Sistemasi: Jurnal Sistem Informasi ISSN:2302-8149
Volume 15, Nomor 8, 2026: 3350-3367 e-ISSN:2540-9719

Intelligent Technique for Optimizing Requirements Elicitation in Software Engineering Projects

[1]Esra Zuhair Majeed*
[1]College of Physical Education and Sport Science, University of Mosul, Mosul, Iraq
*e-mail:*esra.zuhair87@uomosul.edu.iq*

(***received*:** 8 August 2026, ***revised*:** 25 August 2026, ***accepted*:** 26 August 2026)

Abstract

Requirements elicitation is a critical activity in software engineering, as incomplete, ambiguous, or inconsistent requirements can adversely affect software quality and project outcomes. Although Large Language Models (LLMs) have demonstrated considerable potential for supporting Requirements Engineering, conventional prompting approaches typically rely on direct or predefined interactions and lack systematic mechanisms for identifying and resolving information gaps during elicitation. This study proposes Adaptive Context-Driven Prompting (ACDP), an iterative framework designed to improve LLM-supported requirements elicitation through context preservation and gap-driven refinement. ACDP progressively constructs an Elicitation Context (EC) that maintains project objectives, stakeholders, constraints, assumptions, and previously elicited requirements. In parallel, a Requirement Gap Matrix (RGM) systematically identifies missing, partial, ambiguous, and conflicting information and transforms these gaps into targeted re-elicitation prompts for subsequent LLM interactions. The framework was evaluated using public software requirements documents from the PURE dataset and compared with Zero-Shot Prompting and Fixed Multi-Step Prompting under controlled LLM settings. Performance was assessed using reference-based Precision, Recall, and F1-score, together with requirement-quality dimensions including completeness, correctness, consistency, clarity, relevance, ambiguity, and redundancy. Experimental results demonstrate that ACDP achieved a Precision of 0.93, Recall of 0.92, and F1-score of 0.92, outperforming both baseline prompting approaches while also improving requirement completeness, correctness, and overall quality. These findings demonstrate that combining persistent elicitation context with explicit gap diagnosis provides a systematic and effective mechanism for improving LLM-assisted requirements elicitation and offers practical support for requirements analysts in complex software projects.

Keywords: adaptive prompting, large language models (LLMs), prompt engineering, requirements elicitation, requirements engineering, requirement gap analysis.

1. Introduction

The most important activity in software project development is the requirements engineering. Requirements Engineering (RE) is a basic discipline of software development that has become even more crucial in an ever more complex software system. The systematic activities involved in discovering, eliciting, analyzing, documenting, validating, and managing software requirements for the developed software system to meet the needs and objectives of the stakeholders is called RE. Requirements elicitation is one of these activities and is the basis for the remaining activities of the software development lifecycle [1]. Therefore, the quality of elicited requirements directly influences the success of software projects. Incomplete, ambiguous, or incorrect requirements often result in slippage, cost escalation, frequent requirement changes, poor software quality, unreliable system behavior and low level of satisfaction among stakeholders. So, one of the biggest challenges in software engineering is to make the requirements elicitation process more effective [2].

The elicitation of requirements has been traditionally carried out with several techniques such as interviews, questionnaires, workshops, brainstorming sessions, observation, document analysis and prototyping. These techniques are widely used as they enable the communication with stakeholders and helps to identify the functional and nonfunctional requirements [3].When implementing an elicitation activity, there are often multiple issues that arise, including lack of knowledge, requirements ambiguity, differing interpretations, communication problems, and changing business

 ISSN:2302-8149
e-ISSN:2540-9719

requirements [4]. The problems can lead to missing or conflicting requirements, more rework, project delays, and higher development costs. As a result, making requirements elicitation more efficient, consistent, and of high quality is an active research topic, and researchers are interested in investigating more intelligent approaches that can help the analysts along the elicitation process [5].

This study introduces an Adaptive Context-Driven Prompting (ACDP) framework for requirements elicitation with the support of Large Language Model (LLM). ACDP differs from traditional prompting methods which capture requirements from a project description or a set of successive cues, by considering elicitation as a process that is iterative and context sensitive. It then gradually builds up an Elicitation Context (EC) that consists of project objectives, stakeholders, requirements, constraints, and assumptions, and adds a Requirement Gap Matrix (RGM) that helps find missing, partial, ambiguous, or conflicting information about requirements. The gaps identified are used to dynamically generate specific re-elicitation prompts for the LLM to use to focus future interactions on the requirement dimensions that have not been sufficiently covered. The proposed framework is tested experimentally with publicly available software requirements documents taken from the PURE (PUblic REquirements) dataset [23].with the original detailed requirements retained as reference specifications. For the same LLM configuration and initial project information, ACDP is compared with Zero-Shot Prompting and Fixed Multi-Step Prompting. Both reference-based and requirement quality characteristics such as completeness, correctness, consistency, clarity, relevance, ambiguity, and redundancy are taken into account in the assessment. This experimental design will allow the study to evaluate if using a prompting method that is adaptive and gap-driven will lead to improvements in the coverage and quality of elicited requirements, as compared to direct and predefined prompting. This study then provides a systematic approach for elicitation based on LLM that combines cumulative context construction, explicit requirement-gap diagnosis, adaptive re-elicitation, and quality-guided refinement in a single process. Instead of relying on the generative ability of an LLM only, ACDP is based on the changing state of the elicited requirements, offering a more controlled and adaptive way for requirements discovery.

Although the intelligent requirements elicitation field has made significant progress recently, an in-depth analysis of the existing literature shows that there is a gap in research. The traditional methods are still very dependent on the analyst's expertise and the involvement of stakeholders, while machine-learning/recommender-system methods focus mainly on choosing elicitation methods, determining the set of stakeholders, forecasting requirements or facilitating prioritisation. The more recent NLP-, deep-learning- and LLM-based methods offer more automation, but are limited to requirement extraction, classification, or specific types of contextual analysis. Relatively little research has focused on an iterative elicitation mechanism that continually maintains the context of the evolving project, that explicitly identifies gaps in the growing set of requirements, and that dynamically decides what information should be elicited next.

In this study, we propose Adaptive Context-Driven Prompting (ACDP) that combines an Adaptive Elicitation Context (EC), a Requirement Gap Matrix (RGM), and adaptive re-elicitation, all presented in a single iterative process. The novelty of ACDP is not its ability to directly generate these requirements, but in its ability to use the dynamically identified state of requirement gaps to steer further interactions with the LLM to solve the problem.

This paper is organized as follows: the remaining of this paper is organized. Section 2 summarizes the related works of the traditional and intelligent requirements elicitation approaches. The description of requirements elicitation process and quality considerations followed in this study is given in Section 3. The proposed ACDP framework is discussed in Section 4 and the experimental design and evaluation procedure is discussed in Section 5. The results are presented and discussed in Section 6, threats to validity in Section 7, and conclusions and future work in Section 8.

2. Literature Review

The methods and approaches of requirement elicitation have taken a left turn from being purely human based to being increasingly data and intelligence oriented. The past studies focused on choosing appropriate elicitation techniques, understanding the contextual factors that affect the effectiveness of these techniques, while the recent studies explored the application of recommendation systems, machine learning, deep learning and other intelligent mechanisms to assist various elicitation

 ISSN:2302-8149
e-ISSN:2540-9719

activities. This section presents them in turn, starting with the traditional requirements elicitation practice and then intelligent and machine learning-based approaches to requirements elicitation.

2.1 Traditional Requirements Elicitation Techniques

The traditional elicitation of requirements is based on direct interaction between the requirements analysts and the stakeholders and is most often achieved by means of interviewing, workshops, observation, prototyping and user feedback. While these methods are still popular, their success will depend on the nature of the project, the participation of stakeholders and the analysts' skills. Batra and Bhatnagar [6] have suggested five factors that are considered while choosing the most suitable elicitation technique: project, development process, organizational, stakeholder and analyst. Their study showed the role of the context and the analysts' experience in the selection of the technique, but it does not facilitate the elicitation process itself.

In this practical sense, King [8] explored different approaches to requirements elicitation, using a multi case study of contemporary product development. The study revealed that a wide range of agile-like practices were actually used, especially user feedback and prototyping; some of the challenges that were identified included incompleteness of the requirements, limited user availability, insufficient validation, and challenges to elicit non-functional requirements. The results reveal that while traditional elicitation techniques continue to be relevant, there are significant restrictions that support the need for more automatic and intelligent means of support.

2.2 Intelligent and Machine Learning-Based Requirements Elicitation

Conventional elicitation has spurred research into computational methods that will help analysts identify stakeholders, choose elicitation methods, anticipate requirements, and manage and process a range of requirement-related information, especially in the case of large sets of requirements. As a result, the early intelligent approaches were mostly based on recommendation and machine learning algorithms, and later studies started to bring more advanced NLP and deep learning techniques into play. Recently, Large Language Models have also enabled intelligent support beyond classification and recommendation, to the analysis of the information from the stakeholders in context. De Ninno et al. [7] investigated how LLMs can be used in early Requirements Engineering by providing a narrative of user needs and experiences using role-playing scenarios, in order to understand and record users' implicit moral orientations and preferences. Although their research centered on the moral profiles and not necessarily on generic software needs, it did reveal the ability of LLMs to identify tacit knowledge from stakeholders that may not be explicitly communicated during the regular elicitation process.

Gobov and Huchenko [2] used machine learning to suggest the most suitable requirements elicitation techniques, based on project attributes and analyst factors. The data gathered from the 328 practitioners was used to evaluate the predictive performance of their Decision Jungle models, which proved to be promising; however, the approach was aimed not at discovering and refining requirements, but at selecting specific elicitation techniques. Another line of research was work by Mulla and Girase [9] that used a stakeholder recommender approach using the k-Nearest Neighbor (kNN) algorithm. They developed this approach to ensure that they can detect similar stakeholder profiles, anticipate potential needs and help with the needs prioritization for large projects, which is particularly important when there is too much information and stakeholder involvement. Its main focus, however, was the recommendation and prioritization, not interactive requirements elicitation. The overall evolution of this line of research was then studied by Akram et al. [13] who conducted a systematic literature review to understand the requirements elicitation studies based on recommender systems that have been published over the years since 2009 up to 2022. They analyzed that recommender systems have been used for stakeholder identification, requirement prediction and prioritization, and automated selection of elicitation techniques, and that there is still a huge potential for further automation and for more complete support of elicitation. More recent research has branched out from Recommendation Systems to more specialized intelligent methods.

Fedorova et al. [10] have suggested a continuous assessment-driven approach to collecting reliable AI requirements by modifying the EU Assessment List for Trustworthy Artificial Intelligence (ALTAI). The approach takes lifecycle awareness of ethical issues as a starting point, and infers related functional and non-functional requirements, showing how structured assessment can support early-stage requirements elicitation for the development of AI-systems. On a more generic level, Martins et al. [11] compiled evidence from nine secondary studies of RE for machine-learning-based

 ISSN:2302-8149
e-ISSN:2540-9719

systems. They found that non-functional issues, collaborative working between stakeholders and industrial adoption were still recognized as a challenge both in their secondary and tertiary studies, revealing the introduction of new NFRE issues in the field of intelligent systems that do not seem to be addressed by the current practices. Under the theme of moving from decision support to automatic elicitation of the information, Hanna et al. [12] combined deep learning and natural language processing with the aim of extracting and categorizing requirements into functional, non-functional and non-requirement categories. The substantial improvement in the classification performance, resulting from their deep neural network trained on client and user-story transcripts, is the promise of advanced learning techniques to automate important aspects of the elicitation process.Table 1 summarizes the reviewed studies by highlighting their main elicitation focus, adopted approach, key contribution, and major limitation in relation to intelligent requirements elicitation.

Table 1 Summary and comparative analysis of previous requirements elicitation approaches.

Ref.	Study	Technique / Technology	Contribution / Focus	Main Limitation
[6]	Batra and Bhatnagar	Five-factor approach	Context-aware selection of elicitation techniques using project, process, organizational, stakeholder, and analyst factors	Supports technique selection rather than elicitation itself
[8]	King	Multi-case study; agile-like practices	Examined practical elicitation through user feedback and prototyping and identified recurring real-world challenges	Incomplete requirements, limited user availability, and insufficient validation remain unresolved
[2]	Gobov and Huchenko	Machine Learning; Decision Jungle	Predicted suitable elicitation techniques using project and analyst characteristics from 328 practitioners	Recommends predefined techniques without directly eliciting or refining requirements
[9]	Mulla and Girase	kNN recommender system	Supported stakeholder identification, requirement prediction, and prioritization in large projects	Limited to recommendation and prioritization rather than interactive elicitation
[13]	Akram et al.	Systematic Literature Review	Synthesized recommender-system applications in stakeholder identification, requirement prediction, prioritization, and technique selection	Identified the need for greater automation and comprehensive elicitation support
[10]	Fedorova et al.	ALTAI-based continuous assessment	Elicited functional and non-functional trustworthy-AI requirements considering lifecycle-related ethical concerns	Specialized to trustworthy AI systems
[11]	Martins et al.	Tertiary study	Synthesized RE evidence for ML-based systems and highlighted NFR and stakeholder-collaboration challenges	Provides research synthesis rather than a direct elicitation technique
[12]	Hanna et al.	Deep Learning, NLP, DNN	Automated extraction and classification of functional, non-functional, and non-requirement information	Focuses mainly on extraction and classification rather than interactive discovery
[7]	De Ninno et al.	LLMs; role-playing scenarios	Used context-rich narratives and LLM analysis to capture tacit stakeholder orientations and preferences	Focused on moral profiles rather than general software requirements

 ISSN:2302-8149
e-ISSN:2540-9719

Table 1 indicates that the research has gradually shifted from selecting techniques according to the context to recommender systems, machine learning, deep learning and NLP and LLM based support. However, the majority of methods still rely on a single elicitation activity instead of giving systematic support at different phases of the requirements elicitation process.

3. Requirements Elicitation Process and Quality Considerations

Requirements Elicitation is where stakeholders' knowledge and the formal definition of a software system meet. It is not only important to gather stakeholders' statements, but also important to interpret the context of the project, to uncover implicit needs, to cover missing information or ambiguity, and to turn informal knowledge into requirements that can effectively support further development activities. As a result, the elicitation process and what constitutes good quality requirements is critical to the design of intelligent techniques that are designed to support or improve this activity[14].

3.1 Requirements Elicitation Process

Requirements elicitation is a systematic process through which requirements engineers identify, explore, and refine the needs, expectations, constraints, and priorities of stakeholders for a software system[15]. Rather than being limited to the initial collection of user statements, elicitation involves understanding the project context, identifying relevant stakeholders, discovering functional and non-functional requirements, resolving incomplete or unclear information, and progressively refining the obtained requirements[16]. The process is inherently iterative because stakeholders may initially provide incomplete, implicit, or conflicting information, requiring analysts to formulate follow-up questions and revisit previously elicited requirements. Effective elicitation therefore depends not only on communication with stakeholders but also on the analyst's ability to interpret contextual information, identify information gaps, and transform informal stakeholder knowledge into clear and usable software requirements[17]. The entire requirements elicitation process is described in Figure 1 to demonstrate how the information from stakeholders can be slowly transformed into the fine-grained software requirements.

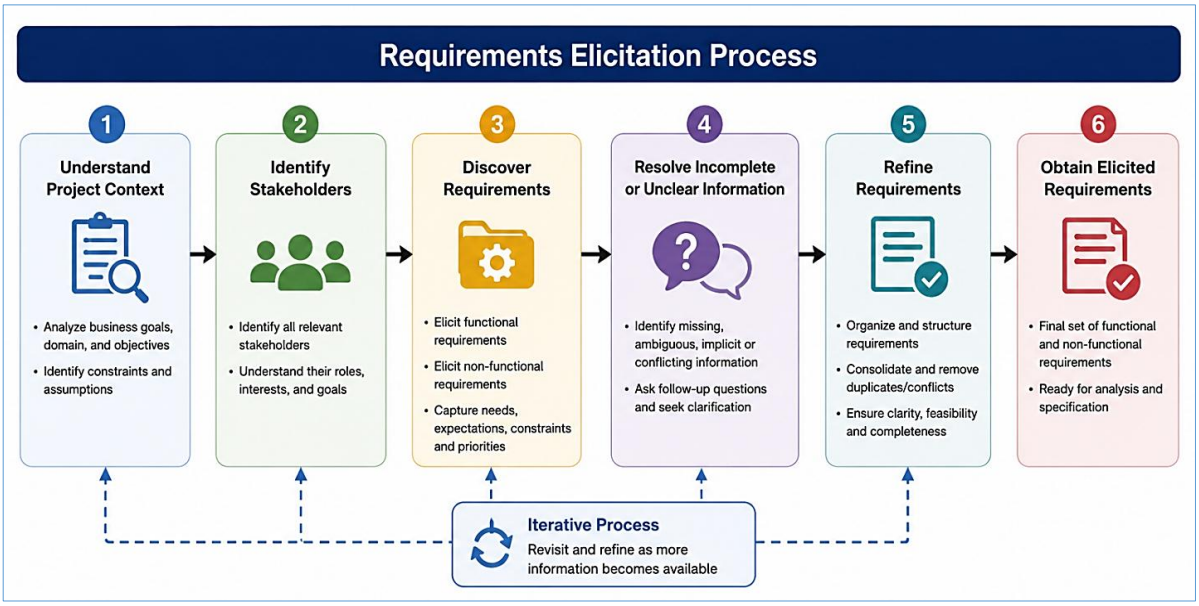

Figure 1 Overview of the requirements elicitation process

Figure 1 depicts the overall software requirements elicitation process in software engineering projects. This process begins with working out the project context and identifying the stakeholders that are relevant to the project. The elicitation process aims to identify both functional and non-functional requirements. It should be observed that stakeholders usually provide inadequate, unclear, or conflicting information in the beginning, thus, the process of elicitation is iterative due to the fact that continuous clarification, refinement, and validation of the obtained information is required. Since

 ISSN:2302-8149
e-ISSN:2540-9719

this is an iterative process, it is necessary that requirements engineers actively work with the information in order to identify the contextual information that is useful, find the gaps in the knowledge they have, and create follow-up questions for requirement clarification. The outcome of this whole process is a collection of structured requirements for software.

3.2 Quality of Elicited Requirements

The efficiency of the process of requirements elicitation can be seen through the quality of the requirements produced. Quality requirements should represent the stakeholders' needs accurately and reduce information loss and ambiguity[18]. In this research study, the quality of the requirements that are elicited is observed in terms of the following aspects: completeness, correctness, consistency, clarity, relevance, ambiguity, and redundancy. Completeness means that all of the important stakeholder's requirements and constraints of the system were captured. Correctness shows whether the requirements are correct in their representation[19]. Consistency means the contradiction-free characters of requirements; clarity and ambiguity relate to the quality of interpretation of requirements. Lastly, relevance makes sure that the requirements created are connected to the project context and redundancy means that requirements don't repeat each other unnecessarily. These characteristics allow one to assess whether intelligent elicitation method allows producing better requirements than methods based on traditional prompting methodologies[20]. The following description of the considered quality dimensions is intended to enable a structured representation of the criteria used to evaluate elicited requirements.The following representation of quality dimensions considered is intended to make a structured representation of the criteria used to evaluate elicited requirements possible.

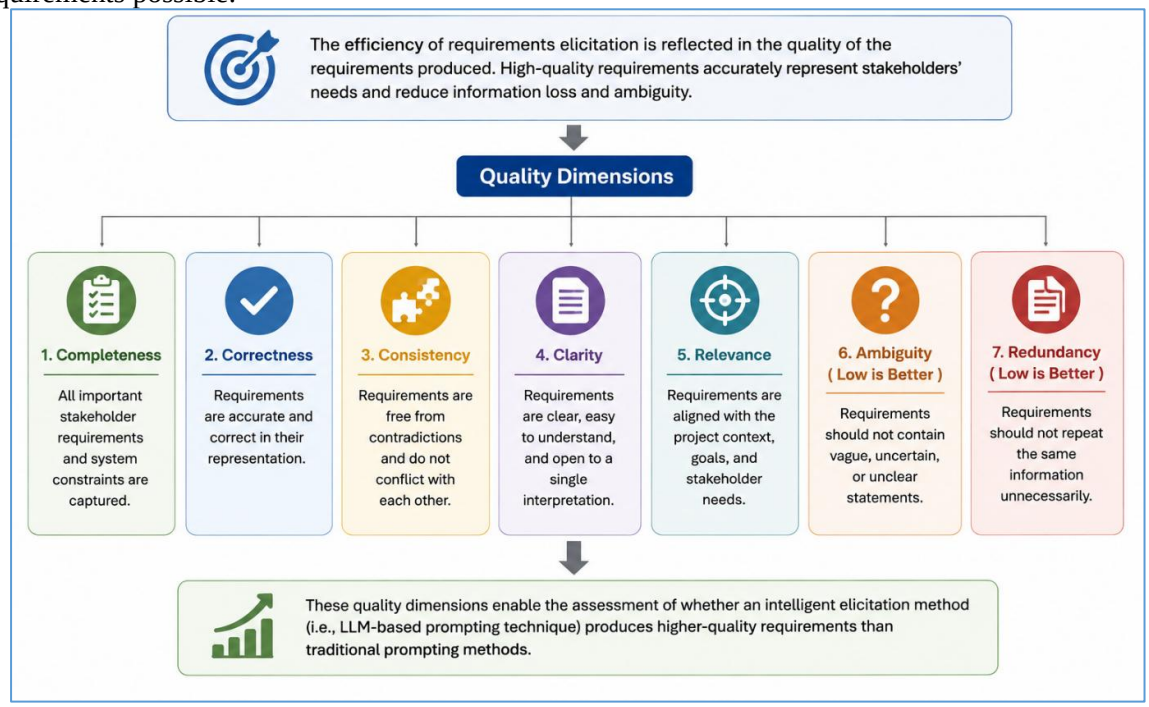

Figure 2 Quality dimensions of elicited software requirements

The overall main quality dimensions of the requirements elicited during the elicitation process are summarized in figure 2. Completeness, correctness, consistency, clarity, relevance, ambiguity and redundancy are among these dimensions. The completeness and correctness of the captured and represented stakeholder needs are considered, while consistency of the resulting requirements is considered. Clarity and ambiguity measure the interpretability and precision of requirement statements, relevant and irrelevant measures how well requirement statements fit the project context and stakeholder needs. Redundancy evaluates whether there is any unnecessary duplication in the requirements set. These dimensions together serve as the criterion for assessing the quality of the requirements produced by the proposed method for generating intelligent requirements with those produced by the traditional methods of elicitation using prompts.

 ISSN:2302-8149
e-ISSN:2540-9719

4. Proposed Methodology

The present study introduces an intelligent method for requirements elicitation, leveraging Large Language Models (LLMs) and adaptive prompt engineering. The approach overcomes the one-shot prompting limitation where an LLM is given a project description and tasked with creating a full set of requirements in one go. This can involve missing information, or incomplete or unclear information, as well as non-functional requirements, contextual constraints, or missing implicit requirements from stakeholders. Rather, the proposed approach considers requirement elicitation to be a context-dependent and iterative process, where the data fed into the LLM is continuously analyzed, augmented, and refined[21].

The proposed method will be called the Adaptive Context-Driven Prompting (ACDP).The proposed method is called (ACDP), which is a series of coordinated stages for requirements elicitation. First considers the context of the project and the stakeholders' point of view, and then looks at the information sources that have been proposed and considers if there were missing, unclear, conflicting or little-explored aspects. The gaps detected in these scenarios are then translated into a set of prompts that help to elicit further descriptions but do not keep re-creating requirements from the original description. Finally, the resulting requirements are elaborated and analyzed based on the quality dimensions discussed in Section 3.2, completeness, correctness, consistency, clarity, relevance, ambiguity and redundancy.

The main idea of ACDP is thus that prompts should be tailored to the gaps in elicitation that are found in the process. This is as opposed to fixed prompt chains which are based on a pre-arranged sequence of prompts regardless of the information retrieved from a specific software project. The methodology is designed to create a more robust and accessible software requirements list through the inherent flexibility of general-purpose LLMs through iterative gap detection, adaptive prompting, and quality-guided refinement[22].

4.1 Overview of the Proposed ACDP Framework

ACDP is a technique for carrying out the requirement gathering and involves a gradual interaction process with the LLM instead of carrying out a single generation procedure. It starts with giving a description of a project that will be later developed through a number of stages, where each preceding stage serves as a ground for the next one. The main aim of the process is to minimize the issues related to incomplete, vague, and missing requirements.

The Elicitation Context (EC) is a key element of ACDP, it is where all project information gathered during the elicitation process is stored and continually updated as the process evolves, including the project objectives, stakeholders identified, constraints, assumptions, and progressively elicited requirements. This changing environment lets the LLM produce subsequent prompts based on the most relevant and current project data. At the same time, the Requirement Gap Matrix (RGM) provides a systematic approach to review the existing requirement set to determine if the information is missing, incomplete, ambiguous, or contradictory. The gaps detected are then fed into targeted re-elicitation prompts, which in turn are fed back into the EC, as shown in Figure 3, in an iterative process of feedback between the three steps of context enrichment, gap diagnosis, and requirement refinement. The resulting requirements are then elaborated on based on the quality dimensions described in section 3.2 until there are no significant gaps anymore.

Figure 3. Overall Architecture of the Proposed Adaptive Context-Driven Prompting (ACDP) Framework

 ISSN:2302-8149
e-ISSN:2540-9719

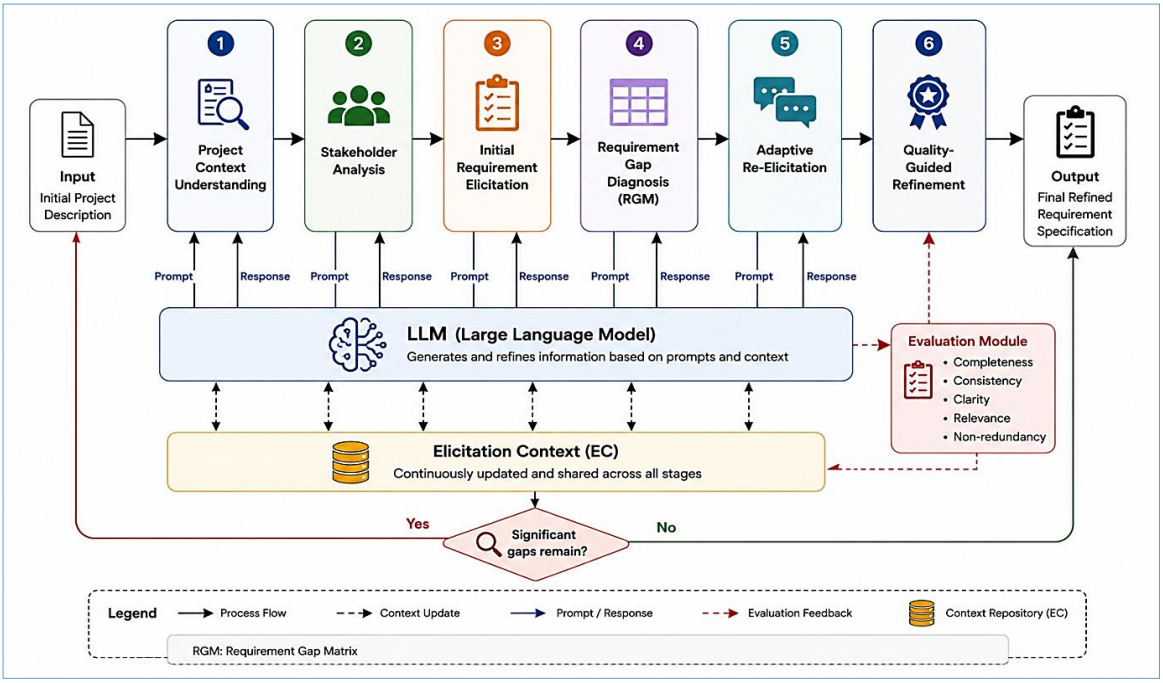

Figure 3 Overall architecture of the proposed adaptive context-driven prompting (ACDP) framework

The workflow of the ACDP model is presented in Figure 4. It begins with an initial description of the project and continues with constructing the evolving Elicitation Context (EC) through stakeholder analysis, context comprehension, and requirement generation processes. The identified requirements are then processed through the requirement gap matrix (RGM), which identifies the information that has not been obtained adequately and provides guidelines for specific re-elicitation measures. The derived requirements are then refined in accordance with specified quality parameters. In case of existence of some gaps, the diagnosis and re-elicitation process is repeated; otherwise, the final product comprising the refined requirement specification is obtained.

4.2 Project Context Understanding and Elicitation Context Construction

The objective of stage one of the Adaptive Context-Driven Prompting (ACDP) framework is to acquire a thorough comprehension of the software project that needs to be worked upon prior to gathering any project requirements. Instead of going straight ahead and requesting that the LLM produce functional and non-functional requirements immediately, this framework comprises of the construction of an Elicitation Context (EC) that takes into account the essential aspects of the project. The EC builds the knowledge base that leads to the capture of all information in further instruction steps.

The stage starts with providing the LLM with the very initial project description acquired from stakeholders or customers. The LLM analyzes this project description by identifying its project domain, business goals, target users, prominent features, limitations, assumptions, and quality expectations (if stated explicitly). Instead of producing project requirements, the aim of the stage is to arrange the information in a way that results in a properly structured project representation that eliminates ambiguity and provides a consistent context for further reasoning. The gathered data is then stored in the Elicitation Context (EC), which will be modified and updated throughout the application of the ACDP. In the course of having gained the new knowledge at further stages, the EC will contain the identified stakeholders and refined requirements that will help the LLM to make the appropriate reasoning and have the most precise project requirements prepared.

At the end of this stage, the framework produces a structured project profile that summarizes the initial understanding of the software system. This profile becomes the primary input to the stakeholder analysis stage and provides a stable contextual basis for adaptive requirements elicitation. To provide a structured representation of the project context, the main information elements extracted by the LLM during this stage are summarized in Table 2.

 ISSN:2302-8149
e-ISSN:2540-9719

Table 2 Information extracted during project context understanding

Context Element	Description
Project Domain	Application domain of the software system
Business Objectives	Primary goals of the project
Target Users	Expected end users and stakeholders
Core Functions	Main services provided by the system
Constraints	Technical, business, or regulatory limitations
Assumptions	Initial assumptions derived from the description

The extracted information is consolidated into a structured Project Profile and stored in the Elicitation Context (EC), which is progressively updated and used as the contextual basis for the subsequent ACDP stages.

4.3 Stakeholder Analysis and Initial Requirements Elicitation

The second step is to define the stakeholders who can interact with, affect or be affected by the proposed software system after building the initial Elicitation Context (EC). The LLM analyzes the project profile to identify the role, goals, responsibilities and likely interaction of the stakeholders in the project with the system. This step helps to broaden the scope of the original project's description by taking into account user needs from a variety of stakeholder perspectives and not just those explicitly expressed.

The LLM then creates an Initial Requirement Set (IRS) based on the stakeholders identified and the EC building up. The requirements are segmented into functional requirements, non-functional requirements, business rules and system constraints. Every requirement is written as an individual, brief statement, and, if possible, linked to the stakeholder or project objective from which it came. This organized representation helps to track the requirements and makes it easier to identify incomplete, ambiguous or missing requirements later. The resulting IRS is not considered the final requirements specification. Rather it is the first elicitation baseline (as used in ACDP) that is passed onto the diagnostic phase where the completeness and adequacy of the elicited information is examined systematically.

4.4 Requirement Gap Diagnosis and Requirement Gap Matrix (RGM)

After the Initial Requirement Set (IRS) is generated, ACDP has a diagnostic stage to ascertain if the elicited information is adequate to represent the software project and the project's stakeholder needs. The LLM does not rush to ask for more requirements, but instead, looks at the IRS along with the cumulative Elicitation Context (EC) to determine what needs to be explored further. The diagnosis also takes into account the lack of the required dimensions and quality issues which might impact the usefulness of the elicited requirements.

In this diagnosis, the suggested framework proposes a Requirement Gap Matrix (RGM) to organize this diagnosis. The RGM analyzes the elicited information on the following aspects of the requirements, which form the key requirement dimensions: stakeholder coverage, functional behavior, non-functional requirements, business rules, system constraints, exception scenarios, dependencies. On each dimension the LLM gives one of five diagnostic states: Covered, Partial, Missing, Ambiguous, and Conflicting. A Covered state means there is enough information available, while Partial and Missing mean there is missing information or it is incomplete. Ambiguous means that there are statements or sections of information that could be interpreted more than one way within the current requirement set; Conflicting means inconsistent statements within the current requirement set.

The RGM is used as a control mechanism for adaptive elicitation. These dimensions are either classified as Covered or are turned into an elicitation objective if the state is not Covered. For instance, if a Missing non-functional requirement is identified, it may lead to questions about security, performance or usability, but if a Missing functional requirement is identified, it will lead to the clarification of the requirement rather than the generation of a new requirement. Thus, ACDP guides the LLM on the information gap rather than repeatedly asking for a set of information requirements. The gap profile that is generated is then fed into the Adaptive Re-Elicitation stage, detailed in Section

 ISSN:2302-8149
e-ISSN:2540-9719

4.5. Table 3 summarizes the diagnostic states and their respective elicitation actions, used by the RGM.

Table 3 Requirement gap matrix (RGM) diagnostic states

RGM State	Meaning	ACDP Action
Covered	Sufficient information is available	No further elicitation
Partial	Requirement information is incomplete	Request missing details
Missing	Expected information is absent	Generate targeted elicitation
Ambiguous	Information allows multiple interpretations	Request clarification
Conflicting	Inconsistent information is detected	Request conflict resolution

This means that the RGM explicitly connects the concept of gap detection to that of prompt adaptation, enabling the following prompts to be tailored to reflect the specific gaps that were found in each project, instead of following a predetermined sequence of questions.

4.5 Adaptive Re-Elicitation through Context-Driven Prompting

After gap diagnosis, ACDP maps an unclosed RGM state to a specific re-elicitation prompt. The prompt is dynamically generated based on three elements: the Elicitation Context (EC), the dimension of the requirement that is affected and the gap type diagnosed. This allows the LLM to concentrate on missing information only without losing the information gathered in the earlier stages of elicitation. Figure 4 shows the structure of the adaptive prompt created by ACDP.

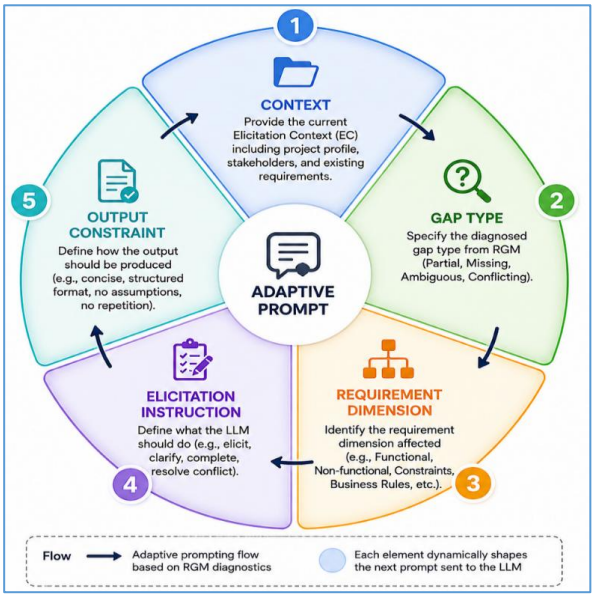

Figure 4 Generic structure of the adaptive prompt in the proposed ACDP framework

Each adaptive prompt is built according to the current EC, the RGM detected gap type, the affected requirement dimension, the required elicitation action and the expected output format (Figure 5). This composition enables the LLM to handle an information gap without having already to elicit any information, and without re-stating the information that has already been captured. For instance, if a security requirement is found to be Missing by RGM, LLM is given the task of analyzing the current project context and only requesting relevant security requirements, but not ones that have already been identified. In an Ambiguous state, clarification is called for, and in a Conflicting state, identification and resolution of inconsistent statements is requested. Table 4 lists the mapping of RGM states to adaptive prompting actions.

4.6 Quality-Guided Refinement and Termination

Each adaptive re-elicitation cycle, the revised requirement set is again reviewed and evaluated in terms of the quality dimensions outlined in Section 3.2; completeness, correctness, consistency, clarity, relevance, ambiguity and redundancy. The revised requirements are also reconsidered using the RGM to assess if there are any substantial information gaps. Fuzzy, duplicate, conflicting, and

 ISSN:2302-8149
e-ISSN:2540-9719

insufficiently defined requirements are clarified and maintained while building on previously verified requirements.

The ACDP process ends when there are no more unresolved Missing, Ambiguous, or Conflicting states in the RGM and all of the critical requirement dimensions have been classified as Covered. When gaps remain if they have not been addressed, they are returned to adaptive prompting back to the next cycle of targeted elicitation. Final output – Final Requirement Set (FRS) with functional and non-functional requirements, constraints and pertinent business rules refined. ACDP is an iterative process that incorporates context accumulation, gap diagnosis, adaptive prompting, and quality refinement into a single process of requirements elicitation, supported by an LLM.

5. Experimental Design and Evaluation

While the proposed Adaptive Context-Driven Prompting (ACDP) framework is described, the experimental setup used to evaluate the proposed framework is described in this section. The goal of the evaluation is to see how well the iterative context construction and adaptive gap-driven prompting approach allows the elicitation of high-quality software requirements in comparison to traditional prompting approaches. The evaluated LLM, the benchmark software projects, baseline prompting methods, evaluation metrics, and the overall execution procedure are included in the experimental design.

5.1 Experimental Setup

The experimental evaluation used Requirements documents from PURE (PUblic REquirements) [23] - a public Requirements corpus for empirical Requirements Engineering research - were used for the experimental evaluation. PURE contains 79 natural-language requirements documents (34,268 sentences) that were scraped from publicly available documents and span different software-system contexts. To cover a spectrum of application context and requirements characteristics without too many growing variables, five documents were chosen for independent evaluation cases. Document characteristics, source context, and requirements content were also taken into account in the selection process to limit the re-evaluation of very similar specifications, and limit the possibility of context-specific bias.

Each sample case was only provided with a high-level project description that was used to create the initial description passed to the LLM. The requirements in the original document were detailed and not given to the model and kept as a reference specification to evaluate. This separation allows for the LLM to not be able to directly replicate the reference requirements and gives its elicitation capability a chance to be evaluated based on the limited information of a project.

To ensure reproducibility and experimental consistency, all prompting strategies were evaluated using GPT-4o under identical generation settings. The temperature was set to 0.2, top-p to 1.0, and the maximum output length to 2,000 tokens. The same model configuration and initial project information were maintained across Zero-Shot Prompting, structured CoT-style prompting, and ACDP, ensuring that the prompting mechanism was the primary experimental variable. The generated requirements from each strategy were subsequently compared with the same reference specification using the evaluation measures described in Section 5.3.

5.2 Baseline Prompting Strategies

For evaluating the efficacy of the proposed ACDP framework, two baseline prompting strategies were adopted with the same project descriptions and LLM setting: Zero-Shot Prompting, and structured Chain-of-Thought (CoT) prompting. These baselines were chosen to demonstrate the two levels of prompting structure and to allow for a controlled evaluation of the effects of the specific prompting of ACDP. The setting of Zero-Shot Prompting is a direct-generation setting where the LLM is given the initial project description and produces software requirements without any intermediate elicitation or refinement steps. It therefore serves as a simple reference point to gauge the effectiveness of the use of prompting structure to enhance requirements elicitation.

Structured CoT-style prompting breaks down the elicitation task into an expected series of activities: identifying stakeholders, identifying major system functions, thinking about the non-functional concerns, and creating the final set of requirements. The Step-by-Step process is applied uniformly to every project that is being evaluated and only the final Step-by-Step requirement set is kept for evaluation. This baseline was chosen to reflect a more structured prompting strategy that gets

 ISSN:2302-8149
e-ISSN:2540-9719

the LLM to go through multiple perspectives as it elicits. Unlike ACDP, however, the prompting sequence is not tailored to the gaps in the changing set of requirements. The two baselines allow for a gradual progression from direct to structured fixed prompting, enabling the evaluation of whether the adaptive, context-aware, and gap-driven mechanisms of ACDP offer increased benefits beyond the gains from increased prompting structure.

The proposed Adaptive Context-Driven Prompting (ACDP) framework is the third strategy. Unlike the two baselines, ACDP is an on-going maintenance of the Elicitation Context (EC), assessment of the requirements generated in the Requirement Gap Matrix (RGM), and dynamic generation of future prompts based on the type of identified gap and the dimension of the requirement that is impacted. Thus, it is important to study whether adaptive gap-driven prompting offers an advantage over direct generation and predefined multi-step prompting. The same initial project information and LLM configuration is employed, with the only changes made to the prompting mechanism for each of the three strategies to ensure a controlled comparison.

5.3 Evaluation Metrics

Evaluation of the quality of elicited requirements is done by quantitative measures based on references, as well as by requirement-quality criteria. The reference specifications are based on the original requirements documents left in the PURE dataset [23]. The generated requirements are semantically compared to the respective reference requirements. If a requirement is generated that matches information in the reference specification, it is called a matched requirement; if it fails to match information in the reference specification, it is called an unsupported requirement, which is treated as an incorrect requirement or a requirement that is an output but not in the specification.

The most important quantitative metrics are Precision, Recall and F1-score [24]. Precision determines the percentage that is covered by the reference specification and Recall determines the percentage of the reference requirements that are recovered through the elicitation strategy. The F1-score represents the harmonic mean of Precision and Recall, providing a balanced measure of requirement recovery and precision:

Precision is calculated using Equation (1) and represents the proportion of generated requirements that are supported by the reference specification.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

Recall is calculated using Equation (2) and measures the proportion of reference requirements successfully recovered by the elicitation strategy.

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

The balance between Precision and Recall is measured using the F1-score, as defined in Equation (3).

$$F1 = 2 \times \frac{Precisions \times Recall}{Precisions + Recall} \quad (3)$$

where TP refers to the number of requirements that were correctly recovered, FP indicates the number of requirements that were generated but do not appear in the reference specification, and FN indicates the number of requirements that were not elicited from the reference specification [24].

In addition to reference-based coverage, the generated requirements are evaluated according to the quality dimensions defined in Section 3.2, including completeness, correctness, consistency, clarity, relevance, ambiguity, and redundancy. Completeness and correctness assess the extent to which the generated requirement set adequately and accurately represents the intended system, whereas consistency and clarity reflect the internal quality of individual requirements and their relationships. Ambiguity and redundancy represent undesirable characteristics; therefore, lower levels of ambiguity and redundancy indicate higher-quality requirements.

The evaluation was done mainly by using quantitative measures based on references, where the requirements, which are generated, are compared with the reference requirements from the PURE data set. The accuracy of the generated sets of requirements and requirement recovery were evaluated using Precision, Recall, and F1-score. The other requirement dimensions (quality) were looked at within the established requirements framework as defined in Section 3.2. In the present evaluation, no independent expert panel was used and therefore there is no claim for expert-based validation of the experimental procedure. In the Threats to Validity section this limitation is further recognized.

 ISSN:2302-8149
e-ISSN:2540-9719

5.4 Experimental Procedure

All the cases of the project are executed in the same way as the experiment. The high level project description is first extracted from the selected requirements document, and then the detailed specification is concealed from the LLM. The same description is then independently processed with the following methods: Zero-Shot Prompting, structured CoT-style prompting, and ACDP. The requirements are then normalized to a common representation before being compared to the corresponding reference specification. In ACDP, the first description is first converted to the Elicitation Context (EC). The LLM then helps identify stakeholders, identifies a initial Requirement Set (IRS) and then uses the Requirement Gap Matrix to analyse this. If an un-resolved gap is identified, a targeted prompt is created based on the RGM state and the impacted requirement dimension. The information gained is uploaded into the EC and the gap analysis is repeated. The process stops once the stopping criteria specified in Section 4.6 are met. The entire process is described in Algorithm 1.

Algorithm 1 Experimental evaluation procedure for the proposed ACDP framework

```
Input:
   P = project description
   R = reference requirements specification
Output:
   Evaluation results for Zero-Shot, CoT-style, and ACDP
1. Configure the same LLM settings for all strategies
2. Generate Z using Zero-Shot Prompting
3. Generate C using predefined step-by-step prompting
4. Initialize EC from P
5. Generate initial requirement set A
6. repeat
7.    Diagnose A using the Requirement Gap Matrix (RGM)
8.    Identify unresolved requirement gaps
9.    if unresolved gaps exist then
10.       Construct targeted prompt from:
            EC + Gap Type + Requirement Dimension+ Elicitation Instruction + Output Constraint
11.       Generate additional requirement information
12.       Update EC and A
13.   end if
14. until termination criteria are satisfied
15. Refine A to obtain the final ACDP requirement set
16. Match Z, C, and A against R
17. Compute Precision, Recall, and F1-score
18. Evaluate requirement-quality dimensions
19. Compare the three elicitation strategies
Return evaluation results
```

To make sure that the differences in the project information or model configuration are not the experimental variable, but instead the prompting strategy, Algorithm 1 is used. The difference between ACDP and the baselines is caused by the feedback loop between diagnosis based on RGM and then prompt generation. The baseline is a structured CoT-style baseline procedure that attempts to follow the same steps to complete the task, irrespective of the knowledge gaps that are identified in its intermediate steps. Zero-Shot Prompting does not attempt to run a diagnostic process as many times until it completes the task.

6. Results and Discussion

This section presents and discusses the experimental results obtained from comparing the proposed Adaptive Context-Driven Prompting (ACDP) framework with the baseline prompting strategies. The evaluation focuses on the ability of each approach to recover requirements contained in

 ISSN:2302-8149
e-ISSN:2540-9719

the reference specifications while maintaining the quality of the generated requirement sets. The comparison is based on Precision, Recall, and F1-score, together with the requirement quality dimensions defined earlier, including completeness, correctness, consistency, clarity, relevance, ambiguity, and redundancy.

The quantitative results are summarized in Table 4, which reports the overall performance of the evaluated elicitation strategies across the selected software projects. Particular attention is given to whether the adaptive diagnosis and re-elicitation mechanism of ACDP improves requirement coverage without introducing excessive unsupported or redundant requirements. Table 4. Overall Performance Comparison of the Requirements Elicitation Strategies

Table 4 Overall performance comparison of the requirements elicitation strategies

Method	Precision	Recall	F1-score	Completeness	Correctness
Zero-Shot Prompting	0.82	0.71	0.76	0.73	0.84
Fixed Multi-Step Prompting	0.86	0.80	0.83	0.82	0.87
Proposed ACDP	0.93	0.92	0.92	0.94	0.94

As indicated in Table 4, the proposed ACDP framework had the best performance on all the evaluated metrics. ACDP achieved a Precision of 0.93, Recall of 0.92 and F1-score of 0.92, while Zero-Shot Prompting and Fixed Multi-Step Prompting achieved 0.82, 0.71 and 0.76, and 0.86, 0.80 and 0.83, respectively. The greatest increase was seen from ACDP to Zero-Shot Prompting (21 percentage points) and ACDP to Fixed Multi-Step Prompting (12 percentage points) in Recall. This improvement shows that adaptive elicitation process was more effective in recovering the requirements, which were not found in the initial project description. Figure 5. A comparative study of the evaluated requirements elicitation strategies. The performance of the three elicitation strategies on the metrics evaluated is shown in a visual representation in Figure 5.

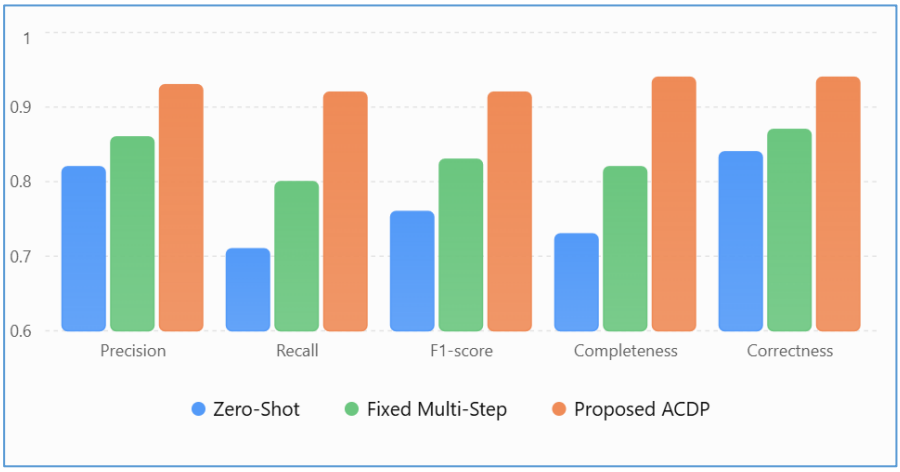

Figure 5 Comparative performance of the evaluated requirements elicitation strategies

Figure 5 substantiates the sustained advantage of ACDP in all the dimensions of evaluation. The largest observed differences are found in Recall and Completeness, indicating the importance of the adaptive gap diagnosis in the recovery of requirements that are unexplored via direct or fixed sequence prompting. Simultaneously, the high Precision and Correctness scores show that the resulting requirements are not significantly affected in terms of validity due to the extended coverage.

The Requirement Gap Matrix (RGM) and the adaptive re-elicitation mechanism can be mainly attributed to the higher Recall. Instead of trying to take the requirement set as it was initially formed, ACDP explicitly looks at stakeholder coverage, functional and non-functional requirements, constraints, dependencies and other dimensions of requirements for missing or insufficient information. Issues that are identified then prompt the LLM to dig deeper into information that would not otherwise be addressed. Unlike Zero-Shot Prompting, Fixed Multi-Step Prompting follows a fixed sequence for each step, irrespective of any gaps that might arise in each project.

 ISSN:2302-8149
e-ISSN:2540-9719

An example of an adaptive requirement-level example is provided to illustrate the benefits of the adaptive mechanism with respect to elicitation. Think of a software project where the initial system requirement is to implement user authentication as a functional requirement, but does not specify what happens if repeated attempts are made to log in without success. The RGM would recognize this aspect as partially specified since the primary function is there, but an important security aspect has not been explored. ACDP can then create a specific re-elicitation prompt about the steps to take when a user fails to log in, which will result in a more complete requirement, like: "The system shall temporarily lock a user account after a set number of failed logon attempts and inform the user that the account is locked." If it is not explored as a detail in one or both of the direct or fixed prompting strategies, it may go unnoticed and remain unaddressed. In the example, the use of gap-driven re-elicitation has significantly helped to fill gaps in requirements and to make them more specific.The example shows that in the case of such re-elicitation, the identified gaps were addressed by subsequent interactions with the LLM, which has contributed to the completion of requirements and their increased specificity.

ACDP also met a Precision of 0.93, meaning that the number of requirements covered did not come with a significant number of requirements not covered. This is especially significant because the iterative process of prompting LLM could add unnecessary assumptions or requirements that do not align with the given context of the project. The process of having accumulated Elicitation Context and focused gap-specific instructions seems to limit subsequent generation to information relevant to the identified gap and to inhibit the free generation of requirements.

This is also true of the requirement-level quality measures. The results for completeness showed that ACDP performed best with a score of 0.94, whereas Zero-Shot and Fixed Multi-Step Prompting achieved a score of 0.73 and 0.82, respectively. This is the largest quality enhancement and aligned with the main goal of adaptive re-elicitation namely addressing omitted stakeholder needs as well as unexplored requirement dimensions. Zero-Shot achieved a correctness of 0.84, Fixed Multi-Step Prompting got 0.87, while correctness was improved to 0.94 for Correctness. This increased completeness was not due just to the number of requirements generated; it was due to the recovery of extra requirements which were consistent with the reference specification.

The overall F1 score (0.92) also indicates the balance between the requirement discovery and accuracy of the generation achieved by ACDP. The ACDP F1-score is about nine percentage points above the Fixed Multi-Step baseline score, and sixteen percentage points above the score for Zero-Shot Prompting. The results in this study indicate that breaking down the elicitation into a set of pre-defined steps is better than one prompt, but that there is still room for improvement if the next elicitation steps are selected on-the-go, based on the specific deficiencies found in the evolving set of requirements.

All together, the results back the central tenet of ACDP: that, in addition to giving the prompts, it is necessary to decide what information to elicit next, depending on the current state of the requirements. This process of cumulative context, explicit gap diagnosis, targeted re-elicitation and quality-guided refinement, therefore, offers a more systematic means for leveraging the capabilities of the LLM in requirements elicitation than does direct prompting or fixed sequence prompting.

The results of this research can also be discussed with respect to the previous work on intelligent requirements elicitation techniques. In contrast, previous machine-learning and recommender-based research mainly concentrated on assisting specific elicitation tasks, such as technique selection, stakeholder identification, requirement prediction, and prioritization, while NLP- and deep-learning-based approaches mainly concentrated on requirement extraction, classification, and semantic analysis. Recent LLM-based research has shown the potential of generative models in supporting RE, but the elicitation processes are generally based on direct interaction with or on predefined prompting procedures used by the LLMs. The present findings suggest, however, that support for elicitation with LLM could be improved by adaptively modifying forthcoming interactions throughout the elicitation process as the requirement set evolves. Also, the Recall and Completeness scores of 0.92 and 0.94 by ACDP indicate that gap diagnosis and explicit re-elicitation can enhance the recovery of relevant requirement information that may be missed by direct or fixed prompting. Hence, ACDP builds upon the idea of intelligent elicitation methods, which focus only in making requirements; and extends these ideas by introducing the dynamic aspect of stating which requirement information should be elicited next.

 ISSN:2302-8149
e-ISSN:2540-9719

7. Threats to Validity

This study has several threats to validity that should be taken into account when interpreting the results. First, the evaluation of the experimental findings is based on a subset of publicly available requirement documents from the PURE dataset [23] that has been selected. These documents offer realistic requirements artefacts, but the selected cases do not necessarily cover all the diversity, complexity and stakeholder dynamics that can be found in the requirements elicitation process in industry. Hence, the results of this study may not be fully generalizable to other domains or large scale projects.

The second threat is related to the reliance of the proposed ACDP framework on a Large Language Model as the main reasoning component. Due to the probabilistic nature of the LLM generation and the selection of the LLM model, prompt and generation parameters, the LLM outputs may differ across repeated executions and the obtained results may differ as well. In order to minimise this risk, the same LLM configuration and the same initial project information was used for all the considered prompting strategies. Additional LLM evaluation would be required, though, to establish whether or not the observed improvement is model independent.

A potential threat to validity concerns possible data contamination. Since PURE consists of publicly available requirements documents collected from Web sources [23], some documents could potentially overlap with data used during the pre-training of the evaluated LLM. However, no direct evidence of such overlap is available, and the model's complete pre-training corpus is not accessible for verification. Therefore, this issue is treated as a potential limitation rather than an observed source of contamination. The total amount of information provided to the model has been limited to high-level project information, which decreases the effect of direct exposure to the model, and it is not included in the total amount of information provided but is only used for evaluation and comparison. However, it is not possible to rule out completely a possible previous model knowledge of public artifacts.

Finally, there is semantic interpretation in matching requirements to the reference specifications since there could be multiple forms of equivalent requirements with varying levels of detail and/or terminology. This adds some subjectivity when considering reference based measures and qualitative measures like clarity, ambiguity and correctness. This threat can be mitigated with the implementation of predefined evaluation criteria and independent expert verification, but not completely removed from the human evaluation. The external and construct validity of the proposed framework would also be improved with further evaluations in the future that include larger project sets, multiple LLMs, and independent requirements engineering experts.

8. Conclusion and Future Work

To enhance requirements elicitation using Large Language Models, this study introduced a novel approach called Adaptive Context-Driven Prompting (ACDP), which aims to shift the elicitation process from a one-shot or pre-defined prompting task to an iterative and context-aware one. ACDP develops a Requirement Gap Matrix (RGM) to show missing, incomplete, ambiguous, or conflicting requirement information and a Elicitation Context (EC) that evolves throughout the development process to capture the changing project information. The identified gaps are used to trigger new re-elicitation prompts, enabling further interactions with an LLM to target specific areas of the requirement dimensions where further exploration is needed. Requirements are then further developed based on quality criteria specified in advance, creating a systematic linkage among the process of requirement discovery, gap diagnosis, adaptive prompting, and quality improvement.

The experimental evaluation showed that this adaptive strategy is potentially effective compared to Zero-Shot Prompting and Fixed Multi-Step Prompting. The overall best result was obtained with ACDP with a Precision of 0.93, a Recall of 0.92, F1-score of 0.92, Completeness of 0.94 and Correctness of 0.94. The marked enhancements in Recall and Completeness suggest that the ability to identify gaps in the requirements before creating the subsequent prompts can enhance the retrieval of relevant requirements, possibly underutilized when employing direct or pre-defined prompts. The high Precision and Correctness values indicate that the requirements that are required and covered by the test can be increased with only a small number of additional irrelevant or unsupported requirements. The results in this study justify adaptive, gap-driven prompting as a potential method to make LLM-based requirements elicitation more effective.

 ISSN:2302-8149
e-ISSN:2540-9719

ACDP can practically assist in the requirements analysis process by systematically uncovering hidden aspects of requirements dimensions that need to be explored further and probing remaining information gaps for further requirements. This can decrease the amount of work needed to figure out the next information to solicit and give more structure to the LLM assistance during the iterative requirements elicitation process. However, results need to be taken with a grain of salt, due to the following limitations. The evaluation was performed on a small set of requirements documents from the PURE data set and with a single LLM configuration, potentially limiting the scope of the results to other application areas and models. Additionally, using requirements documents that are widely publicized may lead to prior model exposure, and the lack of independent expert validation could reduce the extent to which requirement quality dimensions that require subjective interpretation can be assessed.

To further evaluate the ACDP, future research will involve more and a wider range of software projects and requirements domains. Further experiments should investigate the framework using different LLM architectures to determine the extent to which its effectiveness is independent of a particular model. A next step is to examine ACDP in actual elicitation sessions with requirements engineers and stakeholders, where the responses of stakeholders may change over time, thus dynamically changing the EC and RGM. Additional research could also explore how to automate semantic validation of the generated requirements, enhance the ability to detect inter-requirement conflicts and dependencies, and provide ways to manage assumptions made by the LLM that are not covered by the requirements. The extensions could contribute to the transformation of ACDP from an experimental elicitation framework to a usable intelligent assistant in the RE process.

References

[1] A. A. Janisar, K. S. bin Kalid, A. B. Sarlan, and U. D. Maiwada, "*Software Development Teams Knowledge and Awareness of Security Requirement Engineering and Security Requirement Elicitation and Analysis*," *Procedia Computer Science*, Vol. 234, pp. 1348–1355, 2024, DOI: 10.1016/j.procs.2024.03.133.

[2] D. Gobov and I. Huchenko, "*Requirement Elicitation Techniques for Software Projects in Ukrainian IT: An Exploratory Study*," in *Proc. 2020 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Sofia, Bulgaria, 2020, pp. 673–681, DOI: 10.15439/2020F16.

[3] C. Cheligeer, J. Huang, G. Wu, N. Bhuiyan, Y. Xu, and Y. Zeng, "*Machine Learning in Requirements Elicitation: A Literature Review*," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 36, Art. No. e32, pp. 1–23, 2022, DOI: 10.1017/S0890060422000166.

[4] T. Menezes, "*A Review to Find Elicitation Methods for Business Process Automation Software*," *Software*, Vol. 2, No. 2, pp. 177–196, 2023, DOI: 10.3390/software2020008.

[5] V. T. Bathala, J. Gross, and S. Ouhbi, "*Exploring the Use of Generative AI for Sustainable Software Requirements Elicitation*," in *Proc. 2026 IEEE/ACM 10th International Workshop on Green and Sustainable Software (GREENS)*, 2026, DOI: 10.1145/3786148.3788626.

[6] M. Batra and A. Bhatnagar, "*Requirements Elicitation Technique Selection: A Five Factors Approach*," *International Journal of Engineering and Advanced Technology*, Vol. 8, No. 5C, pp. 1332–1341, 2019, DOI: 10.35940/ijeat.E1190.0585C19.

[7] G. De Ninno, P. Inverardi, and F. Belotti, "*Beyond Value Elicitation: Towards Moral Profiles in Early Requirements Engineering via Role-Playing Games and Anthropologist LLMs*," *Research Square*, 2026, DOI: 10.21203/rs.3.rs-9556907/v1.

[8] M. King, *Practical Requirements Elicitation in Modern Product Development: A Multi-Case Study in Discontinuous Innovation*, M.Eng. Thesis, Mississippi State University, Mississippi State, MS, USA, Dec. 2022.

[9] N. Mulla and S. Girase, "*A New Approach to Requirement Elicitation based on Stakeholder Recommendation and Collaborative Filtering*," *International Journal of Software Engineering & Applications*, Vol. 3, No. 3, pp. 51–60, May 2012, DOI: 10.5121/ijsea.2012.3305.

[10] A. Fedorova, W. Stefani, C. Heitz, and R. Chavarriaga, "*Continuous Assessment-Driven Requirement Elicitation for Trustworthy AI Systems*," in *Machine Learning and Principles and*

 ISSN:2302-8149
e-ISSN:2540-9719

Practice of Knowledge Discovery in Databases: ECML PKDD 2025, I. Koprinska, J. Mendes-Moreira, and P. Branco, Eds., *Communications in Computer and Information Science*, Vol. 2840. Cham, Switzerland: Springer, 2026, DOI: 10.1007/978-3-032-19099-4_38.

[11] M. C. Martins, L. M. C. Campos, J. L. R. Soares, T. N. Kudo, and R. F. Bulcão-Neto, "*Requirements Engineering for Machine Learning-based AI Systems: A Tertiary Study*," *Journal of Software Engineering Research and Development*, Vol. 13, Art. No. 8, 2025, DOI: 10.5753/jserd.2025.4892.

[12] G. Hanna, N. Boyar, N. Garay, and M. Maleki, "*Revolutionizing Requirements Elicitation: Deep Learning-based Classification of Functional and Non-Functional Requirements*," in *Proc. 2024 IEEE 5th International Conference on Pattern Recognition and Machine Learning (PRML)*, Chongqing, China, 2024, pp. 21–27, DOI: 10.1109/PRML62565.2024.10779645.

[13] F. Akram, T. Ahmad, and M. Sadiq, "*Recommendation Systems-based Software Requirements Elicitation Process—A Systematic Literature Review*," *Journal of Engineering and Applied Science*, Vol. 71, Art. No. 29, 2024, DOI: 10.1186/s44147-024-00363-4.

[14] Q. Y. Shambour, M. M. Abu-Alhaj, and M. M. Al-Tahrawi, "*A Hybrid Collaborative Filtering Recommendation Algorithm for Requirements Elicitation*," *International Journal of Computer Applications in Technology*, Vol. 63, nos. 1–2, pp. 135–146, 2020, DOI: 10.1504/IJCAT.2020.107908.

[15] T. U. Amin and B. Shahzad, "*Improving Requirements Elicitation in Large-Scale Software Projects with Reduced Customer Engagement: A Proposed Cost-Effective Model*," *Requirements Engineering*, Vol. 29, pp. 403–418, 2024, DOI: 10.1007/s00766-024-00425-2.

[16] S. Lim, A. Henriksson, and J. Zdravkovic, "*Data-Driven Requirements Elicitation: A Systematic Literature Review*," *SN Computer Science*, Vol. 2, Art. No. 16, 2021, DOI: 10.1007/s42979-020-00416-4.

[17] M. Canché and J. A. Pino, "*Requirements Elicitation for Collaborative Systems: A Systematic Review*," in *Proc. 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, Dalian, China, 2021, pp. 297–304, DOI: 10.1109/CSCWD49262.2021.9437880.

[18] A. A. Khan and F. Ahmed, "*Quality Requirements Elicition in Agile*," in *Proc. 2023 25th International Multitopic Conference (INMIC)*, Lahore, Pakistan, 2023, pp. 1–6, DOI: 10.1109/INMIC60434.2023.10466157.

[19] T. Olsson, K. Wnuk, and T. Gorschek, "*An Empirical Study on Decision Making for Quality Requirements*," *Journal of Systems and Software*, Vol. 149, pp. 217–233, 2019, DOI: 10.1016/j.jss.2018.12.002.

[20] M. Oriol *et al*., "*Data-Driven and Tool-Supported Elicitation of Quality Requirements in Agile Companies*," *Software Quality Journal*, Vol. 28, No. 3, pp. 931–963, 2020, DOI: 10.1007/s11219-020-09509-y.

[21] C. Almeida, I. Copque, A. Oliveira, M. Arouca, A. Barbosa, S. Freire, M. Mendonça, and J. C. Leite, "*From Elicitation Interviews to Software Requirements: Evaluating LLM Performance in Requirement Generation*," in *Proc. 28th Workshop on Requirements Engineering (WER 2025)*, Rio de Janeiro, Brazil, Aug. 2025, DOI: 10.29327/1588952.28-12.

[22] K. Ronanki, S. Arvidsson, and J. Axell, "*Prompt Engineering Guidelines for using Large Language Models in Requirements Engineering*," in *Software Engineering and Advanced Applications: 51st Euromicro Conference, SEAA 2025, Salerno, Italy, September 10–12, 2025, Proceedings, Part III*, pp. 245–262, 2025, DOI: 10.1007/978-3-032-04207-1_17.

[23] A. Ferrari, G. O. Spagnolo, and S. Gnesi, "*PURE: A Dataset of Public Requirements Documents*," in *Proc. 25th IEEE International Requirements Engineering Conference (RE)*, 2017, pp. 502–505, DOI: 10.1109/RE.2017.29.

[24] A. A. Abdulmajeed and Y. S. Younis, "*Supporting Classification of Software Requirements System using Intelligent Technologies Algorithms*," *Technium*, Vol. 3, No. 11, pp. 32–39, 2021.