are the quality factors that the system must satisfy per the project contract. These factors should be prioritised according to the nature and needs of the project. Different non-functional requirements are used in different software projects. Various non-functional requirements include Availability, Portability, Accuracy/Correctness, Security, Maintainability, Reliability, Scalability, Performance, Reusability, Flexibility, etc.

2.5 Traceability

Traceability is the ability to track every change made to the requirement and link it to the overall requirements management process [19]. It reduces redundancy and increases the efficiency of the requirements management system. With the help of traceability, the effect of changes in requirements can be analysed [20]. If the components involved in requirements elicitation are traceable, then any change to one component can be propagated to the associated components through traceability. Proper indexing and referencing of the components are required to perform tracing. If any errors are found in the indexing and referencing of components, the probability of failure will increase. Ramamoorthy et al. defined traceability as the ability to trace from one entity to another based on the provided semantic relations [21]. A software requirements specification can be traceable if the root of every requirement is distinct and if it facilitates referencing each requirement in subsequent development or enhancement documentation. As stated by Torkar et al. [22], requirements are interlinked with each other, and they are dependent on other artifacts. Traceability enables the tracing of these interdependencies. Successful implementation of tracing leads to success in a shorter period. Thus, traceability plays a great role in managing requirements. Traceability among VPs, FRs, OCs, and NFRs is ensured in this research through the proposed methodology, as shown in Figure 1.

Requirement elicitation techniques and associated tools are key factors in requirements engineering. A standard outline of requirements engineering approaches by Rempel and Mader [23] was visualised, in which a multi-view approach was proposed, including a detailed differentiation of tools and techniques utilised in managing requirements. The comparison was based on 5 items: requirement traceability, requirements integration, requirement prioritising,

requirement status, and customer satisfaction. Several research issues, such as requirement ontology and traceability, were sparked by this study. By ensuring traceability of requirements, the authors hoped to address issues in requirements elicitation, verification, and validation. It is critical to collect requirements from stakeholders, but difficult to do so without first recognising the end user's needs. Scharifi et al. [24] presented a synopsis of requirements elicitation techniques and highlighted key issues faced by industry, despite the numerous techniques proposed for the process. According to this study, there is no single method that can meet all requirements elicitation requirements because there are so many problems, such as insufficiency in clarity regarding the purview of the project, inadequate stakeholder participation, inadequate communication and negotiation skills, ineffective techniques, tight deadlines, inadequate documentation, inadequate requirements management, and inconsistent requirements.

In addition, some issues arose during the implementation of the elicitation techniques that must be resolved for the software to be developed per the client's requirements. This research examined various issues and challenges with requirements elicitation techniques in industry & it was contingent on analysis of previous data. In addition, this research identified the need for creative techniques to enhance existing methods/statistical analyses; therefore, a literature review was conducted. This literature search was performed in view of several keywords, namely, requirements elicitation, requirements challenges, and validation techniques. Eventually, a critical analysis of the available literature was conducted using thematic analysis. The results achieved were also analysed using various parameters. Future work for this research was identified as developing a systematic strategy to select the most appropriate techniques for collecting software requirements. To enhance the effectiveness of the requirements elicitation process, Tiwari and Rathore [25] presented an approach that demonstrated how to select a subset of requirements elicitation techniques to achieve an ideal outcome. Their strategy consisted of three steps. In the first, they identified various attributes in three important dimensions: the project, the people, and the software development process, which could create an impact on the outcome of the elicitation process. In the subsequent step, they independently developed a three-P framework (3 PM) for each aspect, which posits a connection between the elicitation methods and the three elements of a product. In the third step, they gave a planning model and used it to determine an elicitation method from the subset of elicitation procedures. Case studies were conducted to evaluate and disseminate contextual knowledge on the selection of requirements elicitation techniques in line with their proposed methodology. Basically, they sought to demonstrate significant differences among the properties of various requirements elicitation methods. They expressed their future plans for incorporating industry data as 3P matrices and analysing these matrices, using an organisation-centric case study to select elicitation techniques.

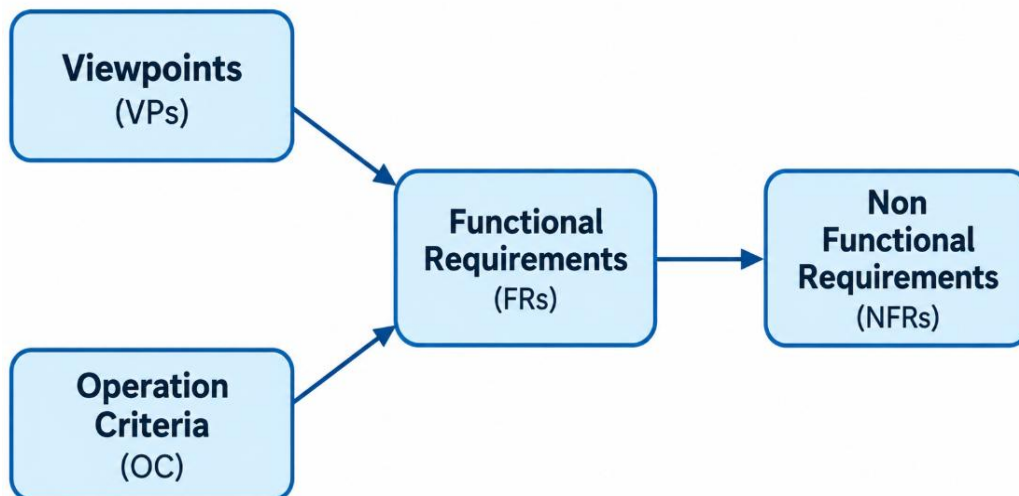


Figure 1. Traceability among VPs, FRs, OC, and NFRs

Proper requirements change management is essential for continuously changing requirements. Traceability is significant for managing these changes and for analysing their effects. Bashir and Qadir [26] presented a synopsis of existing techniques in the field of traceability. According to the authors, current traceability techniques were inadequate, leading to problems in managing changing requirements effectively. So, they classified techniques into three classes and combined them to recognise the demerits of each other, ensuring maximum benefits from traceability. This paper focused on the complications regarding the definitions and techniques of traceability in the literature. A general categorisation of techniques was proposed to enable practitioners to easily select the technique that best meets their needs from the existing options. A software traceability approach was proposed in [27] to support impact analysis of changes in object-oriented software. In this approach, the ability to integrate high-level software models with low-level software models, along with requirements, test cases, design, and code, was demonstrated. This research supported both top-down and bottom-up traceability along with the provision of a direct connection between a component at one level and other components at various levels. The authors developed a software prototype, Catia, to assist C++ software, conducted a case study on an embedded system, and discussed the results. To incorporate requirements into the low-level components, they combined

dynamic and static analysis methods. In a thorough examination, they established an interconnection among the requirements, experiments, and execution code. They provided connections between design model components and code components in static analysis. Because an artefact perspective can provide direct access to the impacted artefacts, the strategy of incorporating both top-down and bottom-up impacts supported provision for efficiency. By applying this proposed strategy within a large framework, the framework's presentation could be corrupted. In large systems, maintenance personnel were typically more interested in acknowledging which classes or methods required modification than in the code's specific statements [28]. In conclusion, this prototype offered traceability infrastructure that included several measures to mitigate the impact of changes. Future work was mentioned as improving the support for GUI (Graphical User Interface) views linked to program views to enhance user interface performance.

Many research studies have been performed on requirement elicitation or traceability. Proper selection of requirement elicitation techniques is significant, as inappropriate requirements may cause high-level user disappointment and decrease the quality of the software [29]. Proper documentation of elicited requirements in line with the client's needs is very useful [30]. It contributes to processing further changes to any requirement. As requirement elicitation can be considered the catalyst that can affect the success of the software product as well as the entire software project, and traceability can be an effective solution to the problems in this regard, a link has been created between requirement elicitation and traceability through this research work. Four components and a six-step procedure have been introduced to elicit requirements and ensure traceability among the corresponding components used in this research. The four components are as follows: viewpoints, functional requirements, operation criteria, and non-functional requirements. Direct Stakeholders can affect the functionality of a system, so the viewpoint method is used to identify the stakeholders of a system as viewpoints and to elicit their corresponding functional requirements as the system's functionalities in this research work [31]. Identification of viewpoints, elicitation of functional requirements, identification of operation criteria, elicitation of non-functional requirements, building relationships between functional requirements and viewpoints, functional requirements and operation criteria, and functional requirements and non-functional requirements, and eventually the traceability among viewpoints, functional requirements, operation criteria, and non-functional requirements are performed correspondingly. A tool has been implemented to ensure traceability among these corresponding components. In addition, the proposed approach is applied to an Automated Teller Machine using the newly developed tool. This research ensures transparency among the stakeholders in accessing the system. Similarly, it increases the client's confidence in the software project team. The ease of accessibility and maintainability of traceability are provided according to the proposed approach. It reduces redundancy and increases the efficiency of the components used in ensuring traceability.

3. Materials and Methods

According to the proposed approach, there are six steps for performing requirements elicitation and traceability analysis using four components (VPs, FRs, OC, and NFRs). The description and Overview Diagram (Figure 2) of the proposed methodology are given below.

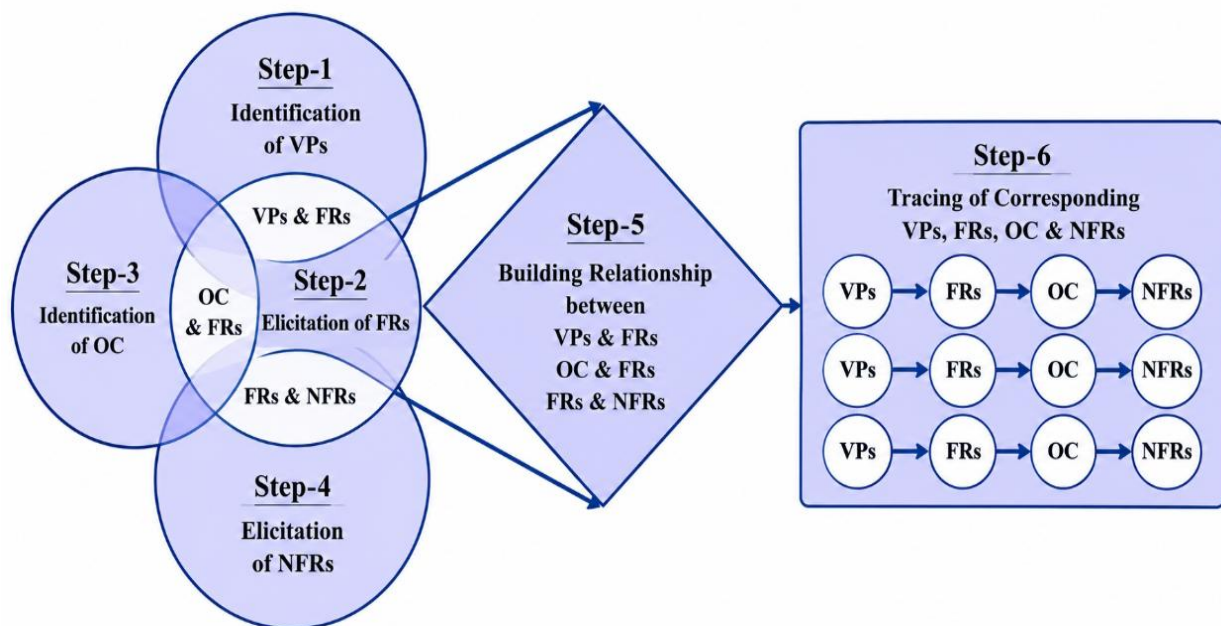


Figure 2. Overview diagram

- **Step-1:** Identifying the VPs of the corresponding system.
- **Step-2:** Eliciting the FR from the activities of the corresponding VPs accordingly.
- **Step-3:** Identifying and setting some OC for the corresponding system.
- **Step-4:** Eliciting the NFRs for the corresponding system and setting some questions to determine the status of the NFRs in the following system.
- **Step-5:** After identifying VPs and OC and eliciting FRs and NFRs, build the relationships between VPs and FRs, OC and FRs, and FRs and NFRs. Correspondingly, VPs, OC, and NFRs have separate relationships with FRs.
- **Step-6:** Finally, trace the corresponding VPs, FRs, OC, and NFRs using the relationships between VPs and FRs, OC and FRs, and FRs and NFRs.

3.1 Eliciting FR

In this research, the viewpoint method is used for eliciting functional requirements. Different VPs have different FRs, as shown in Figure 3.

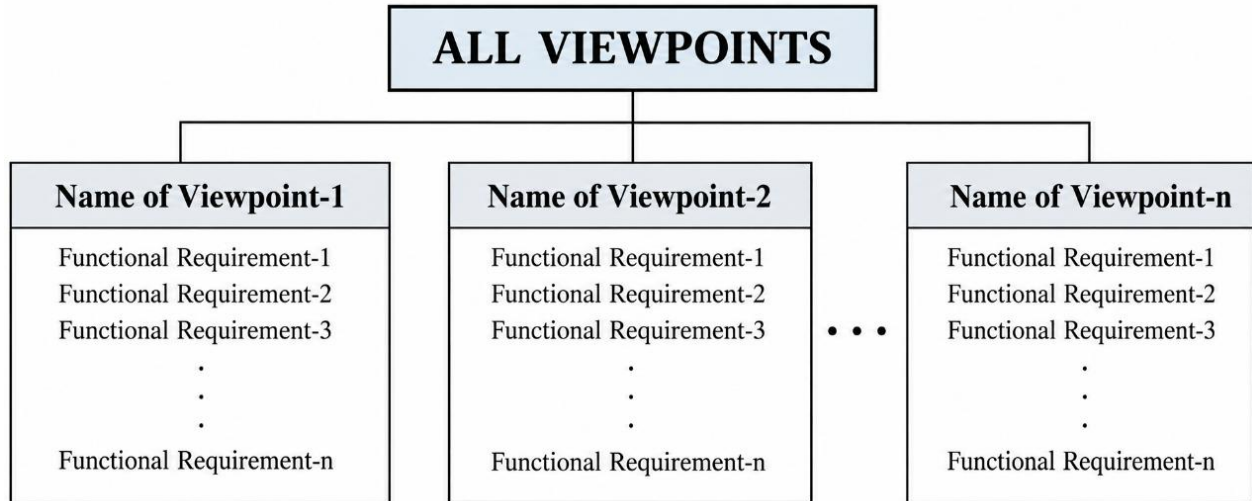


Figure 3. Generalised diagram of viewpoint method for eliciting functional requirements

3.2 Eliciting NFR

According to the proposed approach, the author has selected the following 5 non-functional requirements as the standard NFRs that have to be ensured constantly to various extents in all sorts of projects:

- Availability
- Performance
- Security
- Accuracy/Correctness
- Reliability

To ensure the status of these non-functional requirements in a system, a pre-defined set of dichotomous questions on NFR (NFRQ): Availability, Performance, Security, Accuracy or Correctness, and Reliability will be asked to the stakeholders of the system as shown in Table 1.

Table 1. Generalised NFRQ/A for eliciting NFR

NFR ID	NFR Name	NFR Question	Answer
NFR-1	Availability	Is this service available?	Yes/No
NFR-2	Performance	Does the service process the user's request according to the predefined time?	Yes/No
NFR-3	Security	Are the credentials of the corresponding user secured while accessing the service?	Yes/No
NFR-4	Correctness/Accuracy	Does the service function correctly?	Yes/No
NFR-5	Reliability	Can the users rely on this system to access the service?	Yes/No

3.3 Building Relational Tables

According to the proposed approach, each viewpoint has one or more functional requirements, as shown in the Overview Diagram (Figure 2). So, there is a relationship between VP and FR. The corresponding FRs for a VP are shown in Table 2. According to the proposed approach, each functional requirement has a dependency relationship with one or more operation criteria, as shown in the Overview Diagram (Figure 2). The OC of a dependent FR is shown in Table 3. According to the proposed approach, each functional requirement is related to one or more non-functional requirements, as shown in the Overview Diagram (Figure 2). The corresponding NFRs of an FR can be viewed in Table 4.

Table 2. Sample of a relational table showing the relationship between VP and FR

VP ID	FR ID
V1	FR-1.x FR-1.n
.	.
.	.
Vn	FR-n.x FR-n.n

Table 3. Sample of a relational table showing the relationship between OC and FR

FR ID	Dependent on 'OC ID'
FR-1.x	Select from OC-x to OC-n
FR-1.n	Select from OC-x to OC-n
.	.
.	.
FR-n.x	Select from OC-x to OC-n
FR-n.n	Select from OC-x to OC-n

Table 4. Sample of a relational table showing the relationship between FR and NFR

FR ID	NFR ID
FR-1.x	Select from NFR-1 to NFR-5
FR-1.n	Select from NFR-1 to NFR-5
.	.
.	.
FR-n.x	Select from NFR-1 to NFR-5
FR-n.n	Select from NFR-1 to NFR-5

3.4 Ensuring Traceability

In this research, VP, FR, OC, and NFR can be traced correspondingly using the relationship between VP and FR, OC and FR, and FR and NFR as shown in the Overview Diagram above (Figure 2). According to the proposed approach, the corresponding VP, FR, OC, and NFR can be viewed from the following sample of ensuring traceability in Table 5.

Table 5. Traceability among VP, FR, OC and NFR

VP ID	FR ID	OC ID	NFR ID
V1	FR-1.x	Select from OC-x to OC-n	Select from NFR-1 to NFR-5
	FR-1.n	Select from OC-x to OC-n	Select from NFR-1 to NFR-5
.	.	.	.
.	.	.	.
Vn	FR-n.x	Select from OC-x to OC-n	Select from NFR-1 to NFR-5
	FR-n.n	Select from OC-x to OC-n	Select from NFR-1 to NFR-5

3.5 Case Study

A case study is applied to an Automated Teller Machine to perform requirement elicitation and traceability analysis according to the proposed approach.

ATM:

An ATM (Automated Teller Machine) is an electronic machine that was invented for performing financial transactions. According to the term 'automated', no banking representative, teller, or human cashier is required for the banking platform ATM. It can be considered a self-service banking outlet [32].

Components of ATM:

The automatic teller machine consists of four input devices and four output devices, which are described and shown in Figure 4.

Input Devices

- Keypad: The keypad allows the user to enter the secret identification number of the card for authentication.
- Card Reader: After providing the correct secret identification number, the card reader reads the information on the user's card and passes it to the host server.
- Screen Buttons: It takes the desired service input from the user and processes the user's request.
- Deposit Slot: It receives cash/cheque from the user for deposit purposes.



Figure 4. ATM

Output Devices

- Speaker: It provides audio instructions to the user when the user performs any action in the ATM.
- Display Screen: It displays the main menu, each step, and information about all services, allowing the user to carry out the transaction and other activities.
- Receipt Printer: Prints the user's transaction records and statements.
- Cash Dispenser: It dispenses the amount of money that the user wants to withdraw.

Features of ATM

Automated Teller Machine provides 24×7 access. It reduces the hassle for customers of standing in long queues at the bank for simple transactions and reduces the workload of bank officials. Some important features of an ATM are described and shown in Figure 5.

- Checking balance
- Withdrawing balance
- Transferring balance
- Depositing cash
- Depositing cheque
- Ordering bank statement
- Ordering cheque
- Issuing Mini statement



Figure 5. Steps of performing a transaction using an ATM

Connection with the Database Management System (DBMS)

ATM uses a DBMS from which all information about the bank's account holders can be found. A connection between the ATM and the database management system must be established to perform any transaction and process the user's request. No action will affect the ATM without establishing this connection. After withdrawing and depositing cash using the ATM, the new balance gets updated in the database management system. Thus, the database management system can be considered the back end, and the ATM interface the front end, as shown in Figure 6.

3.6 Implementation

A case study is applied to an Automated Teller Machine to elicit requirements and ensure traceability according to the proposed approach. As discussed in the proposed approach, the 6 steps that are required for eliciting requirements and ensuring traceability among the corresponding components used in eliciting requirements of the ATM are performed as follows. According to the proposed approach, following Steps 1 and 2, the viewpoints for ATM have been identified as shown below: (1) ATM Operator, (2) Account Holder, (3) Other Customer, and (4) System. The numbering of the VPs for ATM has been performed as shown in Table 6. Following Step 1 and Step 2, the viewpoint diagram for ATM is shown in Figure 7, where the FRs of all the VPs have been elicited correspondingly. The numbering of the FRs of the corresponding VPs for ATM has been performed as shown in Table 7.

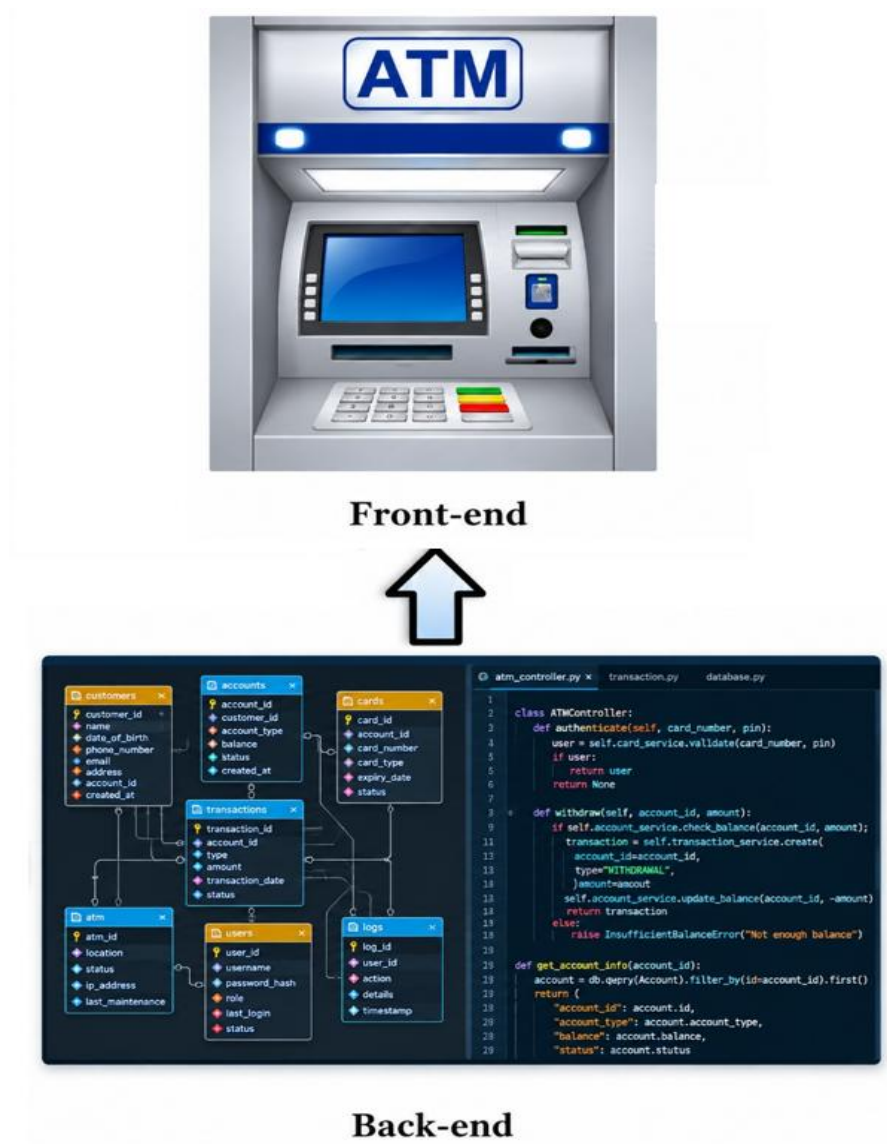


Figure 6. Connection between ATM and database management system

Table 6. Definition table for VP

Viewpoint ID	Name
V1	ATM Operator
V2	Account Holder
V3	Other Customer
V4	System

Table 7. Definition table for FR

FR ID	Description
FR-1.1	Add cash
FR-1.2	Add paper
FR-1.3	Run diagnostics
FR-2.1	Check balance
FR-2.2	Order bank statement
FR-2.3	Withdraw money
FR-2.4	Transfer balance
FR-2.5	Mini statement

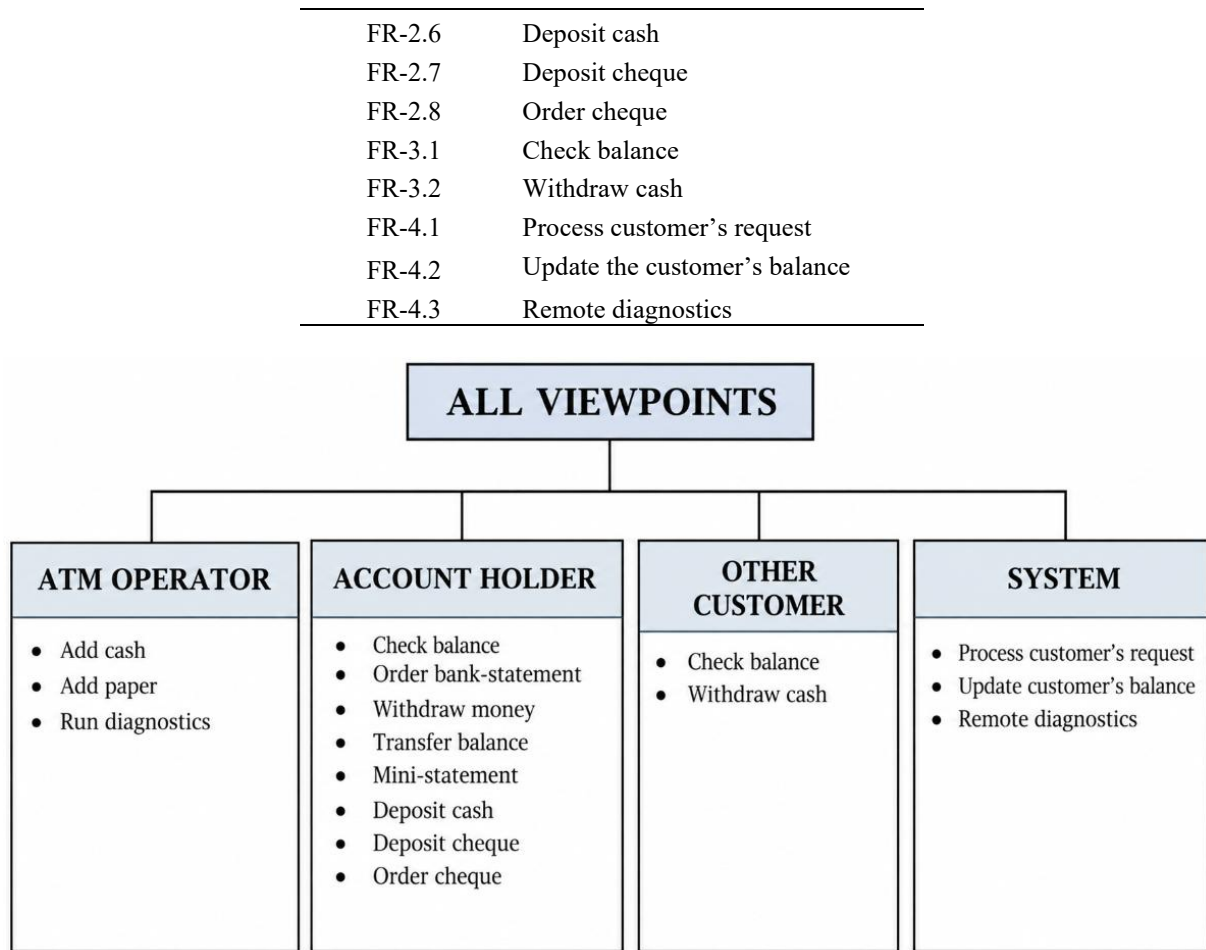


Figure 7. Viewpoint diagram for eliciting FRs

According to the proposed approach, Step 3 involves identifying and defining the operation criteria (OC) for the ATM case study. The identified OC represent the operational conditions associated with the system requirements and are assigned unique numbers to facilitate systematic identification and traceability. This identification provides a structured basis for determining the operational criteria associated with the functional requirements of the ATM system. The identified and numbered OC are presented in Table 8. Step 4 focuses on the elicitation of non-functional requirements (NFRs), where the relevant NFRs are selected, and corresponding questions are formulated to support the elicitation process. These questions assist in identifying the quality attributes and constraints that should be considered during the operation of the ATM system. The elicited and numbered NFRs are presented in Table 9. Step 5 establishes the relationships among the identified requirements to support requirement dependency and traceability. First, the relationships between the viewpoints (VPs) and functional requirements (FRs) are established, as presented in Table 10. Next, the relationships between the FRs and OC are established based on their identified dependencies, as shown in Table 11. Finally, the relationships between the FRs and NFRs are established to identify the relevant quality requirements associated with each functional requirement, as presented in Table 12.

Table 8. Definition table for OC

OC ID	Description
OC-1	Receipt paper must be present in the machine
OC-2	Money must be present in the machine
OC-3	The ATM must be properly connected to the database
OC-4	All the services that the ATM is displaying must be valid and active
OC-5	The credentials of all the customers must be present & updated in the database
OC-6	A pre-defined minimum amount of balance must be present in the customer's account

Table 9. Definition table for NFR

NFR ID	Name	Question
NFR-1	Availability	Is this service available?
NFR-2	Performance	Does the service process the user's request according to the predefined time?
NFR-3	Security	Are the credentials of the corresponding user secured while accessing the service?
NFR-4	Correctness/Accuracy	Does the service function correctly?
NFR-5	Reliability	Can the users rely on this system to access the service?

Table 10. Relational table of VP and FR

VP ID	FR ID
V1	FR-1.1
	FR-1.2
	FR-1.3
	FR-2.1
	FR-2.2
V2	FR-2.3
	FR-2.4
	FR-2.5
	FR-2.6
	FR-2.7
V3	FR-2.8
	FR-3.1
	FR-3.2
V4	FR-4.1
	FR-4.2
	FR-4.3

Table 11. Relational table of FR and OC

FR ID	Dependent on 'OC ID'
FR-1.1	OC-2
FR-1.2	OC-1
FR-1.3	OC-4
FR-2.1	OC-3, OC-4, OC-5
FR-2.2	OC-3, OC-4, OC-5
FR-2.3	OC-2, OC-3, OC-4, OC-5, OC-6
FR-2.4	OC-3, OC-4, OC-5, OC-6
FR-2.5	OC-1, OC-3, OC-4, OC-5
FR-2.6	OC-3, OC-4, OC-5
FR-2.7	OC-3, OC-4, OC-5
FR-2.8	OC-3, OC-4, OC-5
FR-3.1	OC-3, OC-4, OC-5
FR-3.2	OC-2, OC-3, OC-4, OC-5, OC-6
FR-4.1	OC-3, OC-5
FR-4.2	OC-3, OC-5
FR-4.3	OC-3, OC-4

Table 12. Relational table of FR and NFR

FR ID	NFR ID
FR-1.1	Null
FR-1.2	Null
FR-1.3	NFR-1, NFR-2, NFR-4
FR-2.1	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-2.2	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-2.3	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-2.4	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-2.5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-2.6	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-2.7	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-2.8	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-3.1	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-3.2	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
FR-4.1	NFR-2, NFR-3, NFR-4
FR-4.2	NFR-3, NFR-4
FR-4.3	NFR-1, NFR-4

Table 13. Final output (Traceability among VP, FR, OC and NFR)

VP ID	FR ID	OC ID	NFR ID
V1	FR-1.1	OC-2	Null
	FR-1.2	OC-1	Null
	FR-1.3	OC-4	NFR-1, NFR-2, NFR-4
V2	FR-2.1	OC-3, OC-4, OC-5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-2.2	OC-3, OC-4, OC-5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-2.3	OC-2, OC-3, OC-4, OC-5, OC-6	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-2.4	OC-3, OC-4, OC-5, OC-6	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-2.5	OC-1, OC-3, OC-4, OC-5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-2.6	OC-3, OC-4, OC-5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-2.7	OC-3, OC-4, OC-5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-2.8	OC-3, OC-4, OC-5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
V3	FR-3.1	OC-3, OC-4, OC-5	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
	FR-3.2	OC-2, OC-3, OC-4, OC-5, OC-6	NFR-1, NFR-2, NFR-3, NFR-4, NFR-5
V4	FR-4.1	OC-3, OC-5	NFR-2, NFR-3, NFR-4
	FR-4.2	OC-3, OC-5	NFR-3, NFR-4
	FR-4.3	OC-3, OC-4	NFR-1, NFR-4

4. Results and Discussion

According to the proposed approach, following Step 6, the traceability among VP, FR, OC, and NFR has been ensured correspondingly using the relationship between VP and FR, OC and FR, and FR and NFR for the case study of ATM, as shown in Table 13. Microsoft Visual Studio 2022 has been used to implement the tool to achieve the goal of the proposed approach in the ATM case study, as shown in Figure 8. To summarise, the proposed approach, comprising six steps, has been successfully applied using a newly developed tool to the following four components: VP, FR, OC, and NFR, as shown above. In the first step, the VPs are identified. In the second step, FRs are elicited. In the third step, the OC are identified. In the fourth step, NFRs are elicited. In the fifth step, the relationships among FR, VPs, OC, and NFRs are established. Finally, in the sixth step, traceability among VPs, FR, OC, and NFRs is ensured correspondingly. In addition, a case study has been conducted on a small-scale project entitled Automated Teller Machine, which requires low space and time complexity and uses the newly developed tool.

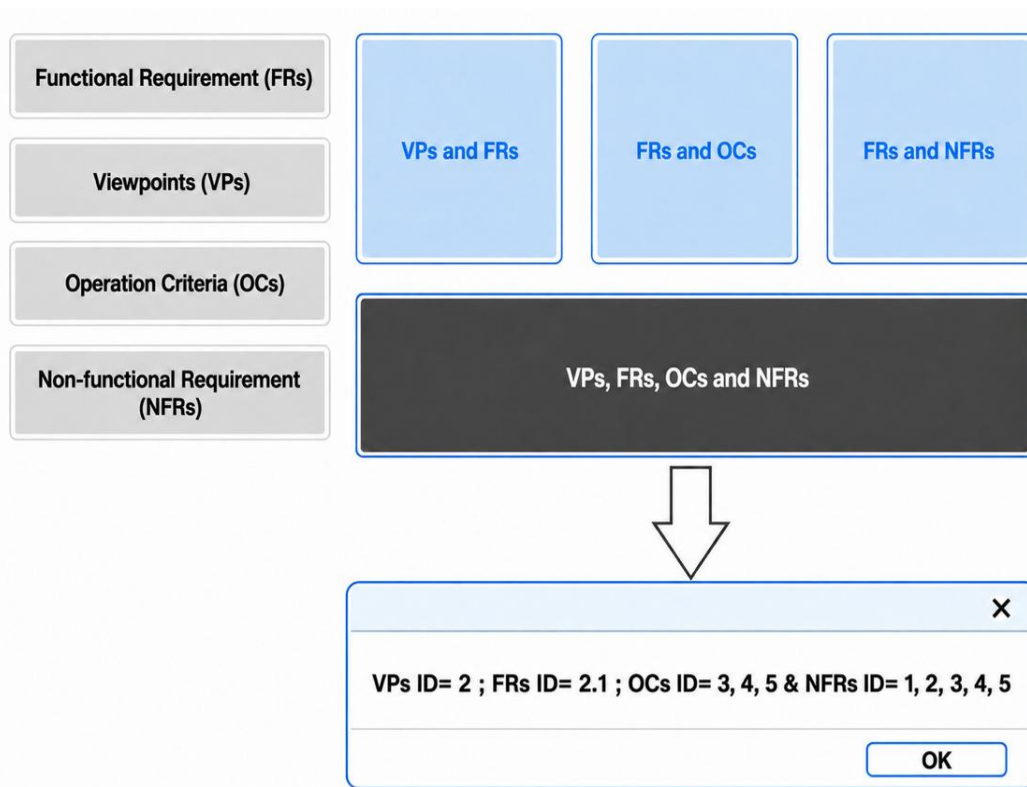


Figure 8. Final output from the tool (Traceability among corresponding VPs, FRs, OC, and NFRs)

5. Conclusion

Finally, an effective approach to eliciting requirements and ensuring traceability among the components used in the elicitation process has been demonstrated in this research. Traceability among VP, FR, OC, and NFR prevents the risk of software project failure at an advanced stage. In this research, a tool has been developed to provide automated traceability. It has shown the desired result in the case study, a small-scale Automated Teller Machine project. However, the scenario will differ for large-scale projects that require high time and space complexity. So, future work can be regarded as the implementation of a large project with high time and space complexity. Moreover, inputs are provided manually to the system according to the proposed approach, which creates a possibility of errors occurring in indexing and referencing. So, another future direction is the creation of an automated system for providing input to the system.

Acknowledgements

The author expresses sincere gratitude to Professor Dr. Shamim H. Ripon of East West University for his guidance and support throughout this research. The author also appreciates the contributions of the editors and reviewers of IJSECS.

Funding

This study was not supported by any grants from funding bodies in the public, private, or nonprofit sectors.

Authors Contribution

Habiba Azrin Mim: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Visualization, Writing – original draft, Writing – review & editing, Supervision.

Conflict of Interest

The author declares no conflicts of interest.

References

- [1] P. Zave, "Classification of research efforts in requirements engineering," *ACM Computing Surveys*, vol. 29, no. 4, pp. 315–321, 1997. <https://doi.org/10.1145/267580.267581>
- [2] T. Shah and S. Patel, "A novel approach for specifying functional and non-functional requirements using RDS (Requirement Description Schema)," *Procedia Computer Science*, vol. 79, pp. 852–860, 2016. <https://doi.org/10.1016/j.procs.2016.03.083>
- [3] A. Jarzebowicz and P. Weichbroth, "A qualitative study on non-functional requirements in agile software development," *IEEE Access*, vol. 9, pp. 40458–40475, 2021. <https://doi.org/10.1109/ACCESS.2021.3064424>
- [4] J. D. Summers, S. Joshi, and B. Morkos, "Requirements evolution: Relating functional and non-functional requirement change on student project success," in *Proceedings of the ASME International Design Engineering*

- Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE)*, Buffalo, NY, USA, 2014, Paper DETC2014-35023. <https://doi.org/10.1115/DETC2014-35023>
- [5] "User stories: An agile introduction," *Agile Modeling*. [Online]. Available: <https://agilemodeling.com/artifacts/userstory.htm>. [Accessed: Dec. 20, 2024].
 - [6] D. E. Avison and G. Fitzgerald, *Information Systems Development: Methodologies, Techniques and Tools*, 4th ed. Berkshire, U.K.: McGraw-Hill Education, 2006.
 - [7] H. F. Hofmann and F. Lehner, "Requirements engineering as a success factor in software projects," *IEEE Software*, vol. 18, no. 4, pp. 58-66, 2001. <https://doi.org/10.1109/MS.2001.936219>
 - [8] P. Loucopoulos and V. Karakostas, *System Requirements Engineering*. London, U.K.: McGraw-Hill, 1995.
 - [9] E. Mumford, *Effective Systems Design and Requirements Analysis: The ETHICS Approach*. Basingstoke, U.K.: Macmillan, 1995.
 - [10] B. Nuseibeh and S. Easterbrook, "Requirements engineering: A roadmap," in *Proceedings of the Conference on the Future of Software Engineering (ICSE '00)*, Limerick, Ireland, 2000, pp. 35-46. <https://doi.org/10.1145/336512.336523>
 - [11] A. Benfell, "Modeling functional requirements using tacit knowledge: A design science research methodology informed approach," *Requirements Engineering*, vol. 26, no. 1, pp. 25-42, 2021. <https://doi.org/10.1007/s00766-020-00330-4>
 - [12] S. Kugele, W. Haberl, M. Tautschnig, and M. Wechs, "Optimizing automatic deployment using non-functional requirement annotations," *Communications in Computer and Information Science*, vol. 17, pp. 400-414, 2008. https://doi.org/10.1007/978-3-540-88479-8_28
 - [13] M. A. Gondal, N. A. Qureshi, H. Mukhtar, and H. F. Ahmed, "An engineering approach to integrate non-functional requirements (NFR) to achieve high-quality software process," in *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS)*, Prague, Czech Republic, 2020, pp. 377-384. <https://doi.org/10.5220/0009568503770384>
 - [14] B. J. M. Abma and I. Kurtev, "Evaluation of requirements management tools with support for traceability-based change impact analysis," M.S. thesis, University of Twente, Enschede, The Netherlands, 2009.
 - [15] B. Ramesh and M. Jarke, "Toward reference models for requirements traceability," *IEEE Transactions on Software Engineering*, vol. 27, no. 1, pp. 58-93, 2001. <https://doi.org/10.1109/32.895989>
 - [16] A. A. Attia and M. A.-K. Soliman, "A new methodology using QFD for tracing requirements in object-oriented software design process," *The Egyptian International Journal of Engineering Sciences and Technology*, vol. 14, no. 3, pp. 288-303, 2011.
 - [17] D. Duarte, C. Farinha, M. M. da Silva, and A. R. da Silva, "Collaborative requirements elicitation with visualization techniques," in *Proceedings of the IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, Toulouse, France, 2012, pp. 343-348. <https://doi.org/10.1109/WETICE.2012.14>
 - [18] I. Sommerville, P. Sawyer, and S. Viller, "Viewpoints for requirements elicitation: A practical approach," in *Proceedings of the IEEE International Conference on Requirements Engineering*, Colorado Springs, CO, USA, 1998, pp. 74-81. <https://doi.org/10.1109/ICRE.1998.667811>
 - [19] M. A. Abbasi, J. Jabeen, Y. Hafeez, D.-B. Batool, and N. Fareen, "Assessment of requirement elicitation tools and techniques by various parameters," *Software Engineering*, vol. 3, no. 2, pp. 7-11, 2015. <https://doi.org/10.11648/j.se.20150302.11>
 - [20] F. Khursheed and M. Suaib, "A survey on importance of requirement traceability in software engineering," *International Journal of Engineering and Innovative Technology*, vol. 5, no. 4, pp. 109-112, 2015.
 - [21] O. C. Z. Gotel and A. C. W. Finkelstein, "An analysis of the requirements traceability problem," in *Proceedings of the IEEE International Conference on Requirements Engineering*, Colorado Springs, CO, USA, 1994, pp. 94-101. <https://doi.org/10.1109/ICRE.1994.292398>
 - [22] R. Torkar, T. Gorschek, R. Feldt, M. Svahnberg, U. A. Raja, and K. Kamran, "Requirements traceability: A systematic review and industry case study," *International Journal of Software Engineering and Knowledge Engineering*, vol. 22, no. 3, pp. 385-433, 2012. <https://doi.org/10.1142/S021819401250009X>
 - [23] P. Rempel and P. Mäder, "Preventing defects: The impact of requirements traceability completeness on software quality," *IEEE Transactions on Software Engineering*, vol. 43, no. 8, pp. 777-797, 2017. <https://doi.org/10.1109/TSE.2016.2622264>
 - [24] E. Scharifi, A. Danilenko, U. Weidig, and K. Steinhoff, "Influence of plastic deformation gradients at room temperature on precipitation kinetics and mechanical properties of high-strength aluminum alloys," *International Journal of Engineering Research and Applications*, vol. 9, no. 6, pp. 42-48, 2019. <https://doi.org/10.9790/9622-0906024248>
 - [25] S. Tiwari and S. S. Rathore, "A methodology for the selection of requirement elicitation techniques," *arXiv preprint arXiv:1709.08481*, 2017. <https://doi.org/10.48550/arXiv.1709.08481>
 - [26] M. F. Bashir and M. A. Qadir, "Traceability techniques: A critical study," in *Proceedings of the 10th IEEE International Multitopic Conference (INMIC)*, Islamabad, Pakistan, 2006, pp. 265-268. <https://doi.org/10.1109/INMIC.2006.358175>
 - [27] S. Ibrahim, N. B. Idris, M. Munro, and A. Deraman, "A requirements traceability to support change impact analysis," *Asian Journal of Information Technology*, vol. 4, no. 4, pp. 345-355, 2005.

- [28] M. Lindvall and K. Sandahl, "Traceability aspects of impact analysis in object-oriented systems," *Journal of Software Maintenance: Research and Practice*, vol. 10, no. 1, pp. 37-57, 1998. [https://doi.org/10.1002/\(SICI\)1096-908X\(199801/02\)10:1%3C37::AID-SMR163%3E3.0.CO;2-R](https://doi.org/10.1002/(SICI)1096-908X(199801/02)10:1%3C37::AID-SMR163%3E3.0.CO;2-R) DBLP
- [29] H. Dar, M. I. Lali, H. Ashraf, M. Ramzan, T. Amjad, and B. Shahzad, "A systematic study on software requirements elicitation techniques and its challenges in mobile application development," *IEEE Access*, vol. 6, pp. 63859-63867, 2018. <https://doi.org/10.1109/ACCESS.2018.2874981>
- [30] E. Serna M. and A. Serna A., "A model for documenting requirements elicitation," *Ingeniería*, vol. 28, no. 2, Art. no. e19411, 2023. <https://doi.org/10.14483/23448393.19411>
- [31] F. Shahbeklu, Requirement Elicitation from Diverse Sources for Software Projects: A Literature Review and Interview Study, M.S. thesis, Blekinge Institute of Technology, Karlskrona, Sweden, 2024.
- [32] "What is ATM-Definition, uses and ATM full meaning," *Paytm*. [Online]. Available: <https://paytm.com/blog/atm/what-is-atm-full-form-definition-uses/>. [Accessed: Dec. 21, 2024].